1 

2 # Reader Management 1.0.1

3 May 31, 2007

Abstract

This document defines Version 1.0 of the wire protocol used by management software to monitor the operating status and health of EPCglobal compliant RFID Readers. This document complements the EPCglobal Reader Protocol Version 1.1 specification [RP1]. In addition, this document defines Version 1.0 of the EPCglobal SNMP RFID MIB.

Audience for this document

The target audience for this specification includes:

- EPC Middleware vendors
- Reader vendors
- Application developers
- Management Software developers
- System integrators

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained by EPCglobal. This version of the document provides minor updates to the SNMP since the Standard was first ratified. The table below summarizes changes to the document since the ratification of Reader Management 1.0.

Date and Version Number	Section(s)	Changes	Approved By
May 31, 2007 1.0.1	10.4.2	- Removed some inconsistencies between Abstract Model and MIB related to Alarm Controls (removed antenna-readpoint level suppression counters and controls and put them at the per-operation level where the abstract model calls for them - Added EpcgAdministrativeStatus type - Added ioValue enumeration to	

		EpcgTriggerType
December 5, 2006 1.0	All	Initial revision

Comments on this document should be sent to the EPCglobal Software Action Group Reader Management Working Group mailing list sag_rm_wg@lists.ecpcglobalinc.org.

Table of Contents

1 Introduction

This document defines Version 1.0 of the wire protocol used by management software to monitor the operating status and health of EPCglobal compliant tag Readers. This document complements the EPCglobal Reader Protocol Version 1.1 specification [RP1]. In addition, this document defines Version 1.0 of the EPCglobal SNMP RFID MIB, and specifies the set of SNMP MIBII groups [MIBII] required to comply with this EPCglobal Reader Management Specification over SNMP. The terms "tag Reader" and "Reader" include RFID tag Readers, supporting any combination of RF protocols, fixed and hand-held, etc. It also includes Readers of other kinds of tags, such as bar codes. Tag Readers, despite the name, may also have the ability to write data into tag memory.

The Reader Management Protocol specifies the interaction between a device capable of interfacing with tags, and management software. These two parties are herein referred to as the Reader and the Host. The Host may be a fully featured Management Console capable of processing SNMP messages, or a dedicated application capable of communicating with the Reader to interface with RFID tags and monitor its health.

The collection of tag data between the Reader and the Host is defined in the EPCglobal Reader Protocol Version 1.1 [RP1]. This document focuses on the communication protocol required to monitor the health of the Reader.

2 Terminology

Within this specification, the terms SHALL, SHALL NOT, SHOULD, SHOULD NOT, MAY, NEED NOT, CAN, and CANNOT are to be interpreted as specified in Annex G of the ISO/IEC Directives, Part 2, 2001, 4th edition [ISODir2]. When used in this way, these terms will always be shown in ALL CAPS; when these words appear in ordinary typeface they are intended to have their ordinary English meaning.

All sections of this document are normative, with the exception of where it is explicitly noted as non-normative.

The following typographical conventions are used throughout the document:

- ALL CAPS type is used for the special terms from [ISODir2] enumerated above.
- `Monospace` type is used to denote programming language, UML, XML identifiers and other coding constructs, as well as for the text of XML documents.

3 Protocol Layers

This document defines two separate but related management protocol specifications.

1. Specifies the EPCglobal SNMP MIB for monitoring the health of a Reader.
2. Specifies the EPCglobal Reader Management Protocol for monitoring the health of a Reader.

Number 2 follows the same layering structure defined by the EPCglobal Reader Protocol Version 1.1 [RP1]. The Reader Management Protocol is specified in three distinct layers, as illustrated below.

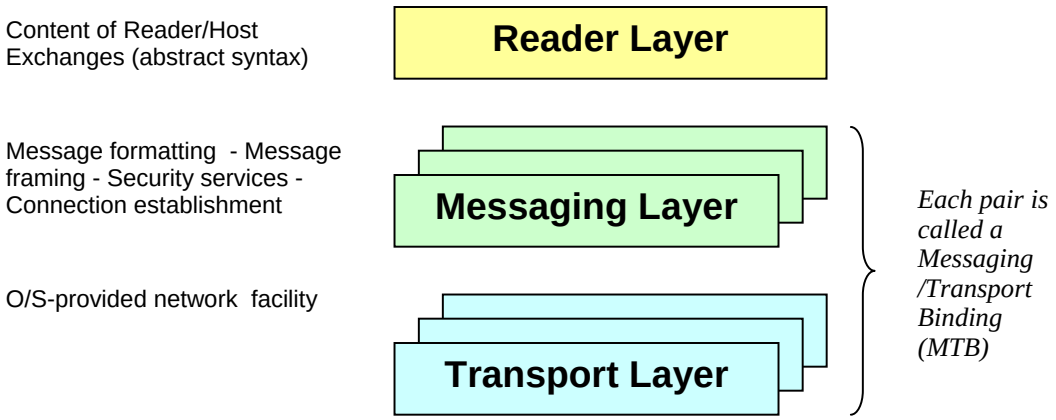


Figure 1 Protocol Layers

The layers are:

- *Reader Layer* This layer specifies the content and abstract syntax of messages exchanged between the Reader and Host. This layer is the heart of the Reader Protocol and the Reader Management Protocol, defining the operations that Readers expose to monitor their health.
- *Messaging layer* This layer specifies how messages defined in the Reader Layer are formatted, framed, transformed, and carried on a specific network transport. Any security services are supplied by this layer. (Examples of security services include authentication, authorization, message confidentiality, and message integrity.) The Messaging Layer specifies how an underlying network connection is established, any initialization messages required to establish synchronization or to initialize security services, and any processing such as encryption that is performed on each message.
- *Transport Layer* This layer corresponds to the networking facilities provided by the operating system or equivalent, and is specified elsewhere.

The Reader Management Protocol specification provides for multiple alternative implementations of the Messaging Layer. Each such implementation is called a Messaging/Transport Binding (MTB). Different MTBs provide for different kinds of transport, e.g., TCP/IP versus Bluetooth versus serial line. Different MTBs may also provide different kinds of security services, means for establishing connections (e.g., whether the Reader contacts the Host or the Host contacts the Reader), and means for provisioning of configuration information.

This specification defines an XML MTB which complements that defined by the Reader Protocol. The Transport details can be found in the Reader Protocol Specification 1.1. The SNMP MIB components are also defined in detail in this specification.

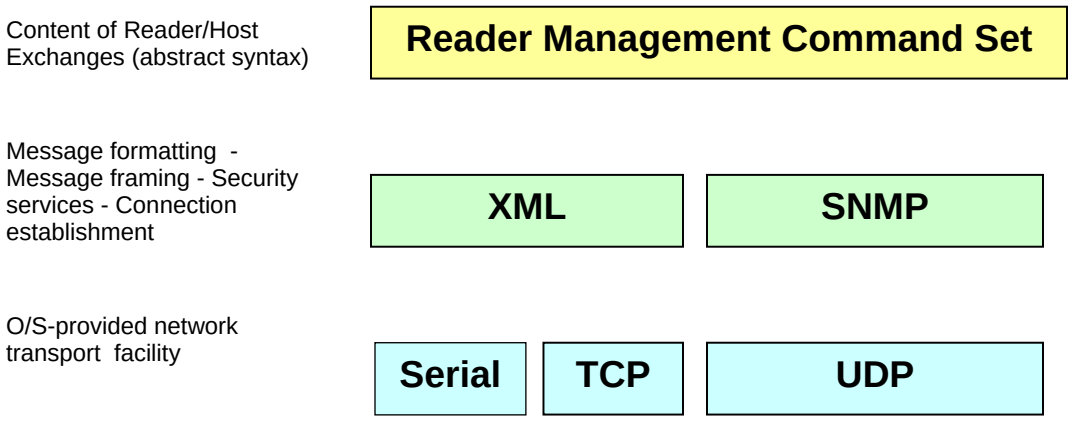


Figure 2 Protocol Layers Mapping

The SNMP MIB is an example of another Message/Transport binding. The MIB is a structuring and representation of Reader Object Model elements that conforms to the SNMP specification "Structure of Management Information Version 2 (SMIv2)" [STD 58, RFC 2578]. The SNMP protocol has a well defined messaging protocol and transport layer for getting and setting information, event notification, and security facilities. The figure above depicts two MTB examples. On the left, the Reader Protocol Message Format using XML as input and output, and TCP as the transport. On the right, SNMP defining the Message formatting and UDP as the message transport. Note that the same Reader Management Command Set applies to both MTBs.

The interface between the Reader Layer and the Messaging Layer is defined in terms of message channels, each representing an independent communication between Reader and Host. The Reader Management Protocol leverages the message channels defined in the EPCglobal Reader Protocol Specification 1.1:

- *Command Channel* The command channel carries requests issued by the Host to the Reader, and responses to these requests carried from the Reader to the Host. All messages exchanged on the command channel follow this request/response pattern. Most configuration interaction between Reader and Host takes place through the command channel. The Reader MAY support one or more *Command Channel*s.
- *Alarm Channel(s)* The alarm channel carries messages issued asynchronously by the Reader to the Host. Messages on the alarm channel only flow to the Host. The alarm channel is primarily used to support a mode of operation in which the Reader asynchronously delivers health and status alerts to the Host, The Reader controls when alarm channel messages are sent to the Host; the Host does not request alarm channel messages. The Reader MAY support one or more Alarm Channels for the delivery of Management Alarms.

- *Notification Channel(s)* In addition, the Reader MAY support one or more Notification Channels for delivery of tag data, as specified in the Reader Protocol Specification 1.1.

The Command channel provides the means to set and get the state, status, or value of attributes of a given object. Under certain conditions, the end-user may choose to disable the setter capabilities, therefore a Reader MAY provide the means to globally disable all *set* methods for all objects. If a set method is invoked after it has been disabled, the ERROR_AUTHORIZATION SHALL be returned.

4 Object Model

The Reader Management Specification defines a set of conceptual objects and operations (a command set) which enables the Host to query the status of these objects in a standard way. This is described using an object model. The Reader Management specification shares a common object model with the Reader Protocol Specification 1.1; however some objects and attributes are not relevant to the Reader Management specification. Figure 3 Reader Monitored Objects UML **Error! Reference source not found.**Figure 3 Reader Monitored Objects UML**Error! Reference source not found.**through

Figure 14 Enumerations UML

Figure 14 Enumerations UML present all of the objects in order to provide a complete model. Only the objects, attributes, and methods relevant to Reader Management are discussed in detail in this specification. Objects that are not relevant to this specification are shown in gray. Object attributes and methods from the Reader Protocol Specification 1.1 and this Reader Management specification are shown in the detailed UML diagrams in Figure 4 ReaderDevice Object UML, Figure 5 Source Object UML, Figure 6 ReadPoint Object UML, Figure 8 Trigger Object UML, and Figure 11 NotificationChannel Object UML. The specific commands defined by this Reader Management specification are listed in detail in Section 5 Reader Layer – Commands, section 6 Reader Layer – Alarm Notifications, and section 7 Enumerated types of this document. Refer to the Reader Protocol Specification 1.1 for a description of the attributes and commands specific to that specification. A compliant Reader NEED NOT implement an object model internally, as long as the command messages can be correctly interpreted and executed, and alarms be correctly generated.

The health of a Reader can be monitored by actively querying Reader for the status of the various monitored objects, and/or by setting asynchronous notification alarms. Each object maintains a set of counters and status monitors that can be queried synchronously. Some objects also implement alarms that are configured to be triggered on a given set of conditions. The AlarmControl Objects define when alarms are generated. The Alarm Objects represent the actual content of the notification or alarm. AlarmControl Objects are defined in Section 4.2. Alarm Objects are defined in Section 4.3.

The following section provides a brief description of the major objects and their relationship to each other. For details on the objects and their operations, refer to Section 5: Reader Layer – Commands

4.1 Monitored Objects

The following sections provide a brief description of the conceptual objects. Each object exposes a set of operations which are described in detail throughout this specification. Figure 3 Reader Monitored Objects UMLFigure 3 Reader Monitored Objects UML illustrates the high level relationship between the components. The list of attributes and commands defined for each object are illustrated in each of the sections describing the objects. Objects with no visible attributes and/ or commands, all in white, are illustrated in further detail in a subsequent UML diagram in the specification.

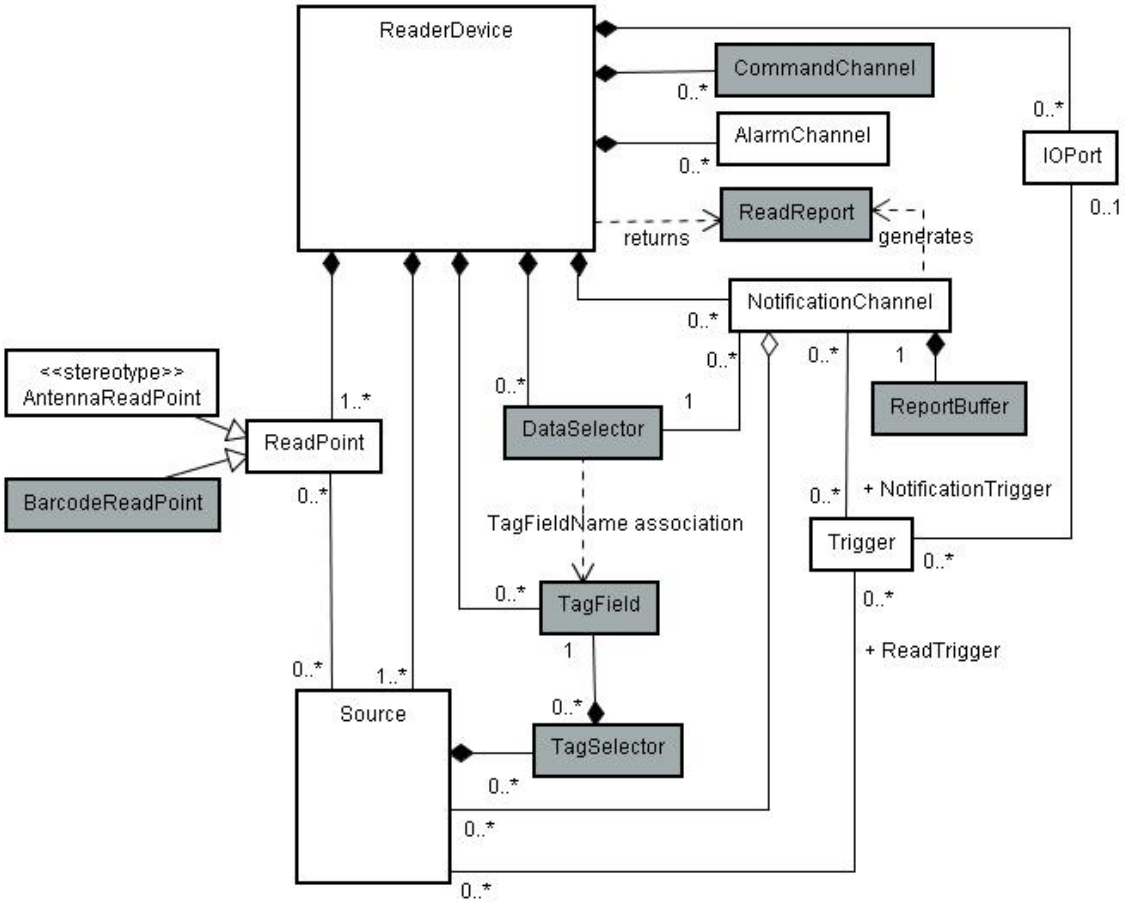


Figure 3 Reader Monitored Objects UML

4.1.1 ReaderDevice Object

The `ReaderDevice` object is the base container for most other objects in the Reader object model including at least one pre-configured `CommandChannel` object. The `ReaderDevice` object also contains several attributes that are used to manage the Reader device and, for network Readers, attributes used to manage the Reader's network interface.

388

389 **Figure 4 ReaderDevice Object UML**

4.1.2 CommandChannel

There are no explicit operations on this object. A compliant Reader MAY have one or more `Command Channels` with a default address where it is listening to incoming commands. If implemented, this default address SHALL be listed in the documentation of a compliant reader.

4.1.3 Source Object

`Sources` read tags and present their data to the Data Acquisition stage of the Read Subsystem in a Reader. Two examples of `Sources` are an RF antenna and a barcode scanner. `Source` objects are associated to the `ReaderDevice` object. One or more `Sources` MAY be associated with a `NotificationChannel`. In the Reader object model, a `Source` object can encapsulate one or more input sensors called `ReadPoints`. Refer to the Reader Protocol Specification 1.1 for further detail.

Figure 5 Source Object UML

4.1.4 ReadPoint Object

A `ReadPoint` can be any physical entity that is capable of acquiring data. A single RF tag reader antenna is a simple example of a `ReadPoint`. In this version of the specification, the only supported `ReadPoint` is an `AntennaReadPoint`. Refer to the Reader Protocol Specification 1.1 for further detail.

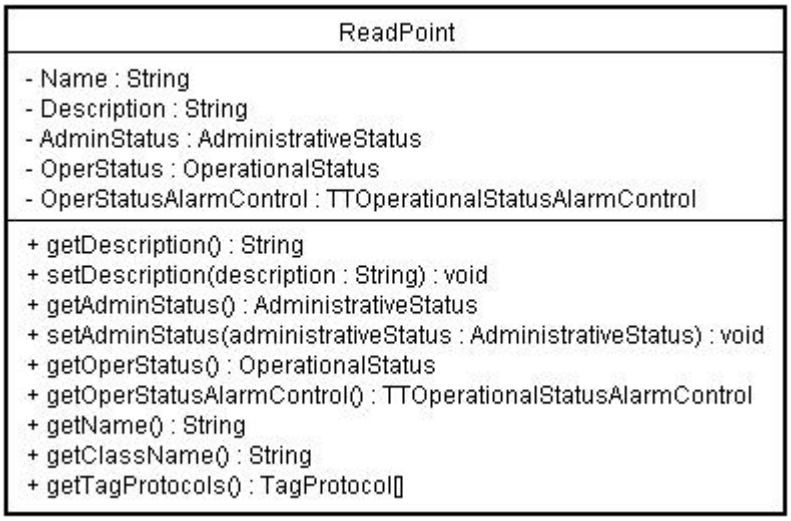


Figure 6 ReadPoint Object UML

4.1.5 AntennaReadPoint Object

`AntennaReadPoints` are `ReadPoints` which are physically implemented using antennas. Each `AntennaReadPoint` maintains a number of statistics to assist in monitoring the health of the reader. Refer to the Reader Protocol Specification 1.1 for further detail.

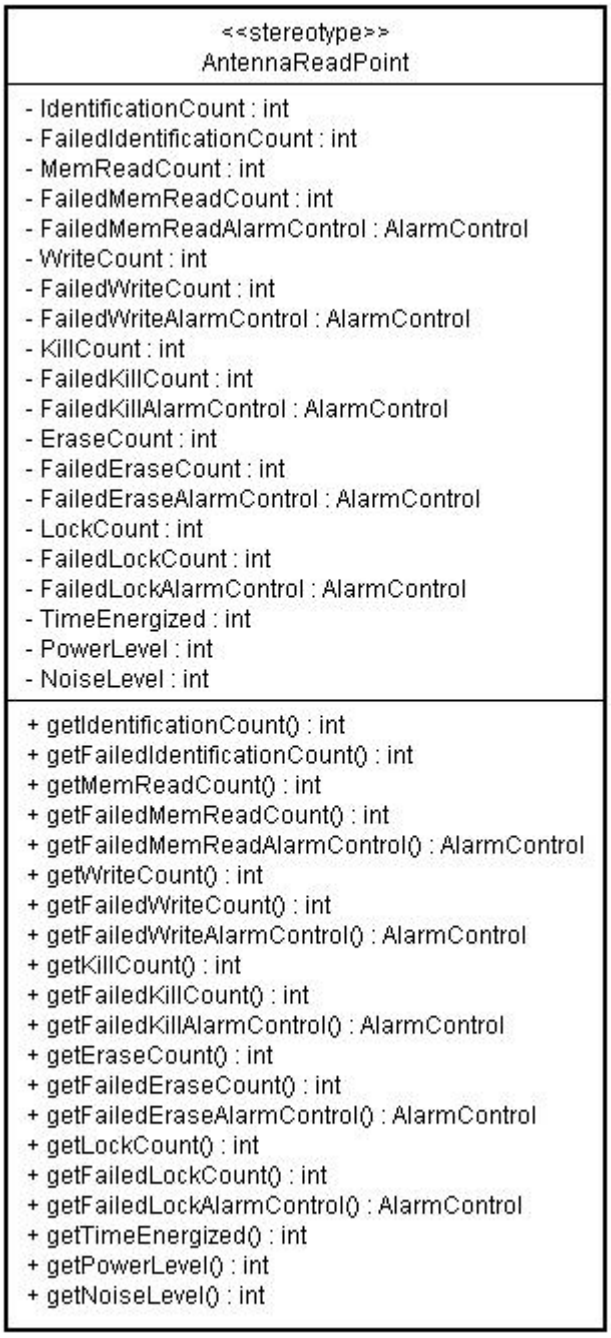


Figure 7 AntennaReadPoint Object UML

4.1.6 Trigger Object

`Trigger` is a non-mutable object that can exist as a composite member of either a `Source` or a `NotificationChannel` object, i.e., `Trigger` objects are added and removed from `Source` and `NotificationChannel` objects. Refer to the Reader Protocol Specification 1.1 for further detail.

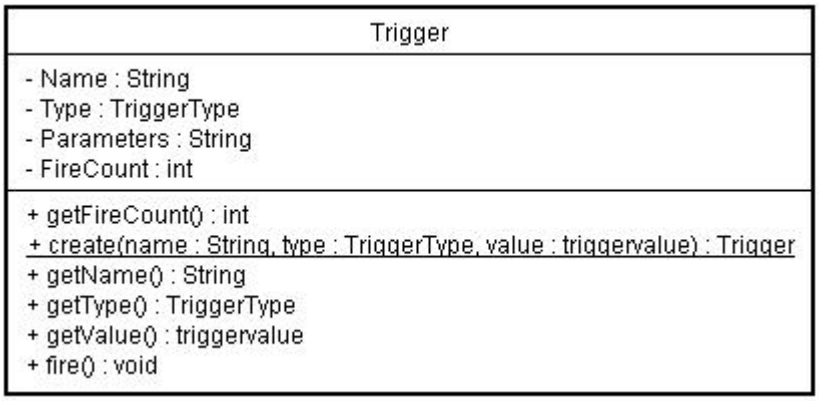


Figure 8 Trigger Object UML

4.1.7 IOPort Object

An `IOPort` provides the description of the hardware element that provides external input and output lines to connect to other components outside the reader device. A hardware vendor is free to define an `IOPort` as it best fits its needs. For example, one vendor may choose to define a single `IOPort` for a 9-pin RS232 connector used to interface with light switches or motion sensors. A second vendor may choose to assign a unique `IOPort` to each of the pins, if it is capable of configuring and monitoring the state of each pin independently. If a reader device exposes its `IOPort`s, it SHALL implement the methods identified as such. Refer to the Reader Protocol Specification 1.1 for further detail.

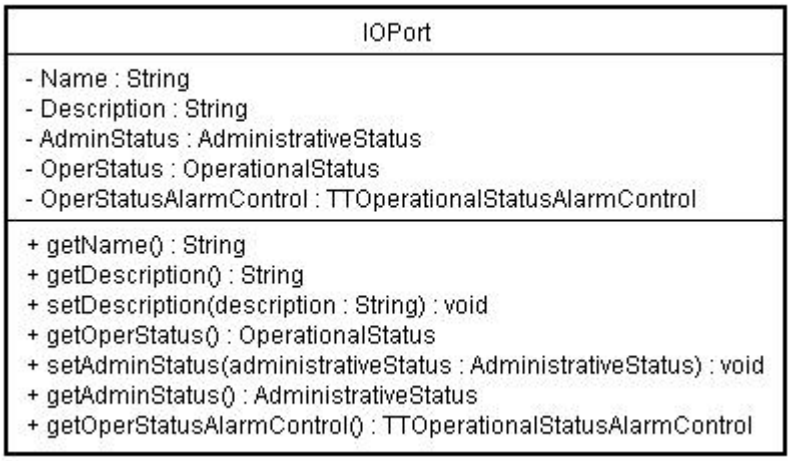


Figure 9 IOPort Object UML

4.1.8 AlarmChannel Object

The `AlarmChannel` carries messages issued asynchronously by the Reader to the Host. Messages on an `AlarmChannel` only flow in this direction. `AlarmChannels` MAY be implemented if a Reader is capable of notifying a host of changes in its health status.

It is strongly advised, but not required, to create/define `AlarmChannels` using the commands specified in this specification. A compliant system SHALL provide the means to create/ define `AlarmChannels` either using the commands specified in this specification, or through a Reader specific means, such as through the use of an out-of-band protocol, or via a configuration file.

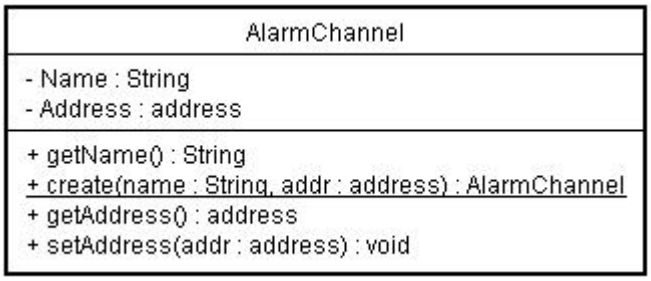


Figure 10 AlarmChannel Object UML

NotificationChannel Object

A `NotificationChannel` object is responsible for communicating information to a host asynchronously. One or more Sources MAY be associated with a NotificationChannel. A DataSelector determines what data is reported to the host. Refer to the Reader Protocol Specification 1.1 for further detail.

Figure 11 NotificationChannel Object UML

4.2 AlarmControl Objects

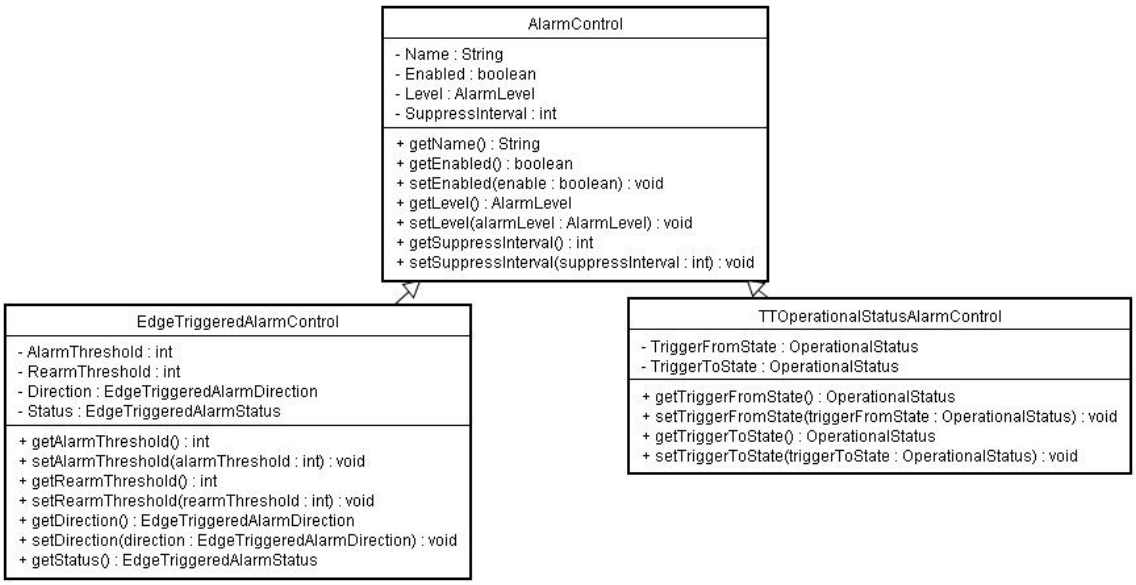
`AlarmControl` is the base class for all classes responsible for controlling the generation of `Alarm`s. `AlarmControls` SHALL be implemented if `AlarmChannels` are implemented. In order to configure an `Alarm` on an object, the monitoring host needs to obtain a reference to the `AlarmControl` and manipulate its attributes. An `Alarm` object represents the actual notification sent to the monitoring host. The UML for the `AlarmControl` objects is illustrated in Figure 12 Alarm Control UML.

There are two types of `AlarmControls` defined by the subclasses:

- `EdgeTriggeredAlarmControl`
-
- `TTOperationalStatusAlarmControl`

467 •

468



469

470 **Figure 12 Alarm Control UML**

471

472 ## 4.3 Alarm Objects

473 The alarms themselves are messages that are sent to management systems and host
474 applications. `Alarm` objects specify, in an abstract manner, the contents of these alarm
475 messages. These `Alarm` objects do not specify persistent data objects maintained by the
476 readers; rather they are instantiated in the course of a reader's generation or a
477 management system's receipt and processing of alarms. `Alarm` objects SHALL be
478 implemented if `AlarmChannel`s are implemented. The UML for the `Alarm` objects is
479 illustrated in Figure 13 Alarm UML.

480

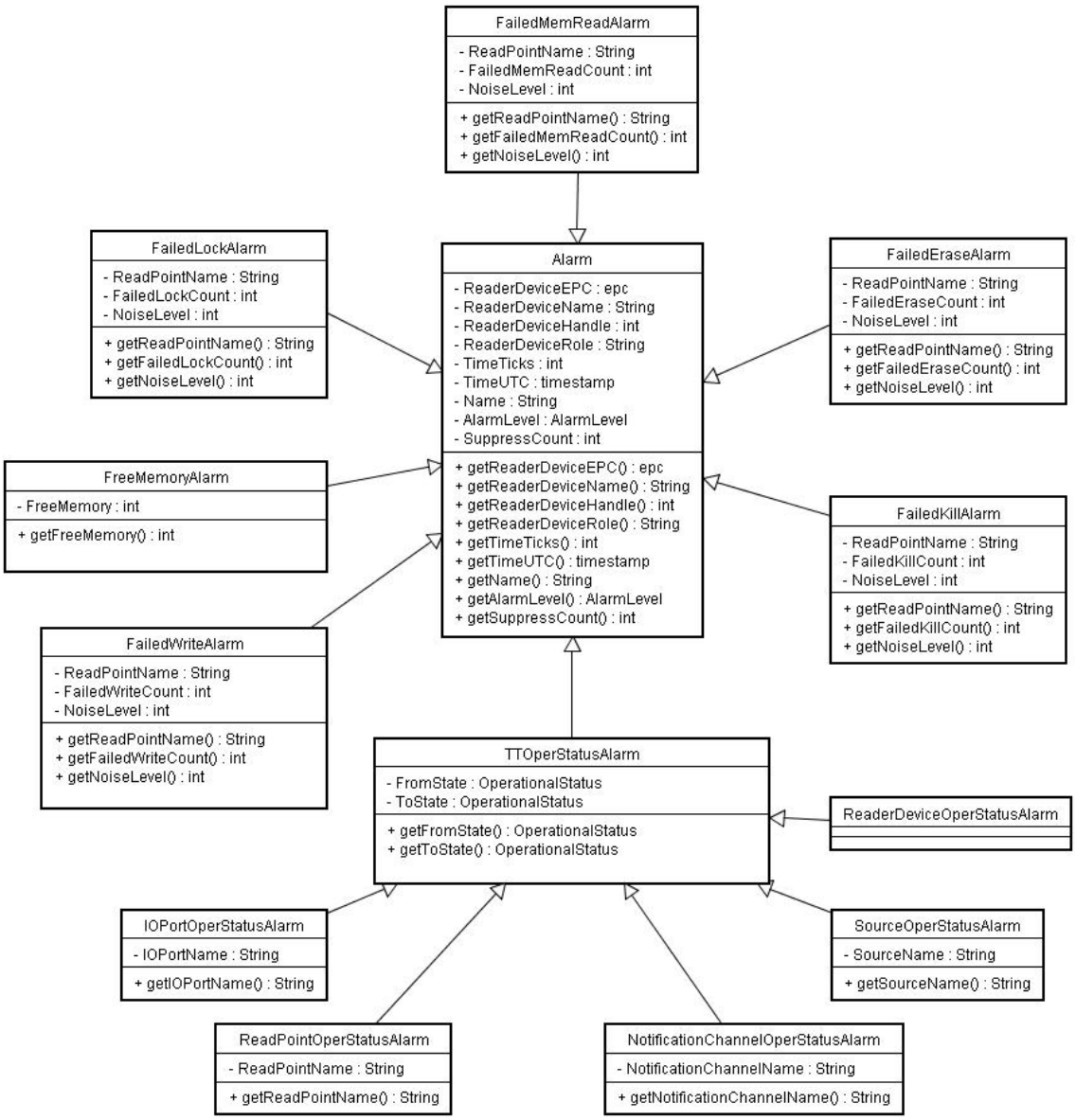


Figure 13 Alarm UML

4.4 Enumerations

The Enumerations contain sets of values used by other objects, such as the administrative Status, Operational Status, Alarm levels, the status of EdgeTriggered Alarms, directions of an alarm value, and the list of operators used to compare alarm levels. The UML for the Enumerations is illustrated in

Figure 14 Enumerations UML

Figure 14 Enumerations UML.

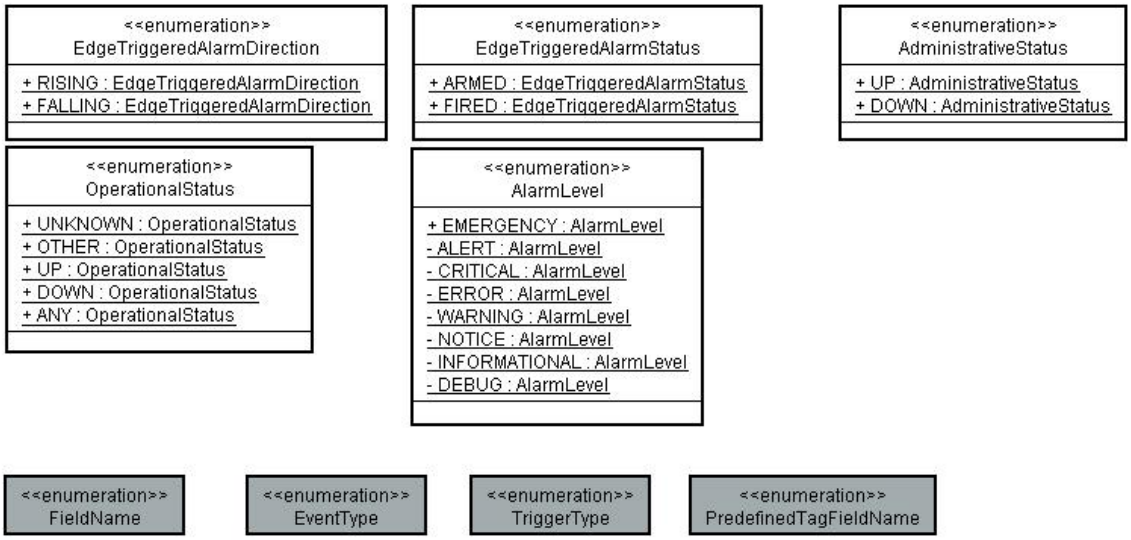


Figure 14 Enumerations UML

5 Reader Layer – Commands

This Reader Management Specification defines a set of operations (a command set) which enables the Host to query the status of these objects in a standard way.

This section specifies messages exchanged at the Reader Layer in order to monitor the health of the Reader.

Messages on the command channel follow a request/response pattern, where the Host sends a request message, and the Reader responds later with a response message (be it a normal response or error response). Messages on the alarm channel are sent asynchronously from Reader to Host.

Reader Layer messages fall into the main functional groups defined by the objects in the previous section. The messages described in this specification are complementary to those defined in the Reader Management section of the Reader Protocol Version 1.1 Specification.

Commands are described below in an abstract manner using a UML notation. This notation is identical to that used by the Reader Protocol Version 1.1.

Commands fall into three categories:

- DO-type commands that cause the Reader to take action.

- SET-type commands that cause the Reader to change internal state variables.
- GET-type commands that cause the Reader to return data to the Host on the command channel.

In short, ‘attributes’ or ‘properties’ (aka variables) with SET and/or GET operations, and ‘functions’ or ‘methods’ performing some action.

All commands are *atomic*, meaning that they SHALL be executed either completely or not at all. For example, when adding arrays of things (e.g., `FieldNames` to a `DataSelector`), and one value is not supported or not known, then no values SHALL be added and error SHALL be raised.

The return values of the command indicate the return values if the command was successfully executed. If the command cannot be executed correctly, the return value SHALL be undefined and an error condition SHALL be raised. The error conditions are explained in Section 8 Error Handling.

UML-like notation for an abstract command:

```
[static] OBJECT.COMMAND ([
                PARAMETER: DataType,
                PARAMETER: DataType,
                ...]): DataType
```

The `static` keyword indicates that a command doesn’t require an object instance in order to be executed.

DataTypes for command parameters and command return values:

- **address**
  See section 10.1.
  epc
  An EPC. The formatting depends on the MTB; it MAY be a string in urn-notation or also the raw binary value.
- **integer**
  All numeric values are 32-bit signed integers.
- **string**

  All strings SHALL be capable of being represented with UTF-8 encoding.
- **timestamp**

A timestamp (date and time) with millisecond accuracy. The exact format depends on the MTB in use.

- ***type*[]**

  A list of the specified data type. The list is unordered unless explicitly stated otherwise.

- **void**
  No data, indicating no parameters or return value (depending on where it's used).

In addition, many commands have object types as parameter or return value. The MTB defines how objects are encoded in the command and reply messages. The following section defines the Reader Management Command Set. A list of allowed errors is listed for each command. Section 8 Error Handling describes the Error conditions in detail.

5.1 ReaderDevice

5.1.1 ReaderDevice.getDescription

The Host queries the Reader for its user defined description. This is a text string that describes the general use of this particular reader.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.getDescription (void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string.` Reader will return its user defined description.

Possible Error Conditions:

ERROR_UNKNOWN

5.1.2 ReaderDevice.setDescription

The Host sets the Readers user-defined description. This is a text string that describes the general use of this particular reader.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.setDescription (description: string): void
```

Parameter(s):

`description` – Data Type: `string.` A text string that describes the general use of this particular reader.

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_PARAMETER_MISSING
ERROR_PARAMETER_INVALID_DATATYPE
ERROR_PARAMETER_LENGTH_EXCEEDED
ERROR_UNKNOWN
ERROR_AUTHORIZATION

5.1.3 ReaderDevice.getLocationDescription

The Host queries the Reader for its user defined location description. This is a textual description of the location of the reader device.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.getLocationDescription (void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string.` Reader will return its user defined location description.

Possible Error Conditions:

ERROR_UNKNOWN

5.1.4 ReaderDevice.setLocationDescription

The Host sets the Readers user-defined location description. This is a textual description of the location of the reader device.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.setLocationDescription (locationDescription:
string): void
```

Parameter(s):

`locationDescription` – Data Type: `string.` A textual description of the location of the reader device.

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_UNKNOWN

ERROR_AUTHORIZATION

5.1.5 ReaderDevice.getContact

The Host queries the Reader for its user defined contact description. This information identifies the individual or organization responsible for administering the Reader.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, the system implements setContact.

Usage:

```
ReaderDevice.getContact (void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string.` Reader will return a contact information to the person responsible for the device or someone acting for him/her.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

For a detailed error description see Reader Protocol chapter 8 Error Handling.

5.1.5.1 ReaderDevice.setContact

The Host sets the Readers user-defined contact description. This information identifies the individual or organization responsible for administering the Reader..

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.setContact (contact: string): void
```

Parameter(s):

`contact` – Data Type: `string.` A contact information to the person responsible for the device or someone acting for him/her.

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_UNKNOWN

ERROR_AUTHORIZATION

5.1.6 ReaderDevice.getSerialNumber

The Host queries the Reader for its serial number. This is the serial number of the reader device. Note that normally the serial number is also an integral part of the EPC. Since the serial number is set by the manufacturer, there is no `setSerialNumber()` command.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.getSerialNumber (void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string.` Reader will return the serial number of the reader device.

719

720 **Possible Error Conditions**:

721 ERROR_UNKNOWN

722 For a detailed error description see Reader Protocol chapter 8 Error Handling.

723

724

725 5.1.7 ReaderDevice.getOperStatus

726 The Host queries the Reader for the `ReaderDevice`'s `OperStatus` attribute, which
727 is of type `OperationalStatus`.

728

729 **Compliance Requirement:** Compliant systems SHALL implement this command.

730

731 **Usage**:

732 `ReaderDevice.getOperStatus (void): OperationalStatus`

733

734 **Parameter(s):**

735 Data Type: `void.` This command takes no parameters.

736

737 **Return Value(s):**

738 Data Type: `OperationalStatus.` Reader will return the ReaderDevice object's
739 OperStatus attribute.

740

741 **Possible Error Conditions**:

742

743 ERROR_UNKNOWN

744

745 5.1.8 ReaderDevice.getOperStatusAlarmControl

746 The Host queries the `ReaderDevice` for its `OperStatusAlarmControl` attribute,
747 of type TT`OperationalStatusAlarmControl`. This attribute is the object that
748 controls the conditions for generating alarms alerting a manager of changes in a
749 `ReaderDevice`'s operational status.

750

751 **Compliance Requirement:** Compliant systems SHALL implement this command if, and
752 only if, AlarmChannels are implemented.

Usage:

```
ReaderDevice.getOperStatusAlarmControl (void):
TTOperationalStatusAlarmControl
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: TT`OperationalStatusAlarmControl`. The ReaderDevice will return its OperStatusAlarmControl attribute.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.1.9 ReaderDevice.getFreeMemory

The Host queries the Reader for the amount of free memory available on the `ReaderDevice`. The `FreeMemory` attribute represents the number of kilobytes (KB) of Read/Write memory available to internal or external resources for buffering or other processing of information. For example, the `FreeMemory` attribute may include the number of kilobytes available for building a `ReportBuffer` (see RP1).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
ReaderDevice.getFreeMemory (void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. Reader will return its available free memory specified in kilo bytes (KB).

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

For a detailed error description see Reader Protocol chapter 8 Error Handling.

5.1.10 ReaderDevice.getFreeMemoryAlarmControl

The Host queries the Reader for the `ReaderDevice` object's `FreeMemoryAlarmControl` attribute, of type `EdgeTriggeredAlarmControl`. This object controls the generation of alarms alerting a managing system of a reduction of a reader's free memory resources below a specified threshold.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
ReaderDevice.getFreeMemoryAlarmControl (void):
EdgeTriggeredAlarmControl
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `EdgeTriggeredAlarmControl.` Reader will return its FreeMemoryAlarmControl object of type EdgeTriggeredAlarmControl.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.1.11 ReaderDevice.getNTPServers

The Host queries the Reader for a list of NTP servers used by the device to synchronize its current UTC clock (TimeUTC).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`ReaderDevice.getNTPServers (void): string[]`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string[].` Reader will return a list of NTP servers used by the device to synchronize its current UTC clock (TimeUTC).

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.1.12 ReaderDevice.getDHCPServer

The Host queries the Reader for the DHCP server currently used by the device for DHCP requests.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`ReaderDevice.getDHCPServer (void): string`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string.` Reader will return the DHCP server currently used by the device for DHCP requests.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.1.13 ReaderDevice.getIOPort

Returns the `IOPort` with the specified name currently associated with this Reader. If no `IOPort` object with the given name exists, the error ERROR_IOPORT_NOT_FOUND is raised.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReaderDevice.getIOPort (name: string): IOPort
```

Parameter(s):

`name` – Data Type: `String.` A name for the IOPort object which should be retrieved from the reader.

Return Value(s):

Data Type: `IOPort.` Reader will return the named IOPort object.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_IOPORT_NOT_FOUND

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_UNKNOWN

5.1.14 ReaderDevice.getAllIOPorts

The Host queries the Reader for all its IOPort objects.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

887 `ReaderDevice.getAllIOPorts (void): IOPort[]`

888

889 **Parameter(s):**

890 Data Type: `void.` This command takes no parameters.

891

892 **Return Value(s):**

893 Data Type: `IOPort[].` Reader will return all its IOPort objects.

894

895 **Possible Error Conditions**:

896 ERROR_COMMAND_NOT_SUPPORTED

897 ERROR_UNKNOWN

898

899 5.1.15 ReaderDevice.resetStatistics

900 Instructs the reader to reset all statistical counters.

901 Upon receiving this command, the reader SHALL set all supported counters to zero.
902 Counters which are affected (if the reader supports them) are:

Object	Counters to reset
AntennaReadPoint	IdentificationCount, FailedIdentificationCount, MemReadCount, FailedMemReadCount, WriteCount, FailedWriteCount, KillCount, FailedKillCount, EraseCount, FailedEraseCount, LockCount, FailedLockCount, TimeEnergized
Source	UnknownToGlimpsedCount, GlimpsedToUnknownCount, GlimpsedToObservedCount, ObservedToLostCount, LostToGlimpsedCount, LostToUnknownCount
Trigger	FireCount

903

904 **Compliance Requirement:** Compliant systems SHALL implement this command.

905

906 **Usage**:

907 `ReaderDevice.resetStatistics (void): void`

908

909 **Parameter(s):**

910 Data Type: `void.` This command takes no parameters.

911

Return Value(s):

Data Type: void. `This command will not return a value`.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

5.1.16 ReaderDevice.removeAlarmChannels

Removes the specified `AlarmChannels` from the list of `AlarmChannels` currently associated with this Reader. If one or more of the `AlarmChannels` given are not known, or if some of the `AlarmChannels` to be removed are currently not associated with this Reader, these are ignored and all other `AlarmChannels` SHALL be removed and the command SHALL complete successfully.

Compliance Requirement: Compliant systems SHALL implement this command, if and only if, AlarmChannel.create() is implemented.

Usage:

```
ReaderDevice.removeAlarmChannels (channels:
AlarmChannel[]): void
```

Parameter(s):

`channels[]` – Data Type: `AlarmChannel.` An array with AlarmChannel objects to be removed from the Reader.

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_UNKNOWN

5.1.17 ReaderDevice.removeAllAlarmChannels

Removes all `AlarmChannels` currently associated with this Reader.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, AlarmChannel is implemented.

Usage:

```
ReaderDevice.removeAllAlarmChannels (void): void
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.1.18 ReaderDevice.getAlarmChannel

Returns the `AlarmChannel` with the specified name currently associated with this Reader. If no `AlarmChannel` object with the given name exists, the error ERROR_ALARM_CHANNEL_NOT_FOUND is raised.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, AlarmChannel.create() is implemented.

Usage:

```
ReaderDevice.getAlarmChannel (name: string): AlarmChannel
```

Parameter(s):

`name` – Data Type: `String.` A name for the AlarmChannel object which should be retrieved from the reader.

Return Value(s):

Data Type: `AlarmChannel.` Reader will return the named AlarmChannel object.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_ALARM_CHANNEL_NOT_FOUND

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_UNKNOWN

5.1.19 ReaderDevice.getAllAlarmChannels

Returns all `AlarmChannels` currently associated with the Reader. If no `AlarmChannel` are currently associated with this object, the command SHALL complete successfully and an empty list SHALL be returned.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, AlarmChannel.create() is implemented.

Usage:

```
ReaderDevice.getAllAlarmChannels (void): AlarmChannel[]
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `AlarmChannel[].` Reader will return an array with all defined AlarmChannel objects for this Reader.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.2 NotificationChannel

5.2.1 NotificationChannel.getLastNotificationAttempt

The Host queries the NotificationChannel object for the timestamp (TimeTicks) when the last attempt was made to send a notification to the given address.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
NotificationChannel.getLastNotificationAttempt (void):
timestamp
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `timestamp.` NotificationChannel object will return the timestamp (TimeTicks) when the last attempt was made to send a notification to the given address.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

For a detailed error description see Reader Protocol chapter 8 Error Handling.

5.2.2 NotificationChannel.getLastSuccessfulNotification

The Host queries the NotificationChannel object for the timestamp (TimeTicks) when the last successful notification was send to the given address.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`NotificationChannel.getLastSuccessfullNotification (void): timestamp`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `timestamp.` NotificationChannel object will return the timestamp (TimeTicks) when the last successful notification was send to the given address.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.2.3 NotificationChannel.getOperStatus

The Host queries the NotificationChannel for its OperationalStatus attribute.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, the system implements Notification Channels.

Usage:

`NotificationChannel.getOperStatus (void): OperationalStatus`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus.` NotificationChannel will return its OperationalStatus attribute.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.2.4 NotificationChannel.setAdminStatus

The Host sets the NotificationChannels AdminStatus object of type AdministrativeStatus. The administrative state identifies if the NotificationChannel has been configured to be operational. For example: UP or DOWN.

Compliance Requirement: Compliant systems SHALL implement this command if the system implements Notification Channels.

Usage:

```
NotificationChannel.setAdminStatus (administrativeStatus:
AdministrativeStatus): void
```

Parameter(s):

`administrativeStatus` – Data Type: `AdministrativeStatus.` The administrative state for the NotificationChannel.

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_UNKNOWN

5.2.5 NotificationChannel.getAdminStatus

The Host queries the NotificationChannel for its AdminStatus object of type AdministrativeStatus.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, the system implements Notification Channels.

Usage:

```
NotificationChannel.getAdminStatus (void):
AdministrativeStatus
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `AdministrativeStatus.` NotificationChannel will return its AdminStatus object of type AdministrativeStatus.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.2.6 NotificationChannel.getOperStatusAlarmControl

The Host queries the `NotificationChannel` object for its `OperStatusAlarmControl` attribute of type `TTOperationalStatusAlarmControl`.. This attribute is the object that controls the conditions for generating alarms alerting a manager of changes in a `NotificationChannel`'s operational status.

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, the system implements Notification Channels and AlarmChannels.

Usage:

```
NotificationChannel.getOperStatusAlarmControl (void):
TTOperationalStatusAlarmControl
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: TT`OperationalStatusAlarmControl.` The NotificationChannel will return its OperStatusAlarmControl object of type TTOperationalStatusAlarmControl.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.3 AlarmChannel

NOTE: AlarmChannel objects are to be treated as RP Notification Channels, which means that they use the same handshake and connection mechanisms. See [RP1] for details.

5.3.1 AlarmChannel.create

Create an `AlarmChannel` object with a given name. If an `AlarmChannel` object with the same name exists already, an error is returned. This is a static method.

The `AlarmChannel` SHALL implicitly be added to the list of all `AlarmChannels` kept by the `ReaderDevice` object.

Compliance Requirement: Compliant systems MAY implement this command. If it is not implemented, and Alarm Notifications are implemented, the system SHALL provide an alternate method of defining AlarmChannels.

Usage:

```
static AlarmChannel.create(
            name: string,
            addr: address): AlarmChannel
```

Parameter(s):

name - Data Type: `string.` The name of the `AlarmChannel` to be created.

`addr` - Data Type: `address.` The (host) address to which the reader will send alarms.

Return Value(s):

Data Type: `AlarmChannel.` The newly created object.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_OBJECT_EXISTS

ERROR_UNKNOWN

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_INVALID_FORMAT

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_AUTHORIZATION

ERROR_TOO_MANY_ALARM_CHANNELS

5.3.2 AlarmChannel.getName

Returns the name of the given `AlarmChannel` object.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannels are implemented.

Usage:

```
AlarmChannel.getName(void): string
```

Parameter(s):

Data Type: `void`. This command SHALL NOT take any parameters.

Return Value(s):

Data Type: `string`. The name of the `AlarmChannel` object is returned.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_CHANNEL_NOT_FOUND

ERROR_UNKNOWN

5.3.3 AlarmChannel.getAddress

Returns the (host) address to which this `AlarmChannel` object sends its alarms.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannels are implemented.

Usage:

1215 `AlarmChannel.getAddress(void): address`

1216

1217 **Parameter(s):**

1218 Data Type: `void.` This command SHALL NOT take any parameters.

1219

1220 **Return Value(s):**

1221 Data Type: `address.` The reporting address for this `AlarmChannel`.

1222

1223 **Possible Error Conditions**:

1224 ERROR_CHANNEL_NOT_FOUND

1225 ERROR_COMMAND_NOT_SUPPORTED

1226 ERROR_UNKNOWN

1227

1228 5.3.4 AlarmChannel.setAddress

1229 Sets the (host) address to which this `AlarmChannel` object sends its alarms.

1230

1231 **Compliance Requirement**: Compliant systems SHALL implement this command if
1232 AlarmChannel.create is implemented. Compliant systems MAY implement this
1233 command if AlarmChannel is implemented by means of an alternate method without
1234 implementing NotificationChannel.create..

1235

1236 **Usage**:

1237 `AlarmChannel.setAddress(addr: address) : void`

1238

1239 **Parameter(s):**

1240 `addr` - Data Type: `address.` The reporting address for this `AlarmChannel`.

1241

1242 **Return Value(s):**

1243 Data Type: `void. This command SHALL NOT return any value.`

1244

1245 **Possible Error Conditions**:

1246 ERROR_COMMAND_NOT_SUPPORTED

1247 ERROR_CHANNEL_NOT_FOUND

ERROR_UNKNOWN

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_INVALID_FORMAT

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_AUTHORIZATION

5.4 ReadPoint

5.4.1 ReadPoint.getClassName

This method returns the class name of the ReadPoint object. In the current specification, the only supported read point class is an "AntennaReadPoint", but other types may be introduced in future versions.

Compliance Requirement: Compliant systems SHALL implement this command.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReadPoint.getClassName(void): string
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: string. The class name for the ReadPoint. The only supported return value is "AntennaReadPoint". Note that future versions of the specification may support other return values.

Possible Error Conditions:

ERROR_UNKNOWN

5.4.2 ReadPoint.getDescription

Returns a description of this ReadPoint.

The description is set via a call to setDescription and returned via this call. In the event there has been no previous call to setDescription then the returned string will have 0 length.

Compliance Requirement: Compliant systems SHALL implement this command .

Usage:

```
ReadPoint.getDescription (void): string
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `string.` The readpoint descriptive name.

Possible Error Conditions:

ERROR_UNKNOWN

5.4.3 ReadPoint.setDescription

Set the current ReadPoint description.

This may later be retrieved using ReadPoint.getDescription.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReadPoint.setDescription(description: string): void
```

Parameter(s):

`description` – Data Type: `String`. A description for this `ReadPoint`. The maximum allowable length for this string is vendor-dependent. Attempts to set a string of excessive length will result in a "length exceeded" error.

Return Value(s):

Data Type: `void`. This command will not return a value.

Possible Error Conditions:

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_INVALID_VALUE

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_UNKNOWN

ERROR_AUTHORIZATION

5.4.4 ReadPoint.getAdminStatus

Returns the current ReadPoint administrative status.

The administrative status represents the host's *desired* status for this ReadPoint. This differs from the operational status which represents the *actual* ReadPoint status. The administrative status is set via a call to setAdminStatus and getAdminStatus simply returns the most-recently-set value, or the initial vendor-specific administrative state if setAdminStatus has not yet been called.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReadPoint.getAdminStatus(void): AdministrativeStatus
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `AdministrativeStatus`.

Possible Error Conditions:

ERROR_UNKNOWN

For a detailed description of these error codes see the Reader Protocol Specification 1.1.

5.4.5 ReadPoint.setAdminStatus

Set the current ReadPoint administrative status.

The administrative status represents the host's *desired* status for this ReadPoint. This differs from the operational status which represents the *actual* ReadPoint status. Hosts may query the most-recently-requested status using getAdminStatus.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReadPoint.setAdminStatus(status: AdministrativeStatus):
void
```

Parameter(s):

`status – Data Type: AdministrativeStatus. The desired administrative status of the ReadPoint.`

Return Value(s):

Data Type: `void.` This command will not return a value.

Possible Error Conditions:

ERROR_PARAMETER_MISSING

ERROR_PARAMETER_INVALID_DATATYPE

ERROR_PARAMETER_INVALID_VALUE

ERROR_UNKNOWN

5.4.6 ReadPoint.getOperStatus

Returns the ReadPoint's current operational status.

The operational status represents the actual status of the ReadPoint. It may be polled using getOperStatus or monitored via alarms (see GetOperStatusAlarmControl).

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
ReadPoint.getOperStatus(void): OperationalStatus
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `OperationalStatus. The operational status of the ReadPoint. It may be “unknown”, “up”, “down” or “other”. See the definition of the OperationalStatus enumeration for details.`

Possible Error Conditions:

ERROR_UNKNOWN

5.4.7 ReadPoint.getOperStatusAlarmControl

The Host queries the `ReadPoint` object for its `OperStatusAlarmControl` attribute of type `TTOperationalStatusAlarmControl`.. This attribute is the object that controls the conditions for generating alarms alerting a manager of changes in a `ReadPoint`’s operational status.

In addition to an alarm, the operational status may also be polled (see getOperStatus).

Compliance Requirement: Compliant systems SHALL implement this command if, and only if, AlarmChannels are implemented.

Usage:

`ReadPoint.getOperStatusAlarm`Control `(void): TTOperationalStatusAlarmControl`

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: TT`OperationalStatusAlarmControl. An alarm control for monitoring the operational status of the ReadPoint.`

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5 AntennaReadPoint

`AntennaReadPoint` extends `ReadPoint`. It is used to track details about the Radio Frequency characteristics and RFID statistics for the reader device.

5.5.1 AntennaReadPoint.getIdentificationCount

Returns the number of the successful tags that have been identified across an `AntennaReadPoint`. In other words, the number of tags whose unique identifier has been successfully detected by the AntennaReadPoint.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Note that if a single tag's identifier is read multiple times, then the read count will increment on every successful read. It is the number of times *any* tag's identifier is read, not the number of *unique* tags which have been read.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
AntennaReadPoint.getIdentificationCount (void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer.` The count of the successful tag identifiers read at this `AntennaReadPoint`.

Possible Error Conditions:

ERROR_UNKNOWN

5.5.2 AntennaReadPoint.getFailedIdentificationCount

Returns the number of the failed tag identification attempts at the `AntennaReadPoint`.

This count only includes failures where the reader can unambiguously determine that a tag was in field that could not be identified. If the reader does not have the means to determine this, this counter may be left at 0.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedIdentificationCount (void):
integer
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `integer.` The count of the failed attempts at reading the identifier for a tag at this antenna ReadPoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.3 AntennaReadPoint.getMemReadCount

Returns the number of tag memory reads at the `AntennaReadPoint`. This should not be confused with the number of times the tag has been successfully queried or has communicated its unique identifier. Such statistics is reported by AntennaReadPoint.getIdentificationCount.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getMemReadCount (void): integer
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `integer.` The count of the successful tag memory reads at this antenna ReadPoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.4 AntennaReadPoint.getFailedMemReadCount

Returns the number of the failed tag memory reads at the `AntennaReadPoint`.

This count only includes failures where it attempts to read tag memory but does not complete successfully.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`AntennaReadPoint.getFailedMemReadCount(void): integer`

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `integer.` The count of the failed tag memory reads at this antenna readpoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.5 AntennaReadPoint.getFailedMemReadAlarmControl

The Host queries the Antenna`ReadPoint` object for its `FailedMemReadAlarmControl` attribute of type `AlarmControl`. This alarm control object controls the generation of alarms triggered by failed read operations of memory banks across the AntennaRead point.

In addition to an alarm, the status of failed memory reads may also be polled (see getMemReadCount).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedMemReadAlarmControl
(void):AlarmControl
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `AlarmControl.` An alarm control for monitoring tag memory read failures.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.6 AntennaReadPoint.getWriteCount

Returns the number of successful tag writes at the `AntennaReadPoint`, including writes to any of the memory banks including the tag identifier.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to ReaderDevice.resetStatistics.

Note that if a single tag is written multiple times, then the count will increment on every successful write. It is the number of times *any* tag has been written, not the number of *unique* tags which have been written.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
AntennaReadPoint.getWriteCount(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer`. The count of the successful writes at this `AntennaReadPoint`. If Write functionality is not supported by the Reader it must return 0 every time.

Possible Error Conditions:

ERROR_UNKNOWN

For a detailed description of these error codes see the Reader Protocol Specification 1.1.

5.5.7 AntennaReadPoint.getFailedWriteCount

Returns the number of the failed attempts to write tags at the `AntennaReadPoint`, including failed writes to any of the memory banks including the tag identifier.

This is the number of times the reader attempted to write a tag, but the operation failed to complete successfully.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedWriteCount(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer`. The count of the failed writes at this `AntennaReadPoint`.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.8 AntennaReadPoint.getFailedWriteAlarmControl

The Host queries the Antenna`ReadPoint` object for its `FailedWriteAlarmControl` attribute of type `AlarmControl`. This alarm control object controls the generation of alarms triggered by failed write operations across the AntennaRead point, including failed writes to any of the memory banks including the tag identifier.

In addition to an alarm, the status of writes may also be polled (see getWriteCount and getFailedWriteCount).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedWriteAlarmControl
(void):AlarmControl
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `AlarmControl`. An alarm control for monitoring the number of failed writes.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

For a detailed description of these error codes see the Reader Protocol Specification 1.1.

5.5.9 AntennaReadPoint.getKillCount

Returns the number of tags successfully killed at the `AntennaReadPoint`.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
AntennaReadPoint.getKillCount(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer.` The count of the successful tag kills at this antenna ReadPoint.

Possible Error Conditions:

ERROR_UNKNOWN

5.5.10 AntennaReadPoint.getFailedKillCount

Returns the number of the failed tag kills at the `AntennaReadPoint`.

This count only includes failures where it attempts to kill a tag but the kill does not complete successfully.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedKillCount(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer.` The count of the failed tag kills at this antenna readpoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.11 AntennaReadPoint.getFailedKillAlarmControl

Return the `AntennaReadPoint`'s failed kill alarm control.

In addition to an alarm, the status of kills may also be polled (see getKillCount and getFailedKillCount).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedKillAlarmControl
(void):AlarmControl
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `AlarmControl.` An alarm control for monitoring tag kill failures.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.12 AntennaReadPoint.getEraseCount

Returns the number of tags successfully erased at the `AntennaReadPoint`.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
AntennaReadPoint.getEraseCount (void): integer
```

Parameter(s):

Data Type: void. This command takes no parameters.

Return Value(s):

Data Type: integer. The count of the successful tag erasures at this antenna ReadPoint.

Possible Error Conditions:

ERROR_UNKNOWN

5.5.13 AntennaReadPoint.getFailedEraseCount

Returns the number of the failed tag erasures at the AntennaReadPoint.

This count only includes failures where it attempts to erase a tag but the erasure does not complete successfully.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to ReaderDevice.resetStatistics.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedEraseCount(void): integer
```

Parameter(s):

Data Type: void. This command takes no parameters.

Return Value(s):

Data Type: integer. The count of the failed tag erasures at this antenna readpoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.14 AntennaReadPoint.getFailedEraseAlarmControl

Return the AntennaReadPoint's failed erase alarm control.

In addition to an alarm, the status of erasures may also be polled (see getEraseCount and getFailedEraseCount).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedEraseAlarmControl
(void):AlarmControl
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `AlarmControl.` An alarm control for monitoring tag erasure failures.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.15 AntennaReadPoint.getLockCount

Returns the number of tags successfully locked at the `AntennaReadPoint`.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getLockCount (void): integer
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `integer.` The count of tags successfully locked at this antenna ReadPoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.16 AntennaReadPoint.getFailedLockCount

Returns the number of the failed tag locks at the `AntennaReadPoint`.

This count only includes failures where it attempts to lock a tag but the lock does not complete successfully.

This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedLockCount(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer.` The count of the failed tag locks at this antenna readpoint.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

AntennaReadPoint.getFailedLockAlarmControl

5.5.16.1

Return the `AntennaReadPoint`'s failed lock alarm control.

In addition to an alarm, the status of locks may also be polled (see getLockCount and getFailedLockCount).

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getFailedLockAlarmControl
(void):AlarmControl
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `AlarmControl.` An alarm control for monitoring tag lock failures.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.17 AntennaReadPoint.getTimeEnergized

Returns the number of milliseconds the `AntennaReadPoint has been energized in order to communicate with tags`.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getTimeEnergized(void): integer
```

Parameter(s):

`Data Type: void. This command takes no parameters.`

Return Value(s):

Data Type: `integer.` The number of milliseconds the `AntennaReadPoint has been energized attempting communication with tags`. This count is automatically reset to 0 whenever the reader is restarted, and may be manually reset via a call to `ReaderDevice.resetStatistics`.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

AntennaReadPoint.getPowerLevel

5.5.17.1

Returns the current transmit power level of the `AntennaReadPoint`. The units of measurement are vendor specific.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getPowerLevel(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer.` The current transmit power level of the `AntennaReadPoint`. The meaning of this value is device-dependent and users should consult documentation provided by the reader manufacturer.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.5.18 AntennaReadPoint.getNoiseLevel

The Host sends a `getNoiseLevel` message to query the Reader for the current noise level observed at the `AntennaReadPoint`. The units of measurement are vendor specific.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
AntennaReadPoint.getNoiseLevel(void): integer
```

Parameter(s):

```
Data Type: void. This command takes no parameters.
```

Return Value(s):

Data Type: `integer`. The current noise level observed at the `AntennaReadPoint`. The meaning of this value is device-dependent and users should consult the documentation provided by the reader manufacturer for the meaning. The only constraint is that larger numbers imply higher levels of noise.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_UNKNOWN

5.6 Source Object

5.6.1 Source.getUnknownToGlimpsedCount

The Host queries the Reader for the number of times a transition from state Unknown to state Glimpsed have been detected for the particular source.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
Source.getUnknownToGlimpsedCount(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. The number times the particular transition has been detected.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.2 Source.getGlimpsedToUnknownCount

The Host queries the Reader for the number of times a transition from state Glimpsed to state Unknown have been detected for the particular source.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`Source.getGlimpsedToUnknownCount(void): integer`

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. The number times the particular transition has been detected.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.3 Source.getGlimpsedToObservedCount

The Host queries the Reader for the number of times a transition from state Glimpsed to state Observed have been detected for the particular source.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`Source.getGlimpsedToObservedCount(void): integer`

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. The number times the particular transition has been detected.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.4 Source.getObservedToLostCount

The Host queries the Reader for the number of times a transition from state Observed to state Lost have been detected for the particular source.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
Source.getObservedToLostCount(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. The number times the particular transition has been detected.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.5 Source.getLostToGlimpsedCount

The Host queries the Reader for the number of times a transition from state Lost to state Observed have been detected for the particular source.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
Source.getLostToGlimpsedCount(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. The number times the particular transition has been detected.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.6 Source.getLostToUnknownCount

The Host queries the Reader for the number of times a transition from state Lost to state Unknown have been detected for the particular source.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
Source.getLostToUnknownCount(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`. The number times the particular transition has been detected.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.7 Source.getOperStatus

The Host queries the Reader for the operational status of this particular Source. The operational status reported is one of the values in the OperationalStatus enumeration, e.g. UP, DOWN, UNKNOWN or OTHER. The operational status represents the actual status of the Source. It may be polled using getOperStatus or monitored via alarms (see GetOperStatusAlarmControl).

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

`Source.getOperStatus(void): OperationalStatus`

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus`. Reader will return the current operational status of this Source.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_SOURCE_NOT_FOUND

5.6.8 Source.getAdminStatus

The Host queries the Reader for the administrative status of a particular Source. The administrative status reported is one of the values in the AdministrativeStatus enumeration, e.g. UP or DOWN.

The administrative status represents the host's *desired* status for this Source. This differs from the operational status which represents the *actual* Source status. The administrative status is set via a call to setAdminStaus and getAdminStatus simply returns the most-recently-set value.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

`Source.getAdminStatus(void): AdministrativeStatus`

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `AdministrativeStatus`. Reader will return the current administrative status of this Source.

Possible Error Conditions:

ERROR_SOURCE_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.6.9 Source.setAdminStatus

This operation is used to set the administrative status of a particular Source. The administrative status assigned is one of the values in the AdministrativeStatus enumeration, e.g. UP or DOWN

The administrative status represents the host's *desired* status for this Source. This differs from the operational status which represents the *actual* status. Hosts may query the most-recently-requested status using getAdminStatus.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
Source.setAdminStatus(administrativeStatus
:AdministrativeStatus): void
```

Parameter(s):

`administrativeStatus` – Data Type: `AdministrativeStatus`. This is the new administrative status to be set.

Return Value(s):

Data Type: `void`. This command will not return a value.

Possible Error Conditions:

ERROR_COMMAND_NOT_SUPPORTED

ERROR_INVALID_ADMIN_STATUS

5.6.10 Source.getOperStatusAlarmControl

The Host queries the `Source` object for its `OperStatusAlarmControl` attribute of type `TTOperationalStatusAlarmControl`. This attribute is the object that controls the conditions for generating alarms alerting a manager of changes in a `Source`'s operational status.

In addition to an alarm, the operational status may also be polled (see getOperStatus).

Compliance Requirement: Compliant systems SHALL implement this command if `AlarmChannel` is implemented.

Usage:

```
Source.getOperStatusAlarm (void):
TTOperationalStatusAlarmControl
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: TT`OperationalStatusAlarmControl.` An alarm control for monitoring the operational status of the Source.

Possible Error Conditions:

ERROR_COMMAND_NOT_FOUND

ERROR_SOURCE_NOT_FOUND

5.7 Trigger Object

5.7.1 Trigger.getFireCount

The Host queries the Reader for the number of times a particular trigger has fired.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

```
Trigger.getFireCount (void): integer
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer`. Reader will return the number of times the trigger has fired.

Possible Error Conditions:

ERROR_TRIGGER_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.8 IOPort Object

5.8.1 IOPort.getName

The Host queries the Reader for the name of a particular IO-port. Since the IO-port names are set by the manufacturer, there is no `setName()` command.

Compliance Requirement: Compliant systems SHALL implement this command if IOPorts are exposed.

Usage:

```
IOPort.getName(void): string
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string`. Reader will return the name of this IO-port.

Possible Error Conditions:

ERROR_IOPORT_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.8.2 IOPort.getDescription

This operation provides a textual description of the IO-port. This is typically used to denote the equipment connected via this IO-port.

Compliance Requirement: Compliant systems SHALL implement this command if IOPorts are exposed.

Usage:

`IOPort.getDescription(void): string`

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string`. Reader will return the description of the IO-port in a free form String. If no description has been set the operation returns a String of length zero (an empty string).

Possible Error Conditions:

ERROR_IOPORT_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.8.3 IOPort.setDescription

This operation is used to associate a textual description with an IO-port. This is typically used to denote the equipment connected via this IO-port.

Compliance Requirement: Compliant systems SHALL implement this command if IOPorts are exposed.

Usage:

`IOPort.setDescription(description : string): void`

Parameter(s):

`description` – Data Type: `string`. The textual description.

Return Value(s):

Data Type: `void`. This operation does not provide a return value.

Possible Error Conditions:

ERROR_IOPORT_NOT_FOUND

ERROR_PARAMETER_LENGTH_EXCEEDED

ERROR_COMMAND_NOT_SUPPORTED

5.8.4 IOPort.getOperStatus

The Host queries the Reader for the operational status of this particular IO-port. The operational status reported is one of the values in the OperationalStatus enumeration, e.g. UP, DOWN, UNKNOWN or OTHER. The operational status represents the actual status of the IOPort. It may be polled using getOperStatus or monitored via alarms (see getOperStatusAlarmControl).

Compliance Requirement: Compliant systems SHALL implement this command if IOPorts are exposed.

Usage:

```
IOPort.getOperStatus(void): OperationalStatus
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus`. Reader will return the current operational status of this IO-port.

Possible Error Conditions:

ERROR_IOPORT_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.8.5 IOPort.getAdminStatus

The Host queries the Reader for the administrative status of a particular IO-port. The administrative status reported is one of the values in the AdministrativeStatus enumeration, e.g. UP or DOWN.

The administrative status represents the host's *desired* status for this IOPort. This differs from the operational status which represents the *actual* IOPort status. The administrative status is set via a call to setAdminStaus and getAdminStatus simply returns the most-recently-set value.

Compliance Requirement: Compliant systems SHALL implement this command if IOPorts.

Usage:

```
IOPort.getAdminStatus(void): AdministrativeStatus
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `AdministrativeStatus.` Reader will return the current administrative status of this IO-port.

Possible Error Conditions:

ERROR_IOPORT_NOT_FOUND

ERROR_COMMAND_NOT_SUPPORTED

5.8.6 IOPort.setAdminStatus

This operation is used to set the administrative status of a particular IO-port. The administrative status assigned is one of the values in the AdministrativeStatus enumeration, e.g. UP or DOWN

The administrative status represents the host's *desired* status for this IOPort. This differs from the operational status which represents the *actual* IOPort status. Hosts may query the most-recently-requested status using getAdminStatus.

Compliance Requirement: Compliant systems SHALL implement this command if IOPorts are exposed.

Usage:

```
IOPort.setAdminStatus(administrativeStatus :
AdministrativeStatus): void
```

Parameter(s):

`administrativeStatus` – Data Type: `AdministrativeStatus`.. This is the new administrative status to be set.

Return Value(s):

2257 Data Type: `void.` This operation does not provide a return value.

2258

2259 **Possible Error Conditions**:

2260 ERROR_COMMAND_NOT_SUPPORTED

2261 ERROR_IOPORT_NOT_FOUND

2262 ERROR_INVALID_ADMIN_STATUS

2263

2264 5.8.7 IOPort.getOperStatusAlarmControl

2265 The Host queries the `IOPort` object for its `OperStatusAlarmControl` attribute of
2266 type `TTOperationalStatusAlarmControl`. This attribute is the object that
2267 controls the conditions for generating alarms alerting a manager of changes in an
2268 `IOPort`'s operational status.

2269 In addition to an alarm, the operational status may also be polled (see getOperStatus).

2270

2271 **Compliance Requirement**: Compliant systems SHALL implement this command if
2272 `AlarmChannel` is implemented..

2273

2274 **Usage**:

2275 `IOPort.getOperStatusAlarmControl (void):`
2276 `TTOperationalStatusAlarmControl`

2277

2278 **Parameter(s):**

2279 Data Type: `void.` This command takes no parameters.

2280

2281 **Return Value(s):**

2282 Data Type: TT`OperationalStatusAlarmControl.` An alarm control for
2283 monitoring the operational status of the IOPort.

2284

2285 **Possible Error Conditions**:

2286 ERROR_COMMAND_NOT_SUPPORTED

2287 ERROR_IOPORT_NOT_FOUND

2288

2289 AlarmControl Object`AlarmControl` is the base class for all classes responsible for
2290 controlling the generation of `Alarm` messages, and consists of four main attributes.

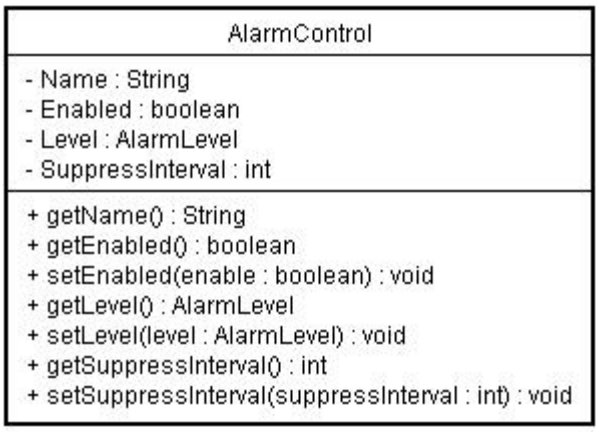


Figure 15 AlarmControl UML

The `Name` attribute (of type `String`) is a unique identifier of an `AlarmControl` object.

The `Enabled` attribute (of type `Boolean`) controls whether the `Alarm` is enabled. If set to false, `Alarm` generation will be inhibited for the specific `Alarm` this object controls.

The Level attribute (of type AlarmLevel) specifies the `Alarm` level (defined by the `AlarmLevel` enumeration) assigned to alarms whose generation this object controls.

The `SuppressInterval` attribute (of type `integer`) specifies the minimum number of seconds that SHALL elapse before the next `Alarm` of the same type is generated. Each object can only generate one of these `Alarm`s per `SupressInterval`. Setting the SuppressInterval to 0 results in an `Alarm` generated for every `Alarm` condition encountered. Setting `SuppressInterval` to a value greater than zero reduces the number of `Alarm`s raised, minimizing the network traffic and load on the health monitoring applications. The ideal value will depend on the monitored environment and how frequently the health monitoring application needs to be interrupted to process `Alarm` conditions. Every `Alarm` contains a `SuppressCount` indicating the number of times the `Alarm` generation has been suppressed during SuppressInterval. This count is reset to 0 after the successful generation of the `Alarm`.

For example, consider two `AntennaReadPoint` instances named AntennaReadPoint1 and AntennaReadPoint2. The `FailedKillAlarmControl` and `FailedWriteAlarmControl` have both been enabled with `SupressInterval` set to 15 seconds. The following table lists the action taken when a number of `Alarm` conditions are encountered at different times.

Time	Object	Alarm Condition	Action
09:00:00	AntennaReadPoint1	Kill Failure	Generate Alarm
09:00:01	AntennaReadPoint2	Kill Failure	Generate Alarm
09:00:05	AntennaReadPoint1	Kill Failure	Suppress Alarm
09:00:06	AntennaReadPoint1	Write Failure	Generate Alarm
09:00:08	AntennaReadPoint1	Kill Failure	Suppress Alarm
09:00:16	AntennaReadPoint1	Kill Failure	Generate Alarm

Defined subclasses of `AlarmControl` are:

- `EdgeTriggeredAlarmControl`
-
- `TTOperationalStatusAlarmControl`

Future revisions of this specification may define new `AlarmControl`s in addition to `EdgeTriggeredAlarmControl` and `TTOperationStatusAlarmControl`

5.8.8 AlarmControl.getName

Queries the reader for the current value of the `AlarmControl`'s `Name` attribute.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.getName(void): string
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string`. The `AlarmControl`'s unique name.

Possible Error Conditions:

None.

5.8.9 AlarmControl.getEnabled

Queries the reader for the current value of the `AlarmControl`'s `Enabled` attribute.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.getEnabled(void): boolean
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `boolean.` Indicates whether the `AlarmControl` is enabled.

Possible Error Conditions:

None.

5.8.10 AlarmControl.setEnabled

Enables or disables alarm generation.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.setEnabled(enable: boolean): void
```

Parameter(s):

enable – Data Type: `boolean. A boolean that enables or disables the AlarmControl.`

Return Value(s):

Data Type: `void.`

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.8.11 AlarmControl.getLevel

Queries the reader for the current value of the `AlarmControl`'s `Level` attribute.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.getLevel(void): AlarmLevel
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `AlarmLevel.`

Possible Error Conditions:

None.

5.8.12 AlarmControl.setLevel

Sets the `Level` attribute.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.setLevel(alarmLevel: AlarmLevel): void
```

Parameter(s):

alarmLevel – Data Type: `AlarmLevel. The desired Alarm Level value.`

Return Value(s):

Data Type: `void.`

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.8.13 AlarmControl.getSuppressInterval

Queries the reader for the current value of the `AlarmControl`'s `SuppressInterval` attribute.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.getSuppressInterval(void): integer
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The value of the AlarmControl SuppressInterval value in msec.`

Possible Error Conditions:

None.

5.8.14 AlarmControl.setSuppressInterval

Sets the `SuppressInterval` attribute.

Compliance Requirement: Compliant systems SHALL implement this command if AlarmChannel is implemented.

Usage:

```
AlarmControl.setSuppressInterval(suppressInterval:
integer): void
```

Parameter(s):

suppressInterval – Data Type: `integer. The desired Alarm SuppressInterval value in msec.`

Return Value(s):

Data Type: `void: This command will not return a value.`

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.9 EdgeTriggeredAlarmControl

This class extends `AlarmControl` to control alarms generated when a monitored, integer-valued, model element first crosses a threshold value (the `AlarmThreshold`). This type of alarm is called "edge triggered". The monitored value must cross a potentially different threshold (the `RearmThreshold`) before it can be triggered again.

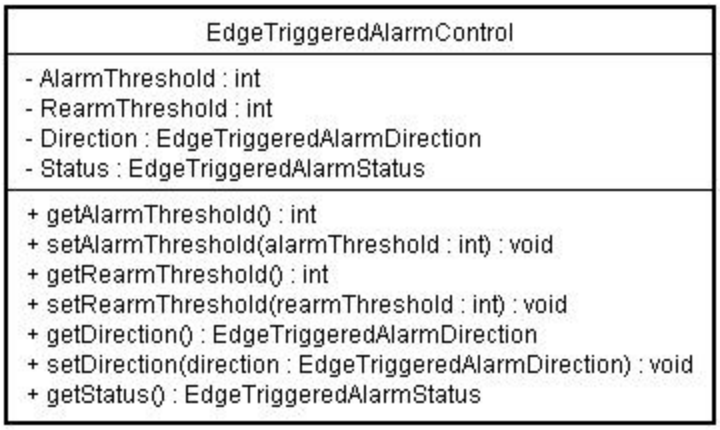


Figure 16 EdgeTriggeredAlarmControl UML

The `Direction` attribute (of enumerated type `EdgeTriggeredAlarmDirection`) indicates whether the alarm will be triggered on an `EdgeTriggeredAlarmDirection.RISING` or `EdgeTriggeredAlarmDirection.FALLING` passage through the `AlarmThreshold`. The `Status` attribute (of enumerated type `EdgeTriggeredAlarmStatus`) identifies the current state of the alarm, i.e., `EdgeTriggeredAlarmStatus.ARMED` or `EdgeTriggeredAlarmStatus.FIRED`.

If alarm control's direction is RISING, an alarm is triggered when the monitored value first becomes greater than the `AlarmThreshold`; once triggered, the alarm enters the SUSPENDED state, remaining in that state until the monitored value moves below (less than) the `RearmThreshold`, at which time the alarm re-enters the ARMED state.

If the alarm control's direction is FALLING, an alarm is triggered when the monitored value first becomes less than the `AlarmThreshold`; once triggered, the alarm enters the SUSPENDED state, remaining in that state until the monitored value moves above (greater than) the `RearmThreshold`, at which time the alarm re-enters the ARMED state.

5.9.1 EdgeTriggeredAlarmControl.getAlarmThreshold

Queries the reader for the current value of the `EdgeTriggeredAlarmControl`'s `AlarmThreshold` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.getAlarmThreshold(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`.

Possible Error Conditions:

None.

5.9.2 EdgeTriggeredAlarmControl.setAlarmThreshold

Command sent to the reader to set the current value of the `EdgeTriggeredAlarmControl`'s `AlarmThreshold` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.setAlarmThreshold(alarmThreshold:
integer): void
```

Parameter(s):

alarmThreshold – Data Type: `integer`. The desired value of `AlarmThreshold`.

Return Value(s):

Data Type: `void`.

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.9.3 EdgeTriggeredAlarmControl.getRearmThreshold

Queries the reader for the current value of the `EdgeTriggeredAlarmControl`'s `RearmThreshold` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.getRearmThreshold(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer`.

Possible Error Conditions:

None.

5.9.4 EdgeTriggeredAlarmControl.setRearmThreshold

Command sent to the reader to set the current value of the EdgeTriggered`AlarmControl`'s `RearmThreshold` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.setRearmThreshold(alarmThreshold:
integer): void
```

Parameter(s):

alarmThreshold – Data Type: `integer`. The desired value of `RearmThreshold`.

Return Value(s):

Data Type: `void`.

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.9.5 EdgeTriggeredAlarmControl.getDirection

Queries the reader for the current value of the `EdgeTriggeredAlarmControl`'s `Direction` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.getDirection (void):
EdgeTriggeredAlarmDirection
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `EdgeTriggeredAlarmDirection.` The return data type is an enumerated type with possible values `RISING` and `FALLING`.

Possible Error Conditions:

None.

5.9.6 EdgeTriggeredAlarmControl.setDirection

Command sent to the reader to set the current value of the `EdgeTriggeredAlarmControl`'s `Direction` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.setDirection(direction:
EdgeTriggeredAlarmDirection): void
```

Parameter(s):

direction: – Data Type: `EdgeTriggeredAlarmDirection.` The desired value of `Direction`.

Return Value(s):

Data Type: `void.`

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.9.7 EdgeTriggeredAlarmControl.getStatus

Queries the reader for the current value of the `EdgeTriggeredAlarmControl`'s `Status` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
EdgeTriggeredAlarmControl.getStatus(void):
EdgeTriggeredAlarmStatus
```

Parameter(s):

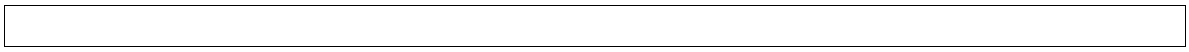
Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `EdgeTriggeredAlarmStatus. The return data type is an enumerated type with possible values ARMED OR FIRED.`

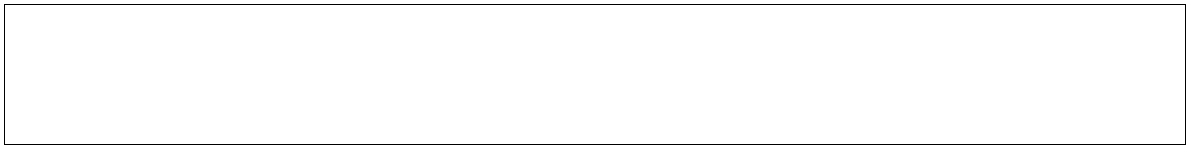
Possible Error Conditions:

None.



5.10 TTOperationalStatusAlarmControl

This class extends `AlarmControl` to control alarms generated when a monitored model element of type `OperationalStatus` transitions to a new value. This type of alarm is called “transition triggered” (abbreviated “TT”).



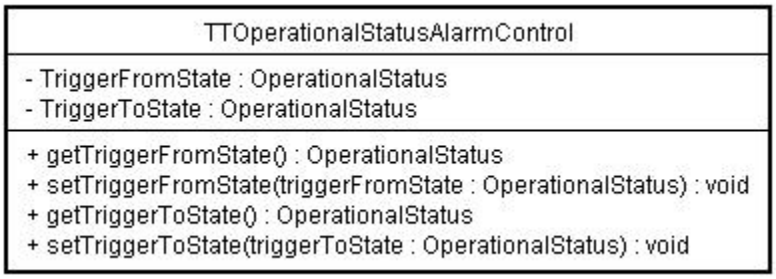


Figure 17 TTOperationalStatusAlarmControl UML

The `TriggerFromState` attribute (of enumerated type `OperationalStatus`) indicates the value of the monitored, `OperationalStatus`-valued, model element prior to the state transition which triggers the alarm. The `TriggerToState` attribute (of enumerated type `OperationalStatus`) indicates the value of the monitored, `OperationalStatus`-valued, model element immediately after the state transition which triggers the alarm.

5.10.1 TTOperationalStatusAlarmControl.getTriggerFromState

Queries the reader for the current value of the `TTOperationalStatusAlarmControl`'s `TriggerFromState` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
TTOperationalStatusAlarmControl.getTriggerFromState(void):
OperationalStatus
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus.` The return data type is an enumerated type with possible values UNKNOWN, OTHER, UP, DOWN and ANY.

Possible Error Conditions:

None.

5.10.2 TTOperationalStatusAlarmControl.setTriggerFromState

Command sent to the reader to set the current value of the `TTOperatonalStatusAlarmControl`'s `TriggerFromState` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
TTOperationalStatusAlarmControl.setTriggerFromState
(triggerFromState:  OperationalStatus): void
```

Parameter(s):

operationalStatus – Data Type: `OperationalStatus.` The desired value of `TriggerFromState.`

Return Value(s):

Data Type: `void.`

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

5.10.3 TTOperationalStatusAlarmControl.getTriggerToState

Queries the reader for the current value of the `TTOperationalStatusAlarmControl`'s `TriggerToState` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
TTOperationalStatusAlarmControl.getTriggerToState (void):
OperationalStatus
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus.` The return data type is an enumerated type with possible values UNKNOWN, OTHER, UP, DOWN and ANY.

Possible Error Conditions:

None.

5.10.4 TTOperationalStatusAlarmControl.setTriggerToState

Command sent to the reader to set the current value of the `TTOperatonalStatusAlarmControl`'s `TriggerToState` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
TTOperationalStatusAlarmControl.setTriggerToState
(triggerToState: OperationalStatus): void
```

Parameter(s):

operationalStatus – Data Type: `OperationalStatus.` The desired value of `TriggerToState.`

Return Value(s):

Data Type: `void.`

Possible Error Conditions:

ERROR_AUTHORIZATION. A management application may not be authorized to alter alarm controls.

6 Reader Layer – Alarm Notifications

6.1 Alarm Objects

The AlarmControl class, and its subclasses, specify those elements of a Reader's object model that control alarm generation. The alarms themselves are messages that are sent to management systems and host applications whose responsibility it is to monitor Reader health and operations. The Reader object model has been expanded to include class definitions that specify, in an abstract manner, the contents of these alarm messages. Note that these `Alarm` objects do not specify persistent data objects maintained by the readers. These objects are instantiated in the course of a reader's generation or a management system's receipt and processing of alarms; once an alarm message is transmitted or processed, its corresponding information object disappears.

In the object descriptions below, the UML for each object is presented with its elements and operations or methods. Each object element has an associated getter. Since `Alarms` can only be generated by the reader device and can not be modified by the consumer, no setters are provided. Getters are used during message processing.

6.2 Alarm

`Alarm` is the base of all the classes within the object model that define the contents of alarm messages.

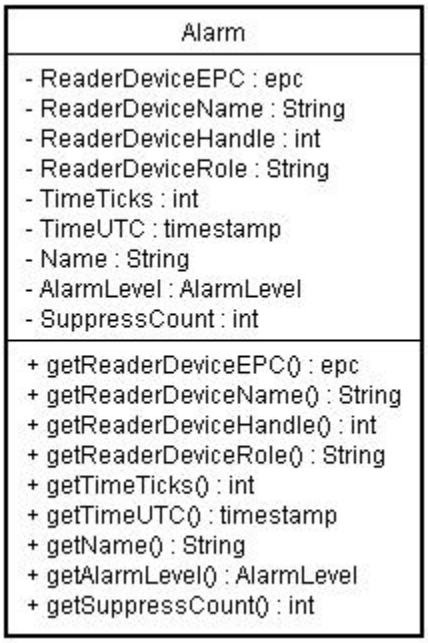


Figure 18 Alarm UML

```
Each Alarm MAY carry a ReaderDeviceEPC attribute (of type epc)
specifying the EPC of the reader that generated the alarm.  The
reader EPC is  ReaderDevice.EPC element of the Object model.
```

2810 Each `Alarm` SHALL carry a `ReaderDeviceName` attribute (of type `String`)
2811 specifying the name of the reader that generated the alarm. The reader name is
2812 `ReaderDevice.Name` element of the Object model.

2813 Each `Alarm` MAY carry a `ReaderDeviceHandle` attribute (of type `integer`)
2814 specifying the handle of the reader that generated the alarm. The reader handle is
2815 `ReaderDevice.Handle` element of the Object model.

2816 Each `Alarm` MAY carry a `ReaderDeviceRole` attribute (of type `String`)
2817 specifying the Role of the reader that generated the alarm. The reader Role is
2818 `ReaderDevice.Role` element of the object model.

2819 Each `Alarm` SHALL also carry a `TimeTicks` attribute (of type `integer`) recording the
2820 number of ticks when the alarm was generated. It's value is the value of the
2821 `ReaderDevice.TimeTicks` element of the Object model at the time the `Alarm` was
2822 generated.

2823 Each `Alarm` MAY also carry a `TimeUTC` attribute (of type `timestamp`) recording when
2824 the alarm was generated. It's value is the value of the `ReaderDevice.TimeUTC`
2825 element of the object model at the time the `Alarm` was generated.

2826 Each `Alarm` SHALL carry a `Name` attribute (of type `String`) identifying the type of
2827 `Alarm`, e.g., "FreeMemoryAlarm", "TagListFullAlarm", "ReadPointOperStatusAlarm".
2828 It SHALL matched on of the Alarm types specified in this section in this
2829 specification.Each `Alarm` SHALL carry an `AlarmLevel` attribute (of type
2830 `AlarmLevel`), indicating the severity level assigned to the alarm.

2831 Each `Alarm` SHALL carry a SuppressCount attribute (of type `integer`), indicating the
2832 number of times the generation of this Alarm has been suppressed. This attribute is
2833 incremented every time an alarm condition is encountered while the
2834 `SuppressInterval` is in effect, preventing the reader from generating the `Alarm`.
2835 This value is reset to 0 after the `Alarm` is generated. Suppression is controlled by the
2836 `SuppressInterval` attribute of `AlarmControl`.

2837 Expanding on the example presented in the `AlarmControl` section, a
2838 `FailedLockAlarm` is generated at 09:00:00. At 09:00:05, the alarm condition is
2839 encountered again while the `SuppressInterval` has not yet been reached. At this
2840 time, the `SuppressCount` for AntennaReadPoint1.FailedKillAlarmControl is
2841 incremented by one and the alarm generation is suppressed. The same occurs at
2842 09:00:08, incrementing `SuppressCount` by 1 again while suppressing the `Alarm`. At
2843 09:00:16, the `Alarm` is generated containing a `SuppressCount` equal to 2, indicating
2844 the `Alarm` was not sent on 2 prior occassions during the interval from 09:00:00 (first
2845 alarm) until now. The `SuppressCount` for this object will then be reset to 0 after the
2846 successful generation of the `Alarm`.

2847

Time	**Object**	**Alarm Condition**	**Action**	**Suppress Count**

09:00:00	AntennaReadPoint1	Kill Failure	Generate Alarm	0
09:00:01	AntennaReadPoint2	Kill Failure	Generate Alarm	0
09:00:05	AntennaReadPoint1	Kill Failure	Suppress Alarm	Increment
09:00:06	AntennaReadPoint1	Write Failure	Generate Alarm	0
09:00:08	AntennaReadPoint1	Kill Failure	Suppress Alarm	Increment
09:00:16	AntennaReadPoint1	Kill Failure	Generate Alarm	2

6.2.1 Alarm.getReaderDeviceEPC

Queries the Alarm for the value of the `ReaderDeviceEPC` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems MAY implement this command.

Usage:

```
Alarm.getReaderDeviceEPC(void): epc
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `epc. The EPC for the reader device.`

Possible Error Conditions:

None.

6.2.2 Alarm.getReaderDeviceName

Queries the Alarm for the value of the `ReaderDeviceName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
Alarm.getReaderDeviceName(void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The name for the reader device.`

Possible Error Conditions:

None.

6.2.3 Alarm.getReaderDeviceHandle

Queries the Alarm for the value of the `ReaderDeviceHandle` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`Alarm.getReaderDeviceHandle(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The handle for the reader device.`

Possible Error Conditions:

None.

6.2.4 Alarm.getReaderDeviceRole

Queries the Alarm for the value of the `ReaderDeviceRole` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`Alarm.getReaderDeviceRole(void): string`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The role for the reader device.`

Possible Error Conditions:

None.

6.2.5 Alarm.getTimeTicks

Queries the Alarm for the value of the `TimeTicks` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
Alarm.getTimeTicks(void): integer
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The TimeTicks for the reader device.`

Possible Error Conditions:

None.

6.2.6 Alarm.getTimeUTC

Queries the Alarm for the value of the `TimeUTC` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

2938 `Alarm.getTimeUTC(void): timestamp`

2939

2940 **Parameter(s):**

2941 Data Type: `void.` This command takes no parameters.

2942

2943 **Return Value(s):**

2944 Data Type: `timestamp. The UTC time for the reader device.`

2945

2946 **Possible Error Conditions**:

2947 None.

2948 6.2.7 Alarm.getName

2949 Queries the Alarm for the value of the `Name` attribute.

2950

2951 **Compliance Requirement**: If this alarm is implemented, compliant systems SHALL
2952 implement this command.

2953

2954 **Usage**:

2955 `Alarm.getName(void): string`

2956

2957 **Parameter(s):**

2958 Data Type: `void.` This command takes no parameters.

2959

2960 **Return Value(s):**

2961 Data Type: `string. The name for the alarm.`

2962

2963 **Possible Error Conditions**:

2964 None.

2965 6.2.8 Alarm.getAlarmLevel

2966 Queries the Alarm for the value of the `AlarmLevel` attribute.

2967

2968 **Compliance Requirement**: If this alarm is implemented, compliant systems SHALL
2969 implement this command.

Usage:

```
Alarm.getAlarmLevel(void): AlarmLevel
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `AlarmLevel. The alarm severity level.`

Possible Error Conditions:

None.

6.2.9 Alarm.getSuppressCount

Queries the Alarm for the value of the `SupressCount` attribute.

Compliance Requirement: Compliant systems SHALL implement this command.

Usage:

```
Alarm.getSuppressCount(void): integer
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The number of times since last alarm generation that this alarm has been suppressed from being generated.`

Possible Error Conditions:

None.

6.3 FreeMemoryAlarm

`FreeMemoryAlarm` extends the Alarm class. Its receipt signals the movement of a reader device's free memory (represented in the abstract model by ReaderDevice.FreeMemory) below a specified threshold value. The abstract model's `ReaderDevice.FreeMemoryAlarmControl` object controls the triggering of alarms of this type.

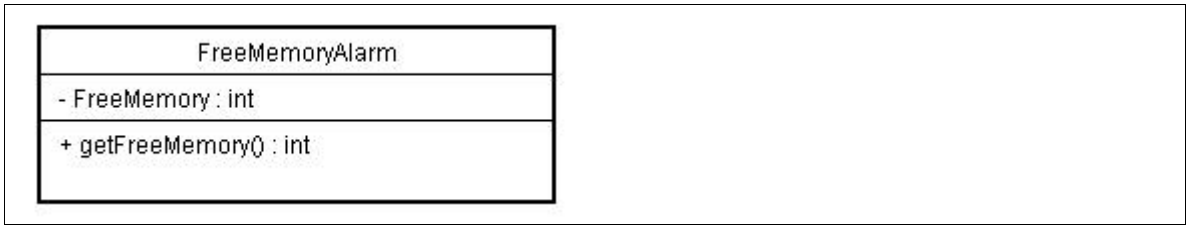


Figure 19 FreeMemoryAlarm UML

The `FreeMemory` attribute SHALL carry the value of `ReaderDevice.FreeMemory` when the alarm was triggered.

Compliance Requirement: Compliant systems MAY implement this alarm.

6.3.1 FreeMemoryAlarm.getFreeMemory

Queries the FreeMemoryAlarm for the value of the `FreeMemory` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FreeMemoryAlarm.getFreeMemory(void): integer
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The free memory for the reader device.`

Possible Error Conditions:

None.

6.4 FailedWriteAlarm

`FailedWriteAlarm` extends the Alarm class. Its receipt signals a tag write or tag block write failure. The abstract model's `AntennaReadPoint.FailedWriteAlarmControl` data element controls the triggering of alarms of this type.

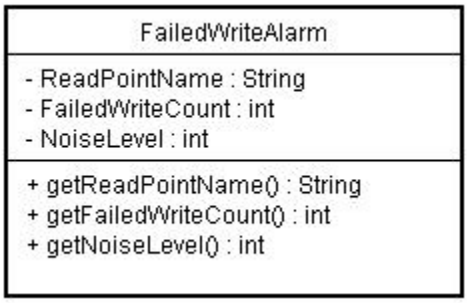


Figure 20 FailedWriteAlarm UML

The `ReadPointName` attribute identifies the read point (an `AntennaReadPoint`) over which the write failure occurred. This SHALL be the value of that read point's `ReadPoint.Name` element.

The `FailedWriteCount` attribute SHALL carry the value of `AntennaReadPoint.FailedWriteCount` element after the write failure occurred.

The `NoiseLevel` attribute MAY carry the value of the `AntennaReadPoint.NoiseLevel` element when the write failure occurred.

Compliance Requirement: Compliant systems MAY implement this alarm.

6.4.1 FailedWriteAlarm.getReadPointName

Queries the FailedWriteAlarm for the value of the `ReadPointName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FailedWriteAlarm.getReadPointName(void): string
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the ReadPoint.`

Possible Error Conditions:

None.

6.4.2 FailedWriteAlarm.getFailedWriteCount

Queries the FailedWriteAlarm for the value of the `FailedWriteCount` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FailedWriteAlarm.getFailedWriteCount(void): integer
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `integer. The FailedWriteCount for the ReadPoint.`

Possible Error Conditions:

None.

6.4.3 FailedWriteAlarm.getNoiseLevel

Queries the FailedWriteAlarm for the value of the `NoiseLevel` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`FailedWriteAlarm.getNoiseLevel(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The NoiseLevel for the ReadPoint.`

Possible Error Conditions:

None.

6.5 FailedEraseAlarm

`FailedEraseAlarm` extends the Alarm class. Its receipt signals a tag erase or tag block erase failure. The abstract model's `AntennaReadPoint.FailedEraseAlarmControl` data element controls the triggering of alarms of this type.

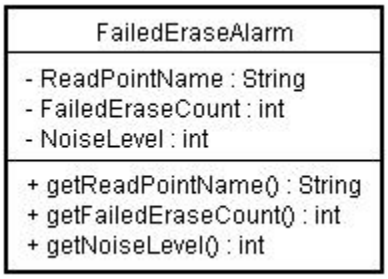


Figure 21 FailedEraseAlarm UML

The `ReadPointName` attribute identifies the read point (an `AntennaReadPoint`) over which the erase failure occurred. This SHALL be the value of that read point's `ReadPoint.Name` element.

The `FailedEraseCount` attribute SHALL carry the value of `AntennaReadPoint.FailedEraseCount` element after the erase failure occurred.

The `NoiseLevel` attribute MAY carry the value of the `AntennaReadPoint.NoiseLevel` element when the erase failure occurred.

Note that the erase failure could be attributable to the use of an incorrect tag erase password.

Compliance Requirement: Compliant systems MAY implement this alarm.

6.5.1 FailedEraseAlarm.getReadPointName

Queries the FailedEraseAlarm for the value of the `ReadPointName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FailedEraseAlarm.getReadPointName(void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the ReadPoint.`

Possible Error Conditions:

None.

6.5.2 FailedEraseAlarm.getFailedEraseCount

Queries the FailedEraseAlarm for the value of the `FailedEraseCount` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FailedEraseAlarm.getFailedEraseCount(void): integer
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The FailedEraseCount for the ReadPoint.`

Possible Error Conditions:

None.

6.5.3 FailedEraseAlarm.getNoiseLevel

Queries the FailedEraseAlarm for the value of the `NoiseLevel` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`FailedEraseAlarm.getNoiseLevel(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

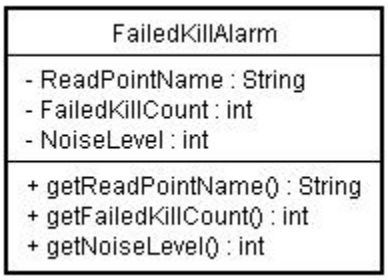
Data Type: `integer. The NoiseLevel for the ReadPoint.`

Possible Error Conditions:

None.

6.6 FailedKillAlarm

`FailedKillAlarm` extends the Alarm class. Its receipt signals a tag kill failure. The abstract model's `AntennaReadPoint.FailedKillAlarmControl` data element controls the triggering of alarms of this type.



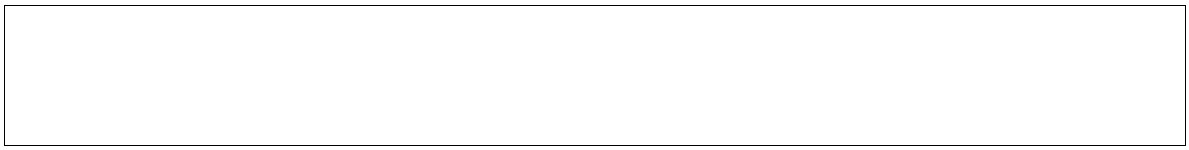


Figure 22 FailedKillAlarm UML

The `ReadPointName` attribute identifies the read point (an `AntennaReadPoint`) over which the kill failure occurred. This SHALL be the value of that read point's `ReadPoint.Name` element.

The `FailedKillCount` attribute SHALL carry the value of `AntennaReadPoint.FailedKillCount` element after the kill failure occurred.

The `NoiseLevel` attribute MAY carry the value of the `AntennaReadPoint.NoiseLevel` element when the kill failure occurred.

Note that the kill failure could be attributable to the use of an incorrect tag kill password.

Compliance Requirement: Compliant systems MAY implement this alarm.

6.6.1 FailedKillAlarm.getReadPointName

Queries the FailedKillAlarm for the value of the `ReadPointName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FailedKillAlarm.getReadPointName(void): string
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the ReadPoint.`

Possible Error Conditions:

None.

6.6.2 FailedKillAlarm.getFailedKillCount

Queries the FailedKillAlarm for the value of the `FailedKillCount` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

`FailedKillAlarm.getFailedKillCount(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The FailedKillCount for the ReadPoint.`

Possible Error Conditions:

None.

6.6.3 FailedKillAlarm.getNoiseLevel

Queries the FailedKillAlarm for the value of the `NoiseLevel` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`FailedKillAlarm.getNoiseLevel(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The NoiseLevel for the ReadPoint.`

Possible Error Conditions:

None.

6.7 FailedLockAlarm

`FailedLockAlarm` extends the Alarm class. Its receipt signals a tag lock failure. The abstract model's `AntennaReadPoint.FailedLockAlarmControl` data element controls the triggering of alarms of this type.

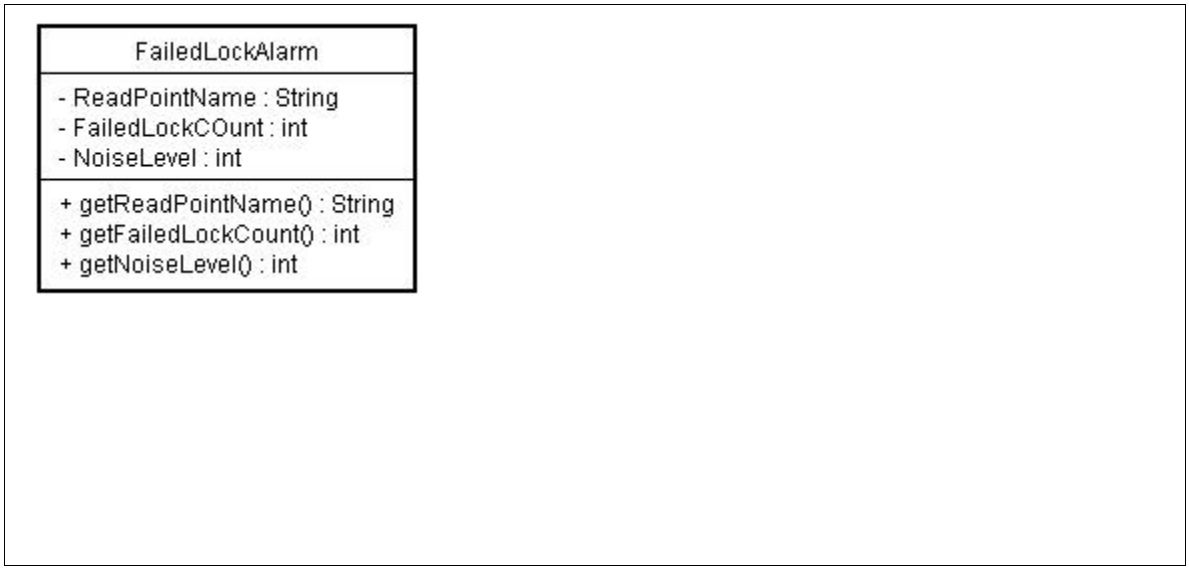


Figure 23 FailedKillAlarm UML

The `ReadPointName` attribute identifies the read point (an `AntennaReadPoint`) over which the lock failure occurred. This SHALL be the value of that read point's `ReadPoint.Name` element.

The `FailedLockCount` attribute SHALL carry the value of `AntennaReadPoint.FailedLockCount` element after the read failure occurred.

The `NoiseLevel` attribute MAY carry the value of the `AntennaReadPoint.NoiseLevel` element when the lock failure occurred.

Note that the lock failure could be attributable to the use of an incorrect tag lock password.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this alarm.

6.7.1 FailedLockAlarm.getReadPointName

Queries the FailedLockAlarm for the value of the `ReadPointName` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`FailedLockAlarm.getReadPointName(void): string`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the ReadPoint.`

Possible Error Conditions:

None.

6.7.2 FailedLockAlarm.getFailedLockCount

Queries the FailedLockAlarm for the value of the `FailedLockCount` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

`FailedLockAlarm.getFailedLockCount(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The FailedLockCount for the ReadPoint.`

Possible Error Conditions:

None.

6.7.3 FailedLockAlarm.getNoiseLevel

Queries the FailedLockAlarm for the value of the `NoiseLevel` attribute.

Compliance Requirement: Compliant systems MAY implement this command.

Usage:

`FailedLockAlarm.getNoiseLevel(void): integer`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `integer. The NoiseLevel for the ReadPoint.`

Possible Error Conditions:

None.

6.8 FailedMemReadAlarm

`FailedMemReadAlarm` extends the Alarm class. Its receipt signals a tag user-memory read failure. The abstract model's `AntennaReadPoint.FailedMemReadAlarmControl` data element controls the triggering of alarms of this type.

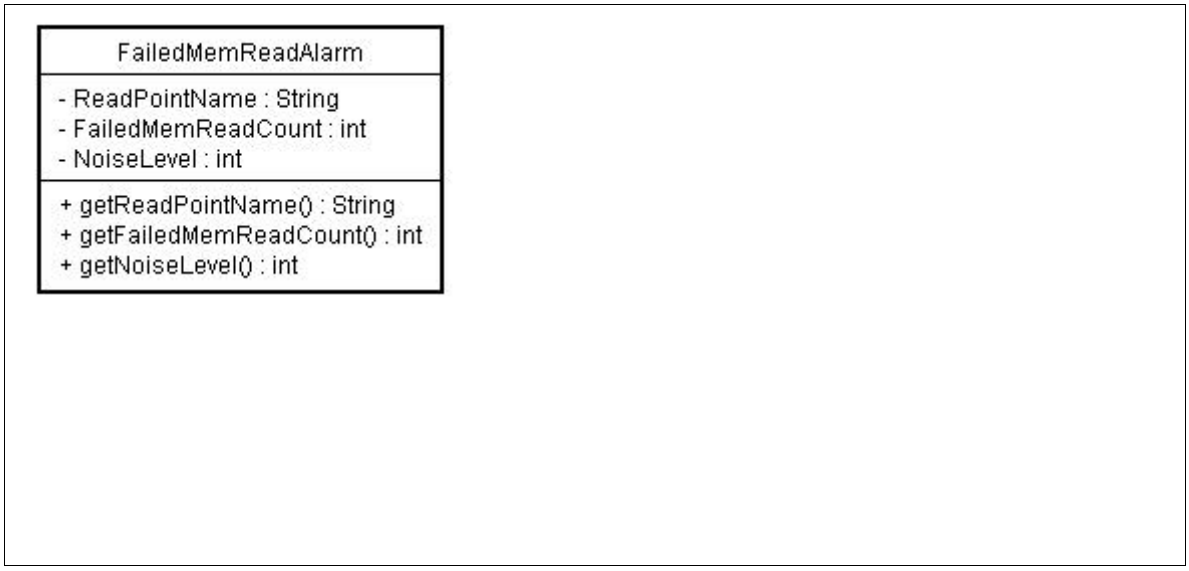


Figure 24 FailedMemReadAlarm UML

The `ReadPointName` attribute identifies the read point (an `AntennaReadPoint`) over which the memory read failure occurred. This SHALL be the value of that read point's `ReadPoint.Name` element.

The `FailedMemReadCount` attribute SHALL carry the value of `AntennaReadPoint.FailedMemReadCount` element after the read failure occurred.

The `NoiseLevel` attribute MAY carry the value of the `AntennaReadPoint.NoiseLevel` element when the memory read failure occurred.

Note that the read failure could be attributable to the use of an incorrect tag memory read password.

Compliance Requirement: Compliant systems MAY implement this alarm.

6.8.1 FailedMemReadAlarm.getReadPointName

Queries the FailedMemReadAlarm for the value of the `ReadPointName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
FailedMemReadAlarm.getReadPointName(void): string
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the ReadPoint.`

Possible Error Conditions:

None.

6.8.2 FailedMemReadAlarm.getFailedMemReadCount

Queries the FailedMemReadAlarm for the value of the `FailedMemReadCount` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

3367 `FailedMemReadAlarm.getFailedMemReadCount(void): integer`

3368

3369 **Parameter(s):**

3370 Data Type: `void.` This command takes no parameters.

3371

3372 **Return Value(s):**

3373 Data Type: `integer. The FailedMemReadCount for the ReadPoint.`

3374

3375 **Possible Error Conditions**:

3376 None.

3377 6.8.3 FailedMemReadAlarm.getNoiseLevel

3378 Queries the FailedMemReadAlarm for the value of the `NoiseLevel` attribute.

3379

3380 **Compliance Requirement**: Compliant systems MAY implement this command.

3381

3382 **Usage**:

3383 `FailedMemReadAlarm.getNoiseLevel(void): integer`

3384

3385 **Parameter(s):**

3386 Data Type: `void.` This command takes no parameters.

3387

3388 **Return Value(s):**

3389 Data Type: `integer. The NoiseLevel for the ReadPoint.`

3390

3391 **Possible Error Conditions**:

3392 None.

3393

6.9 TTOperStatusAlarm

`TTOperStatusAlarm` extends the `Alarm` class, and is the base class for all Transition Triggered Operational Status Alarms. The base class identifies the particular transition between Operational Status states that triggered the alarm.

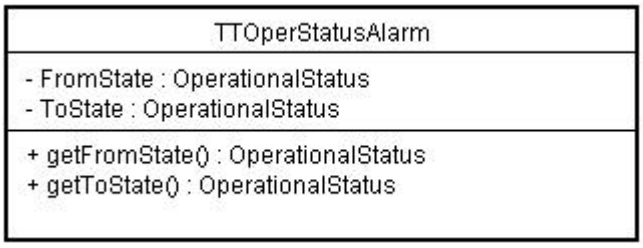


Figure 25 TTOperStatusAlarm UML

The `FromState` attribute SHALL identify the originating `OperationalStatus` before the Alarm is generated.

The `ToState` attribute SHALL identify the `OperationalStatus` at the time the Alarm is generated.

```
Valid values for OperationalStatus are defined in 7.1.2
OperationalStatus.
```

6.9.1 TTOperStatusAlarm.getFromState

Queries the TTOperStatusAlarm for the value of the `OperationalStatus` attribute before the transition.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

`TTOperStatusAlarm.getFromState(void): OperationalStatus`

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus. The OperationalStatus before the transition.`

Possible Error Conditions:

None.

6.9.2 TTOperStatusAlarm.getToState

Queries the TTOperStatusAlarm for the value of the `OperationalStatus` attribute after the transition.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

`TTOperStatusAlarm.getToState(void): OperationalStatus`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `OperationalStatus. The OperationalStatus after the transition.`

Possible Error Conditions:

None.

6.10 ReaderDeviceOperStatusAlarm

`ReaderDeviceOperStatusAlarm` extends the `TTOperStatusAlarm` class. Its receipt signals a change in the operational status of a Reader. The abstract model's `ReaderDevice.OperStatusAlarmControl` data element controls the triggering of alarms of this type.

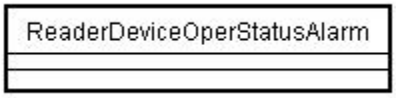


Figure 26 ReaderDeviceStatusAlarm UML

Compliance Requirement: Compliant systems SHALL implement this alarm if AlarmChannel is implemented.

6.11 IOPortOperStatusAlarm

`IOPortOperStatusAlarm` extends the TT`OperStatusAlarm` class. Its receipt signals a change in the operational status of a Reader's IO Port. The abstract model's `IOPort.OperStatusAlarmControl` data element controls the triggering of alarms of this type.

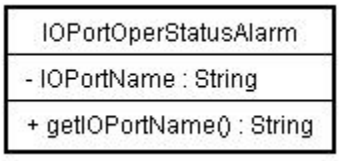


Figure 27 IOPortOperStatusAlarm UML

The `IOPortName` message attribute SHALL identify the name of the IO Port that experienced the alarm-triggering state transition, i.e., the value of the respective `IOPort.Name` model element.

Compliance Requirement: Compliant systems SHALL implement this alarm if AlarmChannel and IOPort are implemented.

6.11.1 IOPortOperStatusAlarm.getIOPortName

Queries the IOPortStatusAlarm for the value of the `IOPortName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
IOPortOperStatus.getIOPortName(void): string
```

Parameter(s):

Data Type: `void`. This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the IOPort.`

Possible Error Conditions:

None.

6.12 ReadPointOperStatusAlarm

`ReadPointOperStatusAlarm` extends the `TTOperStatusAlarm` class. Its receipt signals a change in the operational status of one of a Reader's Read Points. The abstract model's `ReadPoint.OperStatusAlarmControl` data element controls the triggering of alarms of this type.

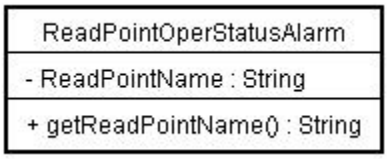


Figure 28 ReadPointOperStatusAlarm UML

The `ReadPointName` message attribute SHALL identify the name of the Read Point that experienced the alarm-triggering state transition, i.e., the value of the respective `ReadPoint.Name` model element.

Compliance Requirement: Compliant systems SHALL implement this alarm if AlarmChannel is implemented.

6.12.1 ReadPointOperStatusAlarm.getReadPointName

Queries the ReadPointOperStatusAlarm for the value of the `ReadPointName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

`ReadPointOperStatusAlarm.getReadPointName (void): string`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the ReadPoint.`

Possible Error Conditions:

None.

6.13 SourceOperStatusAlarm

`SourceOperStatusAlarm` extends the `TTOperStatusAlarm` class. Its receipt signals a change in the operational status of a logical source of EPC data on a Reader. The abstract model's `Source.OperStatusAlarmControl` data element controls the triggering of alarms of this type.

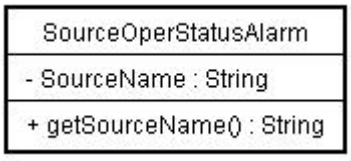


Figure 29 SourceOperStatusAlarm UML

The `SourceName` message attribute SHALL identify the name of the logical source that experienced the alarm-triggering state transition, i.e., the value of the respective `Source.Name` model element.

Compliance Requirement: Compliant systems SHALL implement this alarm if AlarmChannel is implemented.

6.13.1 SourceOperStatusAlarm.getSourceName

Queries the SourceOperStatusAlarm for the value of the `SourceName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

`SourceOperStatusAlarm.getSourceName (void): string`

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the Source.`

Possible Error Conditions:

None.

6.14 NotificationChannelOperStatusAlarm

`NotificationChannelOperStatusAlarm` extends the `TTOperStatusAlarm` class. Its receipt signals a change in the operational status of one of a Reader's notification channels. The abstract model's `NotificationChannel.OperStatusAlarmControl` data element controls the triggering of alarms of this type.

Figure 30 NotificationChannelOperStatusAlarm UML

The `NotificationChannelName` message attribute SHALL identify

the name of the notification channel that experienced the alarm-triggering state transition, i.e., the value of the respective `NotificationChannel.Name` model element.

Compliance Requirement: Compliant systems SHALL implement this alarm if AlarmChannel and NotificationChannel are implemented.

6.14.1 NotificationChannelOperStatusAlarm.getNotificationChannelName

Queries the NotificationChannelOperStatusAlarm for the value of the `NotificationChannelName` attribute.

Compliance Requirement: If this alarm is implemented, compliant systems SHALL implement this command.

Usage:

```
NotificationChannelOperStatusAlarm.getNotificationChannelNa
me(void): string
```

Parameter(s):

Data Type: `void.` This command takes no parameters.

Return Value(s):

Data Type: `string. The name of the NotificationChannel.`

Possible Error Conditions:

None.

7 Enumerated types

7.1.1 AdministrativeStatus

The administrative status represents the host's desired state for an object.

Value	Description
UP	The host desires that the operational status be "up"
DOWN	The host desires that the operational status be "down"

7.1.2 OperationalStatus

The operational status represents the actual state for an object.

Value	Description
UNKNOWN	The reader lacks the ability to determine the operational status of the object. The real status may be "up" ,"down", or "other".
UP	The object is operational. For aggregate objects (i.e. objects that encompass other objects, as a source may encompass multiple read points) this means that all objects that are administratively up are either operationally up or operationally unknown.

DOWN	The object is not operational. For aggregate objects (i.e. objects that encompass other objects, as a source may encompass multiple read points) this means that all objects that are administratively up are either operationally down or operationally unknown.
OTHER	The object is neither fully up, nor fully down. For aggregate objects (i.e. objects that encompass other objects, as a source may encompass multiple read points) this includes the case where all sub-objects are administratively down and the aggregate object is administratively up.
ANY	A wildcard operator. The comparison "ANY"==S is true for all status enumerations, S. This is never returned by the reader, but is useful to hosts describing events. For example a host may desire an alert when the reader state changed from UP to ANY.

7.1.3 EdgeTriggeredAlarmDirection

Controls the direction in which an edge-triggered alarm should fire.

Value	Description
RISING	Trigger on the rising edge
FALLING	Trigger on the falling edge

7.1.4 EdgeTriggeredAlarmStatus

Describes the status of an edge-triggered alarm. See the Alarm object description for an operational model.

Value	Description
ARMED	Alarm will fire when the monitored value crosses the alarm threshold
FIRED	Alarm has fired; monitored value has not yet crossed re-arm threshold

7.1.5 AlarmLevel

The level of a reported alarm. When the host requests an alarm it defines the alarm level (see AlarmControl.setLevel). This additional information is then returned by the reader along with each occurrence of that alarm. It is expected that some management systems may choose to filter based on the alarm level. The alarm levels are modeled after The syslog Protocol IETF draft [SYSLOG].

Value	Description

EMERGENCY	System is unusable
ALERT	Action must be taken immediately
CRITICAL	Critical conditions
ERROR	Error conditions
WARNING	Warning conditions
NOTICE	Normal but significant conditions
INFORMATIONAL	Informational messages
DEBUG	Debug level messages

8 Error Handling

8.1 Error Conditions

There are several types of error conditions:

- Communication errors, either when a host issues a command or when the reader tries to send a notification (e.g., tag list report) to the host
- Command execution errors that occur as response to commands sent from the host to the reader.

Vendor extensions

This section lists the errors that are unique to the Reader Management Specification. Refer to the Reader Protocol Specification 1.1 for the definition of errors inherited from the the Reader Protocol Specification 1.1.

8.2 Communication Errors

8.2.1 Communication Host-to-Reader

These kinds of communication errors occur when the host tries to issue a command to the reader but the reader is not responding. These conditions are reported by the MTB-specific Transport Layer.

8.2.2 Communication Reader-to-Host

This specification does not define how the reader reacts when it isn't able to connect or exchange data over a notification channel; the exact behavior is implementation

dependent. Notifications could be dropped, sent over a secondary channel, queued for later retry, raise a trap in a management console, and/or many other behaviors.

8.3 Command Errors

All commands support a standard `error` structure, for use in the event of errors. This contains standard attributes (some required, some optional) as well as support for vendor extensions.

All standard error conditions defined can be reported with additional data about the error (formatted and transmitted as per the MTB in use):

- SHALL: The **error code** and/or **error name**. At least one of these attributes SHALL be included in the `Error` structure. The exact contents depend on the binding, and SHALL be constant at the binding level, e.g. MTB X can be defined as always including the **error code** (but not the **error name**)
- SHALL (where applicable): Information about the **error cause**. For example, for input parameter errors, information on which parameter failed needs to be included. Depending on the binding, this can be either the name or the index of the parameter. Vendor extensions are allowed here.
- SHALL (where applicable): A **vendor name or identifier** for the responses to vendor-specific commands SHALL be given.
- MAY: A **descriptive text string**, mainly used for logging purposes. The language for the text is implementation-dependent and MAY be configurable.

The error code is a 16-bit (2-byte) integer. The following table defines the mapping of error conditions / error codes unique to Reader Management. Refer to Reader Protocol Specification 1.1 for errors imported from it.

Error Code (Hex)	Error Name	Description
0001	`ERROR_UNKNOWN`	Refer to Reader Protocol Specification 1.1
0002	`ERROR_COMMAND_NOT_SUPPORTED`	Refer to Reader Protocol Specification 1.1
0003	`ERROR_PARAMEER_INVALID_FORMAT`	Refer to Reader Protocol Specification 1.1
0004	`ERROR_PARAMETER_MISSING`	Refer to Reader Protocol Specification 1.1

Error Code (Hex)	Error Name	Description
0009	ERROR_PARAMETER_LENGTH_EXCEEDED	Refer to Reader Protocol Specification 1.1
000D	ERROR_TRIGGER_NOT_FOUND	Refer to Reader Protocol Specification 1.1
000E	ERROR_READPOINT_NOT_FOUND	Refer to Reader Protocol Specification 1.1
0010	ERROR_SOURCE_NOT_FOUND	Raised when the given Source parameter is not known
1000	ERROR_IOPORT_NOT_FOUND	Raised when the given IOPort parameter is not known
1001	ERROR_ALARM_CHANNEL_NOT_FOUND	Raised when the given AlarmChannel parameter is not known
1002	ERROR_INVALID_ADMIN_STATUS	Raised when the specified Administrative status is not allowed for the given state of the object
1003	ERROR_TOO_MANY_ALARM_CHANNELS	Raised when the command failed due to an internal resource restriction. For instance, not enough memory, maximum number of objects reached, etc.
1004	ERROR_AUTHORIZATION	Raised when the command failed due to lack of rights to perform the operation on the object.

3660 **Table 1: Basic Reader Management Error Codes**

9 Vendor Extensions

Vendors SHALL use the error conditions defined above for the commands of the standard command set. Vendors MAY add additional information on the exact cause of the error.

Vendors can extend the standard Reader Management command set. These extension commands MAY use the same general error conditions as defined above, however they MAY also define additional error conditions. These vendor-defined error conditions SHALL use names starting with `<VENDOR>_ERROR`..., for example `ACME_ERROR_WHATSIT_NOT_ADDED_NO_FLABOOZLE` if Acme Corp. needed an error condition indicating there was no WHATSIT associated with a FLABOOZLE.

Vendor extensions may be:

- Additional commands defined for the following objects:

  AlarmChannel
  AlarmControl
  AntennaReadPoint
  EdgeTriggeredAlarmControl
  IOPort

  NotificationChannel
  ReaderDevice
  ReadPoint
  Source

  Trigger
  TTOperationalStatusAlarmControl

- Additional attributes defined for the following objects:

  Alarm

  FailedKillAlarm
  FailedLockAlarm
  FailedMemReadAlarm

  FailedWriteAlarm
  FreeMemoryAlarm
  IOPortOperStatusAlarm
  NotificationChannelOperStatusAlarm
  OperStatusAlarm
  ReaderDeviceOperStatusAlarm
  ReadPointOperStatusAlarm
  SourceOperStatusAlarm

- Commands to additional objects.

- Additional types of Alarm (i.e. additional subclasses of Alarm).

The exact syntax for expressing vendor extensions is detailed for each Message/Transport Binding (MTB).

10 Message/Transport Bindings (MTBs)

The previous section defined an Abstract Command Set independent of message format or communication transport. This section defines the actual mapping of the Command Set to the specific Message Formats defined by the Reader Protocol Version 1.0 and SNMP. Each message format is independent of the other. A compliant application SHALL implement at least one of the Message Formats and MAY implement more than one of the message formats. If a particular Message Format is implemented, the compliant application SHALL implement all of the operations labeled SHALL. If an operation is not supported, the Reader and/ or Host SHALL reply with ERROR_NOT_SUPPORTED.

Currently, two message formats are specified, XML, and SNMP. The XML message format can be used over any of the 3 transports (Serial, TCP, and HTTP) defined by the Reader Protocol Specification 1.1. In the future, additional message formats and transports may be defined.

10.1 Address Notation

All transports support a URI naming convention when an address is required by the command set, e.g. `NotificationChannel.setAddress()`. This uses the following notation except where explicitly stated otherwise:

```
<transport>://<locator>
```

For example:

```
tcp://foo.bar.com:2000
serial://com1:3
```

http://myreaderisawebserver:7183

Note 1: in the case of a serial connection, where multiple channels have to be multiplexed over a single connection, the 'port' is a number that identifies the channel.

Within the SNMP MIB, addresses are defined in compliance with RFC 3291/RFC 4001, the INET-ADDRESS MIB. This allows for addressing via IPv4, IPv6, or DNS Fully Qualified Domain Name.

10.2 Vendor Extension Details

Refer to the Reader Protocol 1.1 Vendor Extensions for the syntax used to define new Commands, Errors, and Attributes using the XML Message format.

Vendor Extensions for the SNMP binding are implemented through the use of a vendor specific MIB. The Vendor specific MIB may reference the EPCglobal Reader Management MIB.

The management application needs to know a priori the vendor extensions applicable to a reader. The XML schema provided by the reader vendor defines the

operations (standard and extensions). SNMP provides for extensions by defining a separate MIB applicable to the reader model/vendor.

10.3 XML Message Format

Refer to the Reader Protocol Version 1.1 Specification for the XML Binding rules, Object identification Scheme and Data Types. The following sections provide the Reader Management specific schema definitions.

10.3.1 Command XML Message Encoding (Host-To-Reader)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="urn:epcglobal:rm:xsd:1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0" xmlns:rm="urn:epcglobal:rm:xsd:1"
xmlns:epcglobal="urn:epcglobal:xsd:1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <xsd:annotation>
      <xsd:documentation xml:lang="en">
         <epcglobal:copyright>Copyright&amp;#169;2005-2006 Epcglobal Inc., All Rights Reserved.</epcglobal:copyright>
         <epcglobal:disclaimer>EPCglobal Inc., its members, officers, directors, employees, or
            agents shall not be liable for any injury, loss, damages, financial or otherwise,
            arising from, related to, or caused by the use of this document. The use of said
            document shall constitute your express consent to the foregoing exculpation.</epcglobal:disclaimer>
         <epcglobal:specification>Reader Management (RM) version 1.0</epcglobal:specification>
      </xsd:documentation>
   </xsd:annotation>
   <xsd:include schemaLocation="RmCommon.xsd"/>
   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="./EpcGlobal.xsd"/>
   <!-- Reader Management Command -->
   <xsd:element name="command">
      <xsd:annotation>
         <xsd:documentation xml:lang="en"> This element defines a reader management command.
The commands are grouped by the object they belong to. To specify the object a command relates to,
the name of the object is used. Therefore, all commands that return the name of the object
become useless, since the name must be known before it can be queried. All the names of a
particular object type are global irrespective of where they belong to. Hence names need to
be unique within their object type. All the commands are encoded by its straight forward
conversion from UML notation. Each command is wrapped in its object type and contains its
parameters as child elements. </xsd:documentation>
      </xsd:annotation>
      <xsd:complexType>
         <xsd:complexContent>
            <xsd:extension base="epcglobal:Document">
               <xsd:sequence>
                  <xsd:element name="id" type="xsd:string">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en">The id of the command</xsd:documentation>
                     </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="targetName" type="xsd:string" minOccurs="0">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en">The name of the object on which this command should be
executed. It can be left out for static methods/ or commands targeted to ReaderDevice object</xsd:documentation>
                     </xsd:annotation>
```

```
                </xsd:element>
                <xsd:choice>
                  <!-- Reader Management (Target) Object Types -->
                  <!-- elements inside are ordered in alphabetical order -->
                  <xsd:element name="alarmChannel" type="rm:AlarmChannelCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">An Alarm channel</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="alarmControl" type="rm:AlarmControlCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">AlarmControl is the base class for all of classes
within the RMP object model responsible for controlling the generation of alarm messages within an EPCglobal
compliant Reader.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="antennaReadPoint" type="rm:AntennaReadPointCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">Extension of Read point object for
antenna.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="edgeTriggeredAlarmControl" type="rm:EdgeTriggeredAlarmControlCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">This class extends AlarmControl to control alarms
generated when a monitored, integer-valued, model element first crosses a threshold value (the
AlarmThreshold).</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="iOPort" type="rm:IOPortCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">The hardware element that provides external input
and output lines to connect to other components outside the reader device.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="notificationChannel" type="rm:NotificationChannelCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">The notification channel carries messages
issued asynchronously by the Reader to the Host.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="readerDevice" type="rm:ReaderDeviceCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">A Reader</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="readPoint" type="rm:ReadPointCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">A ReadPoint</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="source" type="rm:SourceCommand">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">A Read source</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
```

```
                    <xsd:element name="trigger" type="rm:TriggerCommand">
                      <xsd:annotation>
                              <xsd:documentation xml:lang="en">A Read/Notify Trigger</xsd:documentation>
                      </xsd:annotation>
                    </xsd:element>
                    <xsd:element name="tTOperationalStatusAlarmControl"
type="rm:TTOperationalStatusAlarmControlCommand">
                      <xsd:annotation>
                              <xsd:documentation xml:lang="en">This class extends AlarmControl to control alarms
generated when a monitored model element of type OperationalStatus transitions to a new
value.</xsd:documentation>
                      </xsd:annotation>
                    </xsd:element>
                    <xsd:any namespace="##any" processContents="lax">
                      <xsd:annotation>
                              <xsd:documentation>For standard and vendor extensions</xsd:documentation>
                      </xsd:annotation>
                    </xsd:any>
                  </xsd:choice>
                </xsd:sequence>
              </xsd:extension>
            </xsd:complexContent>
          </xsd:complexType>
        </xsd:element>
        <!-- Reader Management (Target) Object Types -->
        <!-- types are ordered in alphabetical order -->
        <!-- inside each type commands are listed in the order in which it is defined in the RM specification. -->
        <xsd:complexType name="AlarmChannelCommand">
          <xsd:choice>
            <xsd:annotation>
              <xsd:documentation xml:lang="en"> Alarm channel object commands.</xsd:documentation>
            </xsd:annotation>
            <xsd:element name="create">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">static method to create an Alarm channel. </xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="name" type="xsd:string">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en"> Alarm channel name </xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                  <xsd:element name="addr" type="rm:AddressParamType">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en"> The (host) address of where alarms will be sent
to.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getName">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">get the alarm channel name</xsd:documentation>
              </xsd:annotation>
```

```
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getAddress">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">get the (host) address to which this AlarmChannel object sends its
alarms.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="setAddress">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">set the (host) address to which this AlarmChannel object sends its
alarms.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="addr" type="rm:AddressParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en"> address to set.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:any namespace="##any" processContents="lax">
            <xsd:annotation>
              <xsd:documentation>For standard and vendor extensions</xsd:documentation>
            </xsd:annotation>
          </xsd:any>
        </xsd:choice>
      </xsd:complexType>
      <xsd:complexType name="AlarmControlCommand">
        <xsd:choice>
          <xsd:annotation>
            <xsd:documentation xml:lang="en"> Alarm control object commands. </xsd:documentation>
          </xsd:annotation>
          <xsd:element name="getName">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">get the alarm control name</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getEnabled">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for the current value of the AlarmControlâ™s
Enabled attribute.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="setEnabled">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">enable or disable alarm generation.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="enable" type="xsd:boolean">
```

```
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">whether to enable the alarm generation.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getLevel">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">query the reader for the current value of the AlarmControlâ™s Level
attribute.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="setLevel">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">set the Level attribute.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="alarmLevel" type="rm:AlarmLevelParamType">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">Alarm level to set</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getSuppressInterval">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">query the reader for the current value of the AlarmControlâ™s
SuppressInterval attribute.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="setSuppressInterval">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">set the SuppressInterval attribute.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="suppressInterval" type="xsd:int">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">SuppressInterval to set in seconds</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:choice>
        </xsd:complexType>
```

```
    <xsd:complexType name="AntennaReadPointCommand">
      <xsd:choice>
        <xsd:annotation>
          <xsd:documentation xml:lang="en"> Antenna read point object commands </xsd:documentation>
        </xsd:annotation>
        <xsd:element name="getIdentificationCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of the successful tags that have been
            identified  across an AntennaReadPoint. </xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFailedIdentificationCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of the failed tag identification attempts at the
AntennaReadPoint. </xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getMemReadCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of tag memory reads at the AntennaReadPoint.
</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFailedMemReadCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of the failed tag memory reads at the
AntennaReadPoint. </xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFailedMemReadAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the AntennaReadPoint's failed memory read alarm
control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getWriteCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of successful tag writes at the AntennaReadPoint.
</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFailedWriteCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of the failed attempts to write tags at the
AntennaReadPoint. </xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFailedWriteAlarmControl">
          <xsd:annotation>
```

```
4079                <xsd:documentation xml:lang="en">get the AntennaReadPoint's failed write alarm
4080 control.</xsd:documentation>
4081              </xsd:annotation>
4082              <xsd:complexType/>
4083            </xsd:element>
4084            <xsd:element name="getKillCount">
4085              <xsd:annotation>
4086                <xsd:documentation xml:lang="en">get the number of tags successfully killed at the AntennaReadPoint.
4087 </xsd:documentation>
4088              </xsd:annotation>
4089              <xsd:complexType/>
4090            </xsd:element>
4091            <xsd:element name="getFailedKillCount">
4092              <xsd:annotation>
4093                <xsd:documentation xml:lang="en">get the number of the failed tag kills at the AntennaReadPoint.
4094 </xsd:documentation>
4095              </xsd:annotation>
4096              <xsd:complexType/>
4097            </xsd:element>
4098            <xsd:element name="getFailedKillAlarmControl">
4099              <xsd:annotation>
4100                <xsd:documentation xml:lang="en">get the AntennaReadPoint's failed kill alarm
4101 control.</xsd:documentation>
4102              </xsd:annotation>
4103              <xsd:complexType/>
4104            </xsd:element>
4105            <xsd:element name="getEraseCount">
4106              <xsd:annotation>
4107                <xsd:documentation xml:lang="en">get the number of tags successfully erased at the
4108 AntennaReadPoint.</xsd:documentation>
4109              </xsd:annotation>
4110              <xsd:complexType/>
4111            </xsd:element>
4112            <xsd:element name="getFailedEraseCount">
4113              <xsd:annotation>
4114                <xsd:documentation xml:lang="en">get the number of the failed tag erasures at the AntennaReadPoint.
4115 </xsd:documentation>
4116              </xsd:annotation>
4117              <xsd:complexType/>
4118            </xsd:element>
4119            <xsd:element name="getFailedEraseAlarmControl">
4120              <xsd:annotation>
4121                <xsd:documentation xml:lang="en">get the AntennaReadPoint's failed erase alarm
4122 control.</xsd:documentation>
4123              </xsd:annotation>
4124              <xsd:complexType/>
4125            </xsd:element>
4126            <xsd:element name="getLockCount">
4127              <xsd:annotation>
4128                <xsd:documentation xml:lang="en">get the number of tags successfully locked at the AntennaReadPoint.
4129 </xsd:documentation>
4130              </xsd:annotation>
4131              <xsd:complexType/>
4132            </xsd:element>
4133            <xsd:element name="getFailedLockCount">
4134              <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">get the number of the failed tag locks at the AntennaReadPoint.
</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFailedLockAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the AntennaReadPoint's failed lock alarm
control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getTimeEnergized">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the number of milliseconds the AntennaReadPoint has been
energized in order to communicate with tags.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getPowerLevel">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the current transmit power level.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getNoiseLevel">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the current noise level.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
    <xsd:complexType name="EdgeTriggeredAlarmControlCommand">
      <xsd:choice>
        <xsd:annotation>
          <xsd:documentation xml:lang="en"> Edge triggered alarm control object commands. </xsd:documentation>
        </xsd:annotation>
        <xsd:element name="getAlarmThreshold">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for the current value of the
EdgeTriggeredAlarmControlâ™s AlarmThreshold attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="setAlarmThreshold">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">set the current value of the EdgeTriggeredAlarmControlâ™s
AlarmThreshold attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
```

```
            <xsd:sequence>
              <xsd:element name="alarmThreshold" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">threshold to set</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getRearmThreshold">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for the current value of the
EdgeTriggeredAlarmControlâ™s RearmThreshold attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="setRearmThreshold">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">set the current value of the RearmThreshold
attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="rearmThreshold" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">threshold to set</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getDirection">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for the current value of the
EdgeTriggeredAlarmControlâ™s Direction attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="setDirection">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">set the current value of the EdgeTriggeredAlarmControlâ™s Direction
attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="direction" type="rm:EdgeTriggeredAlarmDirectionParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">direction to set</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getStatus">
          <xsd:annotation>
```

```
                <xsd:documentation xml:lang="en">query the reader for the current value of the
EdgeTriggeredAlarmControlâ™s Status attribute.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:choice>
        </xsd:complexType>
        <xsd:complexType name="IOPortCommand">
          <xsd:choice>
            <xsd:annotation>
              <xsd:documentation xml:lang="en">IO port object commands. </xsd:documentation>
            </xsd:annotation>
            <xsd:element name="getName">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">get the IO port name</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="getDescription">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">get a textual description of the IO-port.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="setDescription">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">set (associate) a textual description with an IO-
port.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="description" type="xsd:string">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">description to set</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getOperStatus">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">query for the operational status of an IO-port.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="getAdminStatus">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">query for the administrative status of an IO-port.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
```

```
        <xsd:element name="setAdminStatus">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">set the administrative status of an IO-port.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="adminstrativeStatus" type="rm:AdministrativeStatusParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">Adminstrative status to set</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getOperStatusAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the IOPort's operational status alarm
control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
    <xsd:complexType name="NotificationChannelCommand">
      <xsd:choice>
        <xsd:annotation>
          <xsd:documentation xml:lang="en">Notification channel object commands</xsd:documentation>
        </xsd:annotation>
        <xsd:element name="getLastNotificationAttempt">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the timestamp (TimeTicks) when the last attempt was made to
send a notification to the given address.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getLastSuccessfulNotification">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the timestamp (TimeTicks) when the last successful notification
was send to the given address.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getOperStatus">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the NotificationChannel for its OperationalStatus
object.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="setAdminStatus">
          <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">set the administrative status of Notification
channel.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="administrativeStatus" type="rm:AdministrativeStatusParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> administrative status to set. </xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getAdminStatus">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the NotificationChannel for its administrative
status.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getOperStatusAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the notification channel for its operational status alarm
control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
    <xsd:complexType name="ReaderDeviceCommand">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Reader Device object commands. </xsd:documentation>
      </xsd:annotation>
      <xsd:choice>
        <xsd:element name="getDescription">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the Reader for its user defined
description.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="setDescription">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">set the reader's user-defined description.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="description" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> description to set</xsd:documentation>
                </xsd:annotation>
```

```
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getLocationDescription">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for its user defined location
description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="setLocationDescription">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">set the reader's user-defined location
description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="locationDescription" type="xsd:string">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en"> location description to set. </xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getContact">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for its user-defined contact
description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="setContact">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">set the reader's user-defined contact
description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="contact" type="xsd:string">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en"> contact to set</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getSerialNumber">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for its serial number.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getOperStatus">
            <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">query the reader for its OperationalStatus
object.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getOperStatusAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for its operational status alarm
control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFreeMemory">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for its available free memory.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getFreeMemoryAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for its FreeMemoryAlarmControl
object.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getNTPServers">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for a list of NTP servers used by it to synchronize its
current UTC clock (TimeUTC).</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getDHCPServer">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">query the reader for the DHCP server currently used by the device for
DHCP requests.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getIOPort">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the IO Port with the specified name currently associated with this
Reader.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="name" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> IO port name </xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getAllIOPorts">
          <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">query the reader for all its IOPort objects.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="resetStatistics">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reset reader's entire internal statistic counters to
zero.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="removeAlarmChannels">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">remove the specified AlarmChannels from the list of AlarmChannels
currently associated with the reader.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="channels" type="rm:AlarmChannelListParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> alarm channels to remove. </xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="removeAllAlarmChannels">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">remove all AlarmChannels currently associated with the
reader.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getAlarmChannel">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get the AlarmChannel with the specified name currently associated
with the reader.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="name" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> Alarm channel name </xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getAllAlarmChannels">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">get all AlarmChannels currently associated with the
reader.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
```

```
            <xsd:annotation>
               <xsd:documentation>For standard and vendor extensions</xsd:documentation>
            </xsd:annotation>
         </xsd:any>
      </xsd:choice>
   </xsd:complexType>
   <xsd:complexType name="ReadPointCommand">
      <xsd:annotation>
         <xsd:documentation xml:lang="en"> Read point object commands.</xsd:documentation>
      </xsd:annotation>
      <xsd:choice>
         <xsd:element name="getClassName">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">get the class name of the ReadPoint object.  In the current
specification, the only supported read point class is an "AntennaReadPoint".</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="getDescription">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">get the description of the ReadPoint.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="setDescription">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">set the  ReadPoint description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="description" type="xsd:string">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en"> description to set.</xsd:documentation>
                     </xsd:annotation>
                  </xsd:element>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="getAdminStatus">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">get the current ReadPoint administrative status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="setAdminStatus">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">set the current ReadPoint administrative status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="status" type="rm:AdministrativeStatusParamType">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en"> administrative status to set. </xsd:documentation>
                     </xsd:annotation>
                  </xsd:element>
               </xsd:sequence>
```

```
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getOperStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">get the ReadPoint's current operational status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getOperStatusAlarmControl">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">get the ReadPoint's operational status alarm control.
</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:any namespace="##any" processContents="lax">
            <xsd:annotation>
              <xsd:documentation>For standard and vendor extensions</xsd:documentation>
            </xsd:annotation>
          </xsd:any>
        </xsd:choice>
      </xsd:complexType>
      <xsd:complexType name="SourceCommand">
        <xsd:choice>
          <xsd:annotation>
            <xsd:documentation xml:lang="en">Source object commands. </xsd:documentation>
          </xsd:annotation>
          <xsd:element name="getUnknownToGlimpsedCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for the number of times a transition from state
Unknown to state Glimpsed have been detected for the particular source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getGlimpsedToUnknownCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for the number of times a transition from state
Glimpsed to state Unknown have been detected for the particular source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getGlimpsedToObservedCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for the number of times a transition from state
Glimpsed to state Observed have been detected for the particular source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getObservedToLostCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">query the reader for the number of times a transition from state
Observed to state Lost have been detected for the particular source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getLostToGlimpsedCount">
```

```
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the number of times a transition from state Lost to
state Glimpsed have been detected for the particular source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="getLostToUnknownCount">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the number of times a transition from state Lost to
state Unknown have been detected for the particular source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="getOperStatus">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the operational status of this particular
Source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="getAdminStatus">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the administrative status of this particular
Source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="setAdminStatus">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">set the administrative status of this particular
Source.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="administrativeStatus" type="rm:AdministrativeStatusParamType">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en">administrative status to set.</xsd:documentation>
                     </xsd:annotation>
                  </xsd:element>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="getOperStatusAlarmControl">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">get this particular source's operational status alarm control.
</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:any namespace="##any" processContents="lax">
            <xsd:annotation>
               <xsd:documentation>For standard and vendor extensions</xsd:documentation>
            </xsd:annotation>
         </xsd:any>
      </xsd:choice>
   </xsd:complexType>
```

```
   <xsd:complexType name="TriggerCommand">
      <xsd:choice>
         <xsd:annotation>
            <xsd:documentation xml:lang="en"> Trigger object commands. </xsd:documentation>
         </xsd:annotation>
         <xsd:element name="getFireCount">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the number of times a particular trigger has
fired.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:any namespace="##any" processContents="lax">
            <xsd:annotation>
               <xsd:documentation>For standard and vendor extensions</xsd:documentation>
            </xsd:annotation>
         </xsd:any>
      </xsd:choice>
   </xsd:complexType>
   <xsd:complexType name="TTOperationalStatusAlarmControlCommand">
      <xsd:choice>
         <xsd:annotation>
            <xsd:documentation xml:lang="en"> Transition triggered alarm control object commands.
</xsd:documentation>
         </xsd:annotation>
         <xsd:element name="getTriggerFromState">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the current value of the
TTOperationalStatusAlarmControlâ™s TriggerFromState attribute.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="setTriggerFromState">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">set the current value of the TTOperatonalStatusAlarmControlâ™s
TriggerFromState attribute.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="triggerFromState" type="rm:OperationalStatusParamType">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en">trigger from state to set</xsd:documentation>
                     </xsd:annotation>
                  </xsd:element>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="getTriggerToState">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">query the reader for the current value of the
TTOperationalStatusAlarmControlâ™s TriggerToState attribute.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="setTriggerToState">
            <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">set the current value of the TTOperatonalStatusAlarmControlâ™s
TriggerToState attribute.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="triggerToState" type="rm:OperationalStatusParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">trigger to state to set</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
</xsd:schema>
```

10.3.2 Reply XML Message Encoding (Reader-To-Host)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="urn:epcglobal:rm:xsd:1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0" xmlns:rm="urn:epcglobal:rm:xsd:1"
xmlns:epcglobal="urn:epcglobal:xsd:1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:annotation>
        <xsd:documentation xml:lang="en">
            <epcglobal:copyright>Copyright&#169;2005-2006 Epcglobal Inc., All Rights Reserved.</epcglobal:copyright>
            <epcglobal:disclaimer>EPCglobal Inc., its members, officers, directors, employees, or
                agents shall not be liable for any injury, loss, damages, financial or otherwise,
                arising from, related to, or caused by the use of this document. The use of said
                document shall constitute your express consent to the foregoing exculpation.</epcglobal:disclaimer>
            <epcglobal:specification>Reader Management (RM) version 1.0</epcglobal:specification>
        </xsd:documentation>
    </xsd:annotation>
    <xsd:include schemaLocation="RmCommon.xsd"/>
    <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="./EpcGlobal.xsd"/>
    <!-- Reader Management Reply -->
    <xsd:element name="reply">
        <xsd:annotation>
            <xsd:documentation xml:lang="en"> This element defines a reader management reply. Just like commands,
replies are grouped by the object they belong to. To specify the object a reply relates to, the name of the object is
used. All the replies contain a single return value inside the reply. return value is enclosed by the element
 "returnValue". In case of a single return value it is directly added to this element. In case of an array of return
values they are enclosed within element
 "list" with each value in turn enclosed in element "value". Each reply is wrapped in
 its object type. </xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:complexContent>
                <xsd:extension base="epcglobal:Document">
```

```
            <xsd:sequence>
              <xsd:element name="id" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">The id of the command for which  this is the
reply.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
              <xsd:element name="resultCode" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> the error code for the command - 0  for
success.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
              <xsd:choice>
                <!-- Error information -->
                <xsd:element name="error" type="rm:ErrorType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">Information about the error.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
                <!-- Reader Management (Target) Object Types -->
                <!-- elements inside are ordered in alphabetical order -->
                <xsd:element name="alarmChannel" type="rm:AlarmChannelReply">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">An Alarm channel</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
                <xsd:element name="alarmControl" type="rm:AlarmControlReply">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">AlarmControl is the base class for all of classes
within the RMP object model responsible for controlling the generation of alarm messages within an EPCglobal
compliant Reader.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
                <xsd:element name="antennaReadPoint" type="rm:AntennaReadPointReply">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">Extension of Read point object for
antenna.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
                <xsd:element name="edgeTriggeredAlarmControl" type="rm:EdgeTriggeredAlarmControlReply">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">This class extends AlarmControl to control alarms
generated when a monitored, integer-valued, model element first crosses a threshold value (the
AlarmThreshold).</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
                <xsd:element name="iOPort" type="rm:IOPortReply">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The hardware element that provides external input
and output lines to connect to other components outside the reader device.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
                <xsd:element name="notificationChannel" type="rm:NotificationChannelReply">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The notification channel
```

```
                        carries messages issued asynchronously by the Reader to the Host.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
                <xsd:element name="readerDevice" type="rm:ReaderDeviceReply">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">A Reader</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
                <xsd:element name="readPoint" type="rm:ReadPointReply">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">A ReadPoint</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
                <xsd:element name="source" type="rm:SourceReply">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">A Read source</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
                <xsd:element name="trigger" type="rm:TriggerReply">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">A Read/Notify Trigger</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
                <xsd:element name="tTOperationalStatusAlarmControl"
type="rm:TTOperationalStatusAlarmControlReply">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">This class extends AlarmControl to control alarms
generated when a monitored model element of type OperationalStatus transitions to a new
value.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
                <xsd:any namespace="##any" processContents="lax">
                    <xsd:annotation>
                        <xsd:documentation>For standard and vendor extensions</xsd:documentation>
                    </xsd:annotation>
                </xsd:any>
            </xsd:choice>
        </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
 </xsd:complexType>
</xsd:element>
<xsd:complexType name="ErrorType">
 <xsd:annotation>
  <xsd:documentation xml:lang="en">Information about the error.</xsd:documentation>
 </xsd:annotation>
 <xsd:sequence>
  <xsd:element name="name" type="xsd:string" minOccurs="0">
   <xsd:annotation>
    <xsd:documentation xml:lang="en">Name of the error.</xsd:documentation>
   </xsd:annotation>
  </xsd:element>
  <xsd:element name="cause" type="xsd:string" minOccurs="0">
   <xsd:annotation>
    <xsd:documentation xml:lang="en">Cause of the error.</xsd:documentation>
   </xsd:annotation>
```

```
        </xsd:element>
        <xsd:element name="description" type="xsd:string" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">Description of the error.</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:sequence>
    </xsd:complexType>
    <!-- Reader Management (Target) Object Types -->
    <!-- types are ordered in alphabetical order -->
    <!-- inside each type replies are listed in the order in which it is defined in the RM specification. -->
    <xsd:complexType name="AlarmChannelReply">
      <xsd:choice>
        <xsd:annotation>
          <xsd:documentation xml:lang="en"> Alarm channel object replies.</xsd:documentation>
        </xsd:annotation>
        <xsd:element name="create">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for create an Alarm channel.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:AlarmChannelParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en"> created alarm channel.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getName">
          <xsd:annotation>
            <xsd:documentation xml:lang="en"> reply for get Name.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">This AlarmChannel's name</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getAddress">
          <xsd:annotation>
            <xsd:documentation xml:lang="en"> reply for get Address.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:AddressParamType">
```

```
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">reporting address for this AlarmChannel.
</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="setAddress">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for set Address</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:choice>
        </xsd:complexType>
        <xsd:complexType name="AlarmControlReply">
          <xsd:choice>
            <xsd:annotation>
              <xsd:documentation xml:lang="en"> Alarm control object replies. </xsd:documentation>
            </xsd:annotation>
            <xsd:element name="getName">
              <xsd:annotation>
                <xsd:documentation xml:lang="en"> reply for get Name.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="xsd:string">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">This AlarmControl's name</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getEnabled">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for  get enabled.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="xsd:boolean">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">Whether alarm control is enabled.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="setEnabled">
              <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">reply for set enabled.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getLevel">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get Level.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:AlarmLevelParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">current value of the AlarmControl's
Level.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="setLevel">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for set Level.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getSuppressInterval">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get SuppressInterval.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">current value of the AlarmControl's SuppressInterval in
seconds.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="setSuppressInterval">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for set SuppressInterval.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
    <xsd:complexType name="AntennaReadPointReply">
      <xsd:choice>
        <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en"> Antenna read point object replies </xsd:documentation>
          </xsd:annotation>
          <xsd:element name="getIdentificationCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get identification count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The count of the successful tag identifiers read at this
AntennaReadPoint.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFailedIdentificationCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get failed identification count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The count of the failed attempts at reading the identifier for a
tag at this antenna ReadPoint.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getMemReadCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get mem read count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The count of the successful tag memory reads at this antenna
ReadPoint.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFailedMemReadCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get failed mem read count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
```

```
                        <xsd:documentation xml:lang="en">The count of the failed  tag memory reads at this antenna
ReadPoint.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getFailedMemReadAlarmControl">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get failed mem read alarm control.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AlarmControlParamType">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">An alarm control for monitoring tag memory read
failures.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getWriteCount">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get write count.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">The count of the successful writes at this
AntennaReadPoint.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getFailedWriteCount">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get failed write count.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">The count of the failed writes at this
AntennaReadPoint.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getFailedWriteAlarmControl">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get failed write alarm control.</xsd:documentation>
        </xsd:annotation>
```

```
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AlarmControlParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">An alarm control for monitoring the number of failed
writes.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getKillCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get kill count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The count of the successful tag kills at this antenna
ReadPoint.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFailedKillCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get failed kill count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The count of the failed tag kills at this antenna
readpoint.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFailedKillAlarmControl">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get failed kill alarm control.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AlarmControlParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">An alarm control for monitoring tag kill
failures.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
```

```
        <xsd:element name="getEraseCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get erase count.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">The count of the successful tag erasures at this antenna
ReadPoint.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getFailedEraseCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get failed erase count.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">The count of the failed tag erasures at this antenna
readpoint.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getFailedEraseAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get failed erase alarm control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:AlarmControlParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">An alarm control for monitoring tag erasure
failures.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getLockCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get lock count.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">The count of the successful tag locks at this antenna
ReadPoint.</xsd:documentation>
                </xsd:annotation>
```

```
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFailedLockCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get failed lock count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The count of the failed tag locks at this antenna
readpoint.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFailedLockAlarmControl">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get failed lock alarm control.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AlarmControlParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">An alarm control for monitoring tag lock
failures.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getTimeEnergized">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get time energized.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The number of milliseconds the AntennaReadPoint has been
energized attempting communication with tags.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getPowerLevel">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get power level.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
```

```
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">current transmit power level of the antenna
ReadPoint.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getNoiseLevel">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get noise level.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="xsd:int">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">current noise level observed at the antenna
readpoint.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:choice>
        </xsd:complexType>
        <xsd:complexType name="EdgeTriggeredAlarmControlReply">
          <xsd:choice>
            <xsd:annotation>
              <xsd:documentation xml:lang="en"> Edge triggered alarm control object replies. </xsd:documentation>
            </xsd:annotation>
            <xsd:element name="getAlarmThreshold">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get Alarm threshold.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="xsd:int">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">current value of the EdgeTriggeredAlarmControl's
AlarmThreshold .</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="setAlarmThreshold">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for  set alarm threshold.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
```

```
        <xsd:element name="getRearmThreshold">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get RearmThreshold.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">current value of the EdgeTriggeredAlarmControl's
RearmThreshold.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="setRearmThreshold">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for set RearmThreshold.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getDirection">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get direction.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:EdgeTriggeredAlarmDirectionParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">current value of the EdgeTriggeredAlarmControl's
Direction.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="setDirection">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for set direction.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getStatus">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get status.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:EdgeTriggeredAlarmStatusParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">current value of the EdgeTriggeredAlarmControl's
Status.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
```

```
      </xsd:element>
      <xsd:any namespace="##any" processContents="lax">
        <xsd:annotation>
          <xsd:documentation>For standard and vendor extensions</xsd:documentation>
        </xsd:annotation>
      </xsd:any>
    </xsd:choice>
  </xsd:complexType>
  <xsd:complexType name="IOPortReply">
    <xsd:choice>
      <xsd:annotation>
        <xsd:documentation xml:lang="en">IO port object replies. </xsd:documentation>
      </xsd:annotation>
      <xsd:element name="getName">
        <xsd:annotation>
          <xsd:documentation xml:lang="en"> reply for get Name.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="returnValue" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">This IO port's name</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="getDescription">
        <xsd:annotation>
          <xsd:documentation xml:lang="en">reply for get description.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="returnValue" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">a textual description of the IO-port.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="setDescription">
        <xsd:annotation>
          <xsd:documentation xml:lang="en">reply for set description.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType/>
      </xsd:element>
      <xsd:element name="getOperStatus">
        <xsd:annotation>
          <xsd:documentation xml:lang="en">reply for get operational status.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">the operational status of the IO-port.</xsd:documentation>
```

```
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getAdminStatus">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get administrative status.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="rm:AdministrativeStatusParamType">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">the administrative status of the IO-port.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="setAdminStatus">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for set admin status.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="getOperStatusAlarmControl">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get operational status alarm control.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="rm:TTOperationalStatusAlarmControlParamType">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">An alarm control for monitoring the operational status of the
IOPort.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:choice>
        </xsd:complexType>
        <xsd:complexType name="NotificationChannelReply">
          <xsd:choice>
            <xsd:annotation>
              <xsd:documentation xml:lang="en">Notification channel object replies</xsd:documentation>
            </xsd:annotation>
            <xsd:element name="getLastNotificationAttempt">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get last notification attempt.</xsd:documentation>
              </xsd:annotation>
```

```
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the timestamp (TimeTicks) when the last attempt was made to
send a notification to the given address.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getLastSuccessfulNotification">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get last successful notifcation.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the timestamp (TimeTicks) when the last successful
notification was send to the given address.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getOperStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get operational status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the operational status of the notification
channel.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="setAdminStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for set administrative status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getAdminStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get administrative status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AdministrativeStatusParamType">
                  <xsd:annotation>
```

```
                        <xsd:documentation xml:lang="en">the administrative status of the notification
channel.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="getOperStatusAlarmControl">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get operational status alarm control.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="rm:TTOperationalStatusAlarmControlParamType">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">An alarm control for monitoring the operational status of the
notification channel.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:choice>
        </xsd:complexType>
        <xsd:complexType name="ReaderDeviceReply">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">Reader Device object replies. </xsd:documentation>
          </xsd:annotation>
          <xsd:choice>
            <xsd:element name="getDescription">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for get description.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType>
                <xsd:sequence>
                  <xsd:element name="returnValue" type="xsd:string">
                    <xsd:annotation>
                      <xsd:documentation xml:lang="en">user defined description of the reader.</xsd:documentation>
                    </xsd:annotation>
                  </xsd:element>
                </xsd:sequence>
              </xsd:complexType>
            </xsd:element>
            <xsd:element name="setDescription">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">reply for set description.</xsd:documentation>
              </xsd:annotation>
              <xsd:complexType/>
            </xsd:element>
            <xsd:element name="getLocationDescription">
              <xsd:annotation>
```

```
5753                <xsd:documentation xml:lang="en">reply for get location description.</xsd:documentation>
5754              </xsd:annotation>
5755              <xsd:complexType>
5756                <xsd:sequence>
5757                  <xsd:element name="returnValue" type="xsd:string">
5758                    <xsd:annotation>
5759                      <xsd:documentation xml:lang="en">user defined location description of the reader
5760  device.</xsd:documentation>
5761                    </xsd:annotation>
5762                  </xsd:element>
5763                </xsd:sequence>
5764              </xsd:complexType>
5765            </xsd:element>
5766            <xsd:element name="setLocationDescription">
5767              <xsd:annotation>
5768                <xsd:documentation xml:lang="en">reply for set location description.</xsd:documentation>
5769              </xsd:annotation>
5770              <xsd:complexType/>
5771            </xsd:element>
5772            <xsd:element name="getContact">
5773              <xsd:annotation>
5774                <xsd:documentation xml:lang="en">reply for get contact.</xsd:documentation>
5775              </xsd:annotation>
5776              <xsd:complexType>
5777                <xsd:sequence>
5778                  <xsd:element name="returnValue" type="xsd:string">
5779                    <xsd:annotation>
5780                      <xsd:documentation xml:lang="en">user defined contact description.</xsd:documentation>
5781                    </xsd:annotation>
5782                  </xsd:element>
5783                </xsd:sequence>
5784              </xsd:complexType>
5785            </xsd:element>
5786            <xsd:element name="setContact">
5787              <xsd:annotation>
5788                <xsd:documentation xml:lang="en">reply for set contact.</xsd:documentation>
5789              </xsd:annotation>
5790              <xsd:complexType/>
5791            </xsd:element>
5792            <xsd:element name="getSerialNumber">
5793              <xsd:annotation>
5794                <xsd:documentation xml:lang="en">reply for get serial number.</xsd:documentation>
5795              </xsd:annotation>
5796              <xsd:complexType>
5797                <xsd:sequence>
5798                  <xsd:element name="returnValue" type="xsd:string">
5799                    <xsd:annotation>
5800                      <xsd:documentation xml:lang="en">the serial number of the reader device.</xsd:documentation>
5801                    </xsd:annotation>
5802                  </xsd:element>
5803                </xsd:sequence>
5804              </xsd:complexType>
5805            </xsd:element>
5806            <xsd:element name="getOperStatus">
5807              <xsd:annotation>
5808                <xsd:documentation xml:lang="en">reply for get operational status.</xsd:documentation>
```

```
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the operational status of the reader.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getOperStatusAlarmControl">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get operational status alarm control.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:TTOperationalStatusAlarmControlParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">An alarm control for monitoring the operational status of the
reader.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFreeMemory">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get free memory.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">available free memory.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getFreeMemoryAlarmControl">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get free memory alarm control.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:EdgeTriggeredAlarmControlParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">FreeMemoryAlarmControl.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getNTPServers">
            <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">reply for get NTP servers.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="list">
                      <xsd:complexType>
                        <xsd:sequence>
                          <xsd:element name="value" type="xsd:string"
minOccurs="0" maxOccurs="unbounded">
                            <xsd:annotation>
                              <xsd:documentation xml:lang="en">
NTP servers used by the device to synchronize its current UTC clock (TimeUTC). </xsd:documentation>
                            </xsd:annotation>
                          </xsd:element>
                        </xsd:sequence>
                      </xsd:complexType>
                    </xsd:element>
                  </xsd:sequence>
                </xsd:complexType>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getDHCPServer">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get DHCP server.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">the DHCP server currently used by the device for DHCP
requests.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getIOPort">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get IO port.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:IOPortParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">the named IO port object.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getAllIOPorts">
```

```
            <xsd:annotation>
               <xsd:documentation xml:lang="en">reply for get all IO ports.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="returnValue">
                     <xsd:complexType>
                        <xsd:sequence>
                                 <xsd:element name="list">
                                          <xsd:complexType>
                                                   <xsd:sequence>
                                                            <xsd:element name="value" type="rm:IOPortParamType"
minOccurs="0" maxOccurs="unbounded">
                                                                     <xsd:annotation>
                                                                              <xsd:documentation xml:lang="en">
IOPort objects. </xsd:documentation>
                                                                     </xsd:annotation>
                                                            </xsd:element>
                                                   </xsd:sequence>
                                          </xsd:complexType>
                                 </xsd:element>
                        </xsd:sequence>
                     </xsd:complexType>
                  </xsd:element>
               </xsd:sequence>
            </xsd:complexType>
         </xsd:element>
         <xsd:element name="resetStatistics">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">reply for reset statistics.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="removeAlarmChannels">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">reply for remove alarm channels.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="removeAllAlarmChannels">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">reply for remove all alarm channels.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
         </xsd:element>
         <xsd:element name="getAlarmChannel">
            <xsd:annotation>
               <xsd:documentation xml:lang="en">reply for get alarm channel.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
               <xsd:sequence>
                  <xsd:element name="returnValue" type="rm:AlarmChannelParamType">
                     <xsd:annotation>
                        <xsd:documentation xml:lang="en">the named AlarmChannel object.</xsd:documentation>
                     </xsd:annotation>
                  </xsd:element>
```

```
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getAllAlarmChannels">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get all alarm channels.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="list">
                      <xsd:complexType>
                        <xsd:sequence>
                          <xsd:element name="value"
type="rm:AlarmChannelParamType" minOccurs="0" maxOccurs="unbounded">
                            <xsd:annotation>
                              <xsd:documentation xml:lang="en">
Alarm channels. </xsd:documentation>
                            </xsd:annotation>
                          </xsd:element>
                        </xsd:sequence>
                      </xsd:complexType>
                    </xsd:element>
                  </xsd:sequence>
                </xsd:complexType>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
    <xsd:complexType name="ReadPointReply">
      <xsd:annotation>
        <xsd:documentation xml:lang="en"> Read point object replies.</xsd:documentation>
      </xsd:annotation>
      <xsd:choice>
        <xsd:element name="getClassName">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get class name.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:string">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">The class name for the ReadPoint.  The only supported return
value is  "AntennaReadPoint". </xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
```

```
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getDescription">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:string">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">The readpoint descriptive name.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="setDescription">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for set description.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getAdminStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get administrative status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AdministrativeStatusParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the administrative status of the read
point.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="setAdminStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for set administrative status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType/>
          </xsd:element>
          <xsd:element name="getOperStatus">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get operational status.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the operational status of the read point.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
```

```
        </xsd:element>
        <xsd:element name="getOperStatusAlarmControl">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get operational status alarm control.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:TTOperationalStatusAlarmControlParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">An alarm control for monitoring the operational status of the
read point.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
    <xsd:complexType name="SourceReply">
      <xsd:choice>
        <xsd:annotation>
          <xsd:documentation xml:lang="en">Source object replies. </xsd:documentation>
        </xsd:annotation>
        <xsd:element name="getUnknownToGlimpsedCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get unknown to glimpsed count.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">the number of times a transition from state Unknown to state
Glimpsed have been detected for the particular source.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="getGlimpsedToUnknownCount">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get glimpsed to unknown count.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="xsd:int">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">the number of times a transition from state Glimpsed to state
Unknown have been detected for the particular source.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
```

```
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getGlimpsedToObservedCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get Glimpsed to Observed count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the number of times a transition from state Glimpsed to state
Observed have been detected for the particular source.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getObservedToLostCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get Observed to Lost count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the number of times a transition from state Observed to state
Lost have been detected for the particular source.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getLostToGlimpsedCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get Lost to Glimpsed count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
                    <xsd:documentation xml:lang="en">the number of times a transition from state Lost to state
Observed have been detected for the particular source.</xsd:documentation>
                  </xsd:annotation>
                </xsd:element>
              </xsd:sequence>
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="getLostToUnknownCount">
            <xsd:annotation>
              <xsd:documentation xml:lang="en">reply for get Lost to Unknown count.</xsd:documentation>
            </xsd:annotation>
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="returnValue" type="xsd:int">
                  <xsd:annotation>
```

```
                        <xsd:documentation xml:lang="en">the number of times a transition from state Lost to state
Unknown have been detected for the particular source.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getOperStatus">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get operational status.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">the operational status of the source.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="getAdminStatus">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get administrative status.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="returnValue" type="rm:AdministrativeStatusParamType">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">the administrative status of the source.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="setAdminStatus">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for administrative status.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType/>
    </xsd:element>
    <xsd:element name="getOperStatusAlarmControl">
        <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get operational status alarm control.</xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="returnValue" type="rm:TTOperationalStatusAlarmControlParamType">
                    <xsd:annotation>
                        <xsd:documentation xml:lang="en">An alarm control for monitoring the operational status of
source.</xsd:documentation>
                    </xsd:annotation>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
```

```
6256          <xsd:any namespace="##any" processContents="lax">
6257            <xsd:annotation>
6258              <xsd:documentation>For standard and vendor extensions</xsd:documentation>
6259            </xsd:annotation>
6260          </xsd:any>
6261        </xsd:choice>
6262      </xsd:complexType>
6263      <xsd:complexType name="TriggerReply">
6264        <xsd:choice>
6265          <xsd:annotation>
6266            <xsd:documentation xml:lang="en"> Trigger object replies. </xsd:documentation>
6267          </xsd:annotation>
6268          <xsd:element name="getFireCount">
6269            <xsd:annotation>
6270              <xsd:documentation xml:lang="en">reply for get fire count.</xsd:documentation>
6271            </xsd:annotation>
6272            <xsd:complexType>
6273              <xsd:sequence>
6274                <xsd:element name="returnValue" type="xsd:int">
6275                  <xsd:annotation>
6276                    <xsd:documentation xml:lang="en">the number of times a particular trigger has
6277  fired.</xsd:documentation>
6278                  </xsd:annotation>
6279                </xsd:element>
6280              </xsd:sequence>
6281            </xsd:complexType>
6282          </xsd:element>
6283          <xsd:any namespace="##any" processContents="lax">
6284            <xsd:annotation>
6285              <xsd:documentation>For standard and vendor extensions</xsd:documentation>
6286            </xsd:annotation>
6287          </xsd:any>
6288        </xsd:choice>
6289      </xsd:complexType>
6290      <xsd:complexType name="TTOperationalStatusAlarmControlReply">
6291        <xsd:choice>
6292          <xsd:annotation>
6293            <xsd:documentation xml:lang="en"> Transition triggered alarm control object replies. </xsd:documentation>
6294          </xsd:annotation>
6295          <xsd:element name="getTriggerFromState">
6296            <xsd:annotation>
6297              <xsd:documentation xml:lang="en">reply for get trigger from state.</xsd:documentation>
6298            </xsd:annotation>
6299            <xsd:complexType>
6300              <xsd:sequence>
6301                <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
6302                  <xsd:annotation>
6303                    <xsd:documentation xml:lang="en">current value of the TTOperationalStatusAlarmControl's
6304  TriggerFromState.</xsd:documentation>
6305                  </xsd:annotation>
6306                </xsd:element>
6307              </xsd:sequence>
6308            </xsd:complexType>
6309          </xsd:element>
6310          <xsd:element name="setTriggerFromState">
6311            <xsd:annotation>
```

```
            <xsd:documentation xml:lang="en">reply for set trigger from state.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="getTriggerToState">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for get trigger to state.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="returnValue" type="rm:OperationalStatusParamType">
                <xsd:annotation>
                  <xsd:documentation xml:lang="en">current value of the TTOperationalStatusAlarmControl's
TriggerToState.</xsd:documentation>
                </xsd:annotation>
              </xsd:element>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
        <xsd:element name="setTriggerToState">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">reply for set trigger to state.</xsd:documentation>
          </xsd:annotation>
          <xsd:complexType/>
        </xsd:element>
        <xsd:any namespace="##any" processContents="lax">
          <xsd:annotation>
            <xsd:documentation>For standard and vendor extensions</xsd:documentation>
          </xsd:annotation>
        </xsd:any>
      </xsd:choice>
    </xsd:complexType>
</xsd:schema>
```

10.3.3 Alarm Notification XML Message Encoding (Reader-To-Host)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="urn:epcglobal:rm:xsd:1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:epcglobal="urn:epcglobal:xsd:1" xmlns:rm="urn:epcglobal:rm:xsd:1">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      <epcglobal:copyright>Copyright&#169;2005-2006 Epcglobal Inc., All Rights Reserved.</epcglobal:copyright>
      <epcglobal:disclaimer>EPCglobal Inc., its members, officers, directors, employees, or
        agents shall not be liable for any injury, loss, damages, financial or otherwise,
        arising from, related to, or caused by the use of this document. The use of said
        document shall constitute your express consent to the foregoing exculpation.</epcglobal:disclaimer>
      <epcglobal:specification>Reader Management (RM) version 1.0</epcglobal:specification>
    </xsd:documentation>
  </xsd:annotation>
  <xsd:include schemaLocation="RmCommon.xsd"/>
  <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="./EpcGlobal.xsd"/>
  <!-- Reader Management Alarm -->
```

```
<xsd:element name="alarm" type="rm:AlarmType">
  <xsd:annotation>
    <xsd:documentation xml:lang="en"> This element defines a reader management alarm.</xsd:documentation>
  </xsd:annotation>
</xsd:element>
<xsd:complexType name="ReaderType">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">Reader Information</xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="readerEPC" type="epcglobal:EPC" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Reader EPC</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="readerName" type="xsd:string" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Reader name</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="readerHandle" type="xsd:int" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Reader handle</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="readerRole" type="xsd:string" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Reader role</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
    <xsd:element name="readerTime" type="rm:readerTime" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Reader time</xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
<!-- Alarms -->
<!-- Alarm types (including their base types) are ordered in alphabetical order -->
<xsd:complexType name="AlarmType" abstract="true">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">Alarm is the base class for all of classes within the RMP object model that
define the contents of alarm messages</xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="epcglobal:Document">
      <xsd:sequence>
        <xsd:element name="id" type="xsd:string">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">id of this alarm.</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="reader" type="rm:ReaderType" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">Reader sending the alarm</xsd:documentation>
          </xsd:annotation>
```

```
            </xsd:element>
            <xsd:element name="name" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Type of Alarm message</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="level" type="rm:AlarmLevelParamType">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Severity level assigned to the alarm</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="suppressCount" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Number of times the generation of this Alarm has been
suppressed</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:any namespace="##any" processContents="lax">
              <xsd:annotation>
                <xsd:documentation>For standard and vendor extensions</xsd:documentation>
              </xsd:annotation>
            </xsd:any>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="FailedEraseAlarmType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">This alarm's receipt signals a tag erase failure.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:AlarmType">
          <xsd:sequence>
            <xsd:element name="readPointName" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Read point where the erase failure
occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="failedEraseCount" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Value of AntennaReadPoint.FailedEraseCount element after the
erase failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="noiseLevel" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Value of the AntennaReadPoint.NoiseLevel element when the
erase failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="FailedKillAlarmType">
```

```
      <xsd:annotation>
        <xsd:documentation xml:lang="en">This alarm's receipt signals a tag kill failure.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:AlarmType">
          <xsd:sequence>
            <xsd:element name="readPointName" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Read point where the kill failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="failedKillCount" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Value of AntennaReadPoint.FailedKillCount element after the kill
failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="noiseLevel" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Value of the AntennaReadPoint.NoiseLevel element when the kill
failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="FailedLockAlarmType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">This alarm's receipt signals a tag lock failure.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:AlarmType">
          <xsd:sequence>
            <xsd:element name="readPointName" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Read point where the lock failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="failedLockCount" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Value of AntennaReadPoint.FailedLockCount element after the
lock failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="noiseLevel" type="xsd:int">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">Value of the AntennaReadPoint.NoiseLevel element when the
lock failure occurred.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="FailedMemReadAlarmType">
```

```
      <xsd:annotation>
         <xsd:documentation xml:lang="en">This alarm's receipt signals a tag user-memory read
failure.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
         <xsd:extension base="rm:AlarmType">
            <xsd:sequence>
               <xsd:element name="readPointName" type="xsd:string">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Read point where the memory read failure
occurred.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="failedMemReadCount" type="xsd:int">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Value of AntennaReadPoint.FailedMemReadCount element after
the memory read failure occurred.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="noiseLevel" type="xsd:int">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Value of the AntennaReadPoint.NoiseLevel element when the
memory read failure occurred.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="FailedWriteAlarmType">
      <xsd:annotation>
         <xsd:documentation xml:lang="en">This alarm's receipt signals a tag write failure.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
         <xsd:extension base="rm:AlarmType">
            <xsd:sequence>
               <xsd:element name="readPointName" type="xsd:string">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Read point where the write failure
occurred.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="failedWriteCount" type="xsd:int">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Value of AntennaReadPoint.FailedWriteCount element after the
write failure occurred.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
               <xsd:element name="noiseLevel" type="xsd:int">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Value of the AntennaReadPoint.NoiseLevel element when the
write failure occurred.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
```

```
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="FreeMemoryAlarmType">
      <xsd:annotation>
         <xsd:documentation xml:lang="en">This alarm's receipt signals the movement of a reader device's free
memory (represented in the abstract model by ReaderDevice.FreeMemory) below a specified threshold
value.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
         <xsd:extension base="rm:AlarmType">
            <xsd:sequence>
               <xsd:element name="freeMemory" type="xsd:int">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">Value of ReaderDevice.FreeMemory when the alarm was
triggered.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="IOPortOperStatusAlarmType">
      <xsd:annotation>
         <xsd:documentation xml:lang="en">This alarm's receipt signals a change in the operational status of a
Reader's IO Port. </xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
         <xsd:extension base="rm:TTOperStatusAlarmType">
            <xsd:sequence>
               <xsd:element name="iOPortName" type="xsd:string">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en">IO port name.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="NotificationChannelOperStatusAlarmType">
      <xsd:annotation>
         <xsd:documentation xml:lang="en">This alarm's receipt signals a change in the operational status of one of a
Reader's notification channels.</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
         <xsd:extension base="rm:TTOperStatusAlarmType">
            <xsd:sequence>
               <xsd:element name="notificationChannelName" type="xsd:string">
                  <xsd:annotation>
                     <xsd:documentation xml:lang="en"> Notification channel name.</xsd:documentation>
                  </xsd:annotation>
               </xsd:element>
            </xsd:sequence>
         </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
   <xsd:complexType name="ReaderDeviceOperStatusAlarmType">
```

```
      <xsd:annotation>
        <xsd:documentation xml:lang="en">This alarm's receipt signals a change in the operational status of a Reader.
</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:TTOperStatusAlarmType"/>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="ReadPointOperStatusAlarmType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">This alarm's receipt signals a change in the operational status of one of a
Reader's Read Points. </xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:TTOperStatusAlarmType">
          <xsd:sequence>
            <xsd:element name="readPointName" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en"> Read point name.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="SourceOperStatusAlarmType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">This alarm's receipt signals a change in the operational status of a logical
source of EPC data on a Reader. </xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:TTOperStatusAlarmType">
          <xsd:sequence>
            <xsd:element name="sourceName" type="xsd:string">
              <xsd:annotation>
                <xsd:documentation xml:lang="en"> Source name.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
          </xsd:sequence>
        </xsd:extension>
      </xsd:complexContent>
    </xsd:complexType>
    <xsd:complexType name="TTOperStatusAlarmType" abstract="true">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Base class for all transition triggered Operational Status Alarms.
</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="rm:AlarmType">
          <xsd:sequence>
            <xsd:element name="fromState" type="rm:OperationalStatusParamType">
              <xsd:annotation>
                <xsd:documentation xml:lang="en">From state of the transition.</xsd:documentation>
              </xsd:annotation>
            </xsd:element>
            <xsd:element name="toState" type="rm:OperationalStatusParamType">
```

```
                <xsd:annotation>
                   <xsd:documentation xml:lang="en">To state of the transition.</xsd:documentation>
                </xsd:annotation>
             </xsd:element>
          </xsd:sequence>
       </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:schema>
```

10.3.4 Common Data Formats

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="urn:epcglobal:rm:xsd:1" elementFormDefault="unqualified"
attributeFormDefault="unqualified" version="1.0" xmlns:rm="urn:epcglobal:rm:xsd:1"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
    </xsd:documentation>
  </xsd:annotation>
  <!-- Reader Management types -->
  <!-- types are ordered in alphabetical order -->
  <xsd:annotation>
    <xsd:documentation xml:lang="en"> Reader management common types. All types end with the
      suffix "Type". All the command parameter or reply return value reference types are
      suffixed with "ParamType". All named objects are referenced using "name". For example,
      NotificationChannelParamType contains a reference (its name) instead of actual notification channel
      itself.</xsd:documentation>
  </xsd:annotation>
  <xsd:simpleType name="AddressParamType">
    <xsd:annotation>
      <xsd:documentation xml:lang="en">The address for a channel</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:anyURI"/>
  </xsd:simpleType>
  <xsd:simpleType name="AdministrativeStatusParamType">
    <xsd:annotation>
      <xsd:documentation xml:lang="en"> administrative status</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="UP"/>
      <xsd:enumeration value="DOWN"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:complexType name="AlarmChannelListParamType">
    <xsd:annotation>
      <xsd:documentation> A list of alarm channels</xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="list">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="value" type="rm:AlarmChannelParamType" maxOccurs="unbounded"/>
```

```
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:simpleType name="AlarmChannelParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">The name of a alarm channel.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="AlarmControlParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">The name of an alarm control.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="AlarmLevelParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en"> Alarm level</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="EMERGENCY"/>
        <xsd:enumeration value="ALERT"/>
        <xsd:enumeration value="CRITICAL"/>
        <xsd:enumeration value="ERROR"/>
        <xsd:enumeration value="WARNING"/>
        <xsd:enumeration value="NOTICE"/>
        <xsd:enumeration value="INFORMATIONAL"/>
        <xsd:enumeration value="DEBUG"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EdgeTriggeredAlarmControlParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">The name of an edge triggered alarm control.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="rm:AlarmControlParamType"/>
    </xsd:simpleType>
    <xsd:simpleType name="EdgeTriggeredAlarmDirectionParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en"> Edge triggered alarm direction.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="RISING"/>
        <xsd:enumeration value="FALLING"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EdgeTriggeredAlarmStatusParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en"> Edge triggered alarm status.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="ARMED"/>
        <xsd:enumeration value="FIRED"/>
      </xsd:restriction>
    </xsd:simpleType>
```

```
    <xsd:simpleType name="IOPortParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">The name of an IO port.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="OperationalStatusParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en"> operational status</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="UNKNOWN"/>
        <xsd:enumeration value="UP"/>
        <xsd:enumeration value="DOWN"/>
        <xsd:enumeration value="OTHER"/>
        <xsd:enumeration value="ANY"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="readerTime">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">Time format</xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="readerNowTick" type="xsd:string" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation xml:lang="en">time in ticks</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="readerNowUTC" type="xsd:dateTime" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation xml:lang="en"> UTC time</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:simpleType name="TTOperationalStatusAlarmControlParamType">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">The name of a transition triggered status alarm
control.</xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="rm:AlarmControlParamType"/>
    </xsd:simpleType>
</xsd:schema>
```

10.3.5 EPCglobal Standard Header

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:epcglobal="urn:epcglobal:xsd:1" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:epcglobal:xsd:1" elementFormDefault="unqualified" attributeFormDefault="unqualified"
version="1.0">
    <xsd:annotation>
      <xsd:documentation>
        <epcglobal:copyright>Copyright&amp;#169;2005-2006 Epcglobal Inc., All Rights Reserved.</epcglobal:copyright>
        <epcglobal:disclaimer>EPCglobal Inc., its members, officers, directors, employees, or agents shall not be liable
for any injury, loss, damages, financial or otherwise, arising from, related to, or caused by the use of this document.
The use of said document shall constitute your express consent to the foregoing exculpation.</epcglobal:disclaimer>
```

```
        <epcglobal:specification>EPCglobal common components Version 1.0</epcglobal:specification>
      </xsd:documentation>
    </xsd:annotation>
    <xsd:complexType name="Document" abstract="true">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">
    EPCglobal document properties for all messages.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:attribute name="schemaVersion" type="xsd:decimal" use="required">
        <xsd:annotation>
          <xsd:documentation xml:lang="en">
      The version of the schema corresponding to which the instance conforms.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
      <xsd:attribute name="creationDate" type="xsd:dateTime" use="required">
        <xsd:annotation>
          <xsd:documentation xml:lang="en">
      The date the message was created. Used for auditing and logging.
            </xsd:documentation>
        </xsd:annotation>
      </xsd:attribute>
    </xsd:complexType>
    <xsd:complexType name="EPC">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">
    EPC represents the Electronic Product Code.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:simpleContent>
        <xsd:extension base="xsd:string"/>
      </xsd:simpleContent>
    </xsd:complexType>
</xsd:schema>

      </xsd:attribute>
    </xsd:complexType>
    <xsd:complexType name="EPC">
      <xsd:annotation>
        <xsd:documentation xml:lang="en">
    EPC represents the Electronic Product Code.
          </xsd:documentation>
      </xsd:annotation>
      <xsd:simpleContent>
        <xsd:extension base="xsd:string"/>
      </xsd:simpleContent>
    </xsd:complexType>
</xsd:schema>
```

10.4 SNMP MIB

This specification defines two MIB modules in the EPCglobal enterprise tree that expose the EPCglobal Reader Object Model across an SNMP protocol interface.

The first of these MIB modules, EPCGLOBAL-SMI-MIB, is the EPCglobal central registration module. It contains the specification of the private enterprise number (22695) the Internet Assigned Numbers Authority (IANA) assigned to EPCglobal. In addition, it defines the top-level organization of the EPCglobal private enterprise namespace.

The second MIB module, EPCGLOBAL-READER-MIB, contains MIB definitions corresponding to objects within the Reader Object Model. This MIB defines model objects that are particular to the health monitoring of RFID Readers.

When there is a clear mapping of Reader Object Model elements to standard MIB-II object definitions, we rely on the IETF-standardized MIB modules to expose these model elements across SNMP, rather then duplicating these object definitions within EPCglobal MIB modules. Below is a table displaying the mapping of selected Reader Object Model elements to object definitions within IETF MIB modules (in particular, SNMPv2-MIB defined in RFC3418/STD 62 and SNMP-TARGET-MIB, RFC3413/STD62).

Reader Object Model element	IETF-Defined MIB Module	MIB OID
ReaderDevice.Manufacturer ReaderDevice.Model ReaderDevice.ManufacturerDescription	SNMPv2-MIB	sysDescr
ReaderDevice.LocationDescription	SNMPv2-MIB	location
ReaderDevice.Contact	SNMPv2-MIB	sysContact
ReaderDevice.TimeTicks	SNMPv2-MIB	sysUpTime
ReaderDevice.AlarmChannels	SNMP-TARGET-MIB	snmpTargetObjects
AlarmChannel.Name	SNMP-TARGET-MIB	snmpTargetAddrName
AlarmChannel.Address	SNMP-TARGET-MIB	snmpTargetAddrTDomain snmpTargetAddrTAddress
Alarm.TimeTicks	Protocol Operations for SNMP (RFC3416)	sysUpTime

In addition to compliance with the EPC Global Reader Management MIB, a compliant system that supports the SNMP binding SHALL implement support for the following IETF-standardized MIB groups:

- the MIB-II System Group defined in the SNMPv2-MIB module, defined in RFC 3418,
- the MIB-II IP Group, defined in the IP-MIB module, RFC 2011,
- the MIB-II Interfaces Group, defined in IF-MIB module, RFC 2863

These groups define the basic System Identification operations, Internet Protocol and Network Interfaces monitoring objects and operations. In order to avoid unnecessary duplication and to harmonize reader management, as a networked device, with that of other SNMP-manageable devices, support of these MIB-II groups is chosen in place of defining these objects within in the EPCglobal RFID Reader Management MIB.

Protocol support for the management of Alarm Channels is optional; however, if a reader vendor chooses to support SNMP-based management of Alarm Channels, the vendor SHALL do so by implementing the SNMP-TARGET-MIB module, defined in RFC 3413.

Within the EPCglobal MIB structure, elements referred to as "Alarms" in the Abstract Model are referred to as "Notifications". This is because the term "Alarm" in SNMP is used as a term of art (for example, in RFC 3877 and RFC 3878) and could lead to confusion within existing SNMP installations.

Alarm TimeTicks are included in the SNMP Notification data structure mandated by RFC 3416, therefore not explicitly included in the MIB.

Unless otherwise specified within the MIB, vendors MAY implement index persistence for any of the EPCglobal defined MIB structures. This statement only applies to the EPCglobal enterprise space.

10.4.1 Vendor Extension Details

Vendors wishing to provide vendor-specific extensions in SNMP MAY provide support for MIB modules outside the EPCglobal MIB space. For example, if a vendor supports vendor-specific MIB on their devices, they MAY also support these MIB modules on EPCglobal compliant readers. Additionally, a EPCglobal compliant readers MAY provide support for any IETF-standard MIB module, such as the HOST-RESOURCES-MIB (RFC 2790).

6965

10.4.2 EPCglobal RFID Reader Management MIB

10.4.2.1 EPCglobal SMI MIB

```
-- ****************************************************************
-- Copyright (c) 2005-2007 EPCglobal Inc(tm), All Rights Reserved.
-- ****************************************************************
EPCGLOBAL-SMI-MIB DEFINITIONS ::= BEGIN

IMPORTS
     enterprises,
     MODULE-IDENTITY
          FROM SNMPv2-SMI;


epcglobal MODULE-IDENTITY
     LAST-UPDATED "200610040000Z"
     ORGANIZATION "EPCglobal, Inc."
     CONTACT-INFO
          "     EPCglobal MIB Administrator
                GS1/EPCglobal, Inc.
                Princeton Pike Corporate Center
                1009 Lenox Drive, Suite 202
                Lawrenceville, NJ 08648
                US

                Tel:   +1 609 620 0200
                Email: mibs@lists.epcglobalinc.org"
     DESCRIPTION
          "The EPCglobal central registration module, containing
          the top-level organization of the EPCglobal private
          enterprise namespace."

     REVISION     "200610040000Z"

     DESCRIPTION
          "Defined in conformance with the EPCglobal Reader Management and
           Reader Protocol specifications"

     ::= {enterprises 22695}  -- assigned by IANA

     --
     -- The EPCglobal private enterprise number assigned by
     -- the Internet Assigned Numbers Authority (IANA).
     --
     epcgSmiManagement          OBJECT IDENTIFIER ::= { epcglobal 1 }
     epcgSmiExperimental        OBJECT IDENTIFIER ::= { epcglobal 2 }
END
```

10.4.2.2 EPCglobal Reader MIB

```
-- Copyright (c) 2005, 2007 EPCglobal Inc(tm), All Rights Reserved.
EPCGLOBAL-READER-MIB DEFINITIONS ::= BEGIN

IMPORTS
     Gauge32,
     Integer32,
     MODULE-IDENTITY,
     NOTIFICATION-TYPE,
     OBJECT-TYPE,
```

```
     Unsigned32,
     Counter32
          FROM SNMPv2-SMI

     SnmpAdminString
          FROM SNMP-FRAMEWORK-MIB

     DateAndTime,
     RowPointer,
     RowStatus,
     TEXTUAL-CONVENTION,
     TruthValue
          FROM SNMPv2-TC

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

     InetAddress,
     InetAddressType
          FROM INET-ADDRESS-MIB

     sysName
          FROM SNMPv2-MIB

     epcgSmiManagement
          FROM EPCGLOBAL-SMI-MIB;

epcgReaderMIB MODULE-IDENTITY
     LAST-UPDATED "200703080000Z"
     ORGANIZATION "EPCglobal, Inc."
     CONTACT-INFO
          "     EPCglobal MIB Administrator
                GS1/EPCglobal, Inc.
                Princeton Pike Corporate Center
                1009 Lenox Drive, Suite 202
                Lawrenceville, NJ 08648
                US

                Tel:   +1 609 620 0200
                Email: mibs@lists.epcglobalinc.org"
     DESCRIPTION
          "The MIB Module describing an EPCglobal compliant RFID reader."

     REVISION     "200703080000Z"

     DESCRIPTION
          "Defined in conformance with the EPCglobal Reader Management and
           Reader Protocol specifications.

           Abbreviations:

           DHCP   - Dynamic Host Configuration Protocol
           DNS    - Domain Name System
           EPC    - Electronic Product Code
           URI    - Uniform Resource Identifier
           RFID   - Radio Frequency Identification
           UTC    - Coordinated Universal Time

          Reference EPCglobal Reader Protocol Specification 1.1"
```

```
     ::= {epcgSmiManagement 1}

--*****************************************************************
--** The EPCglobal Reader MIB module sub-trees
--*****************************************************************
epcgReaderNotifs         OBJECT IDENTIFIER ::= { epcgReaderMIB 0 }
epcgReaderObjects        OBJECT IDENTIFIER ::= { epcgReaderMIB 1 }
epcgReaderConformance    OBJECT IDENTIFIER ::= { epcgReaderMIB 2 }
epcgReaderDevice         OBJECT IDENTIFIER ::= { epcgReaderObjects 1 }
epcgReadPoints           OBJECT IDENTIFIER ::= { epcgReaderObjects 2 }
epcgAntennaReadPoints    OBJECT IDENTIFIER ::= { epcgReaderObjects 3 }
epcgIoPorts              OBJECT IDENTIFIER ::= { epcgReaderObjects 4 }
epcgSources              OBJECT IDENTIFIER ::= { epcgReaderObjects 5 }
epcgNotificationChannels OBJECT IDENTIFIER ::= { epcgReaderObjects 6 }
epcgTriggers             OBJECT IDENTIFIER ::= { epcgReaderObjects 7 }


--*****************************************************************
--** Textual Conventions used by EPCglobal Reader MIB modules      **
--*****************************************************************

EpcgCurrentCountSinceReset ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "A counter that always increments until it is reset."
     SYNTAX  Gauge32

EpcgNotifLevel ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
        "Severity level of an notification."
    SYNTAX INTEGER {
        emergency (0),
        alert (1),
        critical (2),
        error (3),
        warning (4),
        notice (5),
        informational (6),
        debug (7)
    }

EpcgOperationalEnable ::= TEXTUAL-CONVENTION
    STATUS           current
    DESCRIPTION
        "Each bit represent whether a particular state defined
         by the EpcgOperationalStatus is enable for notifications."
    SYNTAX  BITS { unknown(0),
                   other  (1),
                   up     (2),
                   down   (3)
                 }

EpcgOperationalStatus ::= TEXTUAL-CONVENTION
    STATUS           current
    DESCRIPTION
        "A value representing operational status."
    SYNTAX INTEGER { unknown (1),
                     other (2),
                     up (3),
                     down (4)
                   }
```

```
EpcgAdministrativeStatus ::= TEXTUAL-CONVENTION
    STATUS            current
    DESCRIPTION
        "A value representing administrative status."
    SYNTAX INTEGER { up (3),
                     down (4)
                }

EpcgThreshold ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
       "This textual convention defines the allowed values
        used to define a threshold."
    SYNTAX Unsigned32 (0..4294967295)

EpcgTriggerType ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION "The types of allowed triggers."
    SYNTAX INTEGER { none (1),
                     timer (2),
                     continuous (3),
                     ioEdge (4),
                     vendorExtension (5),
                     ioValue (6)
                   }



--*****************************************************************
--** Reader Notifications                                        **
--*****************************************************************

epcgReaderDeviceOperationState NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgRdrDevOperNotifStateLevel,
              epcgRdrDevOperStatusPrior,
              epcgRdrDevOperStatus
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a reader device
         undergoes a change in operational status.  In some
         environment certain states may be transitory and
         do not need to generate notifications.  In addition,
         a device may cycle between two states.  Therefore,
         notifications will be generated whenever the state
         transitions into a state enabled for notifications
         as defined by the epcgRdrDevOperNotifToState
         object.  Also, notifications will be generate whenever
         the state transitions out of a state define by the
         epcgRdrDevOperNotifFromState object.  The
         epcgRdrDevOperStateSuppressInterval object can be
         configured  to suppress the excessive generation of
         notifications."
    ::= { epcgReaderNotifs 1 }

epcgRdrDevMemoryState NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgRdrDevFreeMemoryNotifLevel,
```

```
              epcgRdrDevFreeMemory
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when the reader's
         free memory state changes and the following transitions
         are possible

         normal -> shortage

         A memory shortage is detected when the epcgRdrDevFreeMemory
         object changes to a value that is equal to or below the
         value specified by the epcgRdrDevFreeMemoryOnsetThreshold
         object the object.

         shortage -> normal

         A memory shortage ends when the epcgRdrDevFreeMemory
         object changes to a value that is equal to or greater than
         the value specified by the
         epcgRdrDevFreeMemoryAbateThreshold object.

         The epcgRdrDevFreeMemoryOnsetThreshold and
         epcgRdrDevFreeMemoryAbateThreshold threshold will be
         different for each platform and should be related to the
         amount of available memory.   For example,  a device may
         decided to set the onset threshold to 20 percent of
         total memory and the abate to 35 percent of total memory.

         The value of the epcgRdrDevFreeMemoryOnsetThreshold
         object must be less than the value specified for the
         epcgRdrDevFreeMemoryOnsetAbate object.  In addition,
         the difference between these object should be sufficient
         to prevent excessive number of notification being
         generated when small amounts of memory are used then
         freed.  The epcgRdrDevMemStateSuppressInterval object can
         be configured  to suppress the excessive generation of
         notifications."
    ::= { epcgReaderNotifs 2 }

epcgReadPointOperationState NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgReadPointOperNotifyStateLevel,
              epcgReadPointName,
              epcgReadPointOperStatusPrior,
              epcgReadPointOperStatus
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a read point
         undergoes a change in operational status.  In some
         environment certain states may be transitory and
         do not need to generate notifications.  In addition,
         a device may cycle between two states.  Therefore,
         notifications will be generated whenever the state
         transitions into a state enabled for notifications
         as defined by the epcgRdrDevOperNotifToState
         object.  Also, notifications will be generate whenever
         the state transitions out of a state define by the
         epcgRdrDevOperNotifFromState object.  The
         epcgReadPointOperStateSuppressInterval object can be
```

```
         configured to suppress the excessive generation of
         notifications."
    ::= { epcgReaderNotifs 3 }

epcgReaderAntennaReadFailure NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgAntRdPntReadFailureNotifLevel,
              epcgReadPointName,
              epcgAntRdPntMemoryReadFailures,
              epcgAntRdPntNoiseLevel
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a memory read operation
         across an antenna fails.  To prevent an excessive number
         of notifications the epcgAntRdPntSuppressInterval can be
         configured to suppress generation."
    ::= { epcgReaderNotifs 4 }

epcgReaderAntennaWriteFailure NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgAntRdPntWriteFailuresNotifLevel,
              epcgReadPointName,
              epcgAntRdPntWriteFailures,
              epcgAntRdPntNoiseLevel
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a write operation
         across an antenna fails.  To prevent an excessive number
         of notifications the epcgAntRdPntSuppressInterval can be
         configured to suppress generation"
    ::= { epcgReaderNotifs 5 }

epcgReaderAntennaKillFailure NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgAntRdPntKillFailuresNotifLevel,
              epcgReadPointName,
              epcgAntRdPntKillFailures,
              epcgAntRdPntNoiseLevel
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a kill operation
         across an antenna fails. To prevent an excessive number
         of notifications the epcgAntRdPntSuppressInterval can be
         configured to suppress generation"
    ::= { epcgReaderNotifs 6 }

epcgReaderAntennaEraseFailure NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgAntRdPntEraseFailuresNotifLevel,
              epcgReadPointName,
              epcgAntRdPntEraseFailures,
              epcgAntRdPntNoiseLevel
            }
    STATUS current
    DESCRIPTION
```

```
        "This notification is generated when an erase operation
         across an antenna fails. To prevent an excessive number
         of notifications the epcgAntRdPntSuppressInterval can be
         configured to suppress generation"
    ::= { epcgReaderNotifs 7 }

epcgReaderAntennaLockFailure NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgAntRdPntLockFailuresNotifLevel,
              epcgReadPointName,
              epcgAntRdPntLockFailures,
              epcgAntRdPntNoiseLevel
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a lock operation
         across an antenna fails. To prevent an excessive number
         of notifications the epcgAntRdPntSuppressInterval can be
         configured to suppress generation"
    ::= { epcgReaderNotifs 8 }

epcgReaderIoPortOperationState NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgIoPortOperStatusNotifLevel,
              epcgIoPortName,
              epcgIoPortOperStatusPrior,
              epcgIoPortOperStatus
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when an IO port
         undergoes a change in operation status that falls
         within the set of notification-triggering state transitions
         specified by the epcgIoPortOperStatusNotifFromState and
         epcgIoPortOperStatusNotifToState object values.  The
         epcgIoPortOperStateSuppressInterval object can be
         configured to suppress the excessive generation of
         notifications."
    ::= { epcgReaderNotifs 9 }

epcgReaderSourceOperationState NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgSrcOperStatusNotifyLevel,
              epcgSrcName,
              epcgSrcOperStatusPrior,
              epcgSrcOperStatus
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a source
         undergoes a change in operational status that
         falls  within the set of notification-triggering state
         transitions specified by the epcgSrcOperStatusNotifFromState
         and epcgSrcOperStatusNotifToState object values.  The
         notifications may be generated by a from state, a to state,
         or some combination of from and to states.  The
         epcgRdrSourceOperStateSuppressInterval object can be
         configured  to suppress the excessive generation of
         notifications."
```

```
    ::= { epcgReaderNotifs 10 }

epcgReaderNotificationChanOperState NOTIFICATION-TYPE
    OBJECTS { sysName,
              epcgRdrDevTimeUtc,
              epcgNotifChanOperNotifLevel,
              epcgNotifChanName,
              epcgNotifChanOperStatusPrior,
              epcgNotifChanOperStatus
            }
    STATUS current
    DESCRIPTION
        "This notification is generated when a notification
         channel undergoes a change in operational status that
         falls within the set of notification-triggering state
         transitions specified by the
         epcgNotifChanOperNotifFromState and
         epcgNotifChanOperNotifToState object values.  The
         notifications may be generated by a from state, a to state, or some
         combination of from and to states.  To prevent an excessive number
           of notifications the epcgRdrChanOperStateSuppressInterval can be
           configured to suppress generation"
    ::= { epcgReaderNotifs 11 }


--*****************************************************************
--** Reader Device Information Objects                            **
--*****************************************************************
epcgReaderDeviceInformation
     OBJECT IDENTIFIER ::= { epcgReaderDevice 1 }

-- Many of the objects contained within the ReaderDevice class of the
-- EPCglobal Reader Management abstract model are represented, within
-- the SNMP binding, by objects within in the System group (SNMPv2-MIB,
-- see RFC 3418).  In particular,
--
--     o System group's sysName object represents ReaderDevice.Name
--     o System group's sysDescr object represents ReaderDevice.Manufacturer
--       and ReaderDevice.Model
--     o System group's sysLocation object represents
--       ReaderDevice.LocationDescription
--     o System group's sysContact object represents ReaderDevice.Contact
--     o System group's sysUpTime object represents ReaderDevice.TimeTicks
--
epcgRdrDevDescription    OBJECT-TYPE
    SYNTAX               SnmpAdminString
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
        "The operator's textual description of the reader."
    ::= { epcgReaderDeviceInformation 1 }

epcgRdrDevRole           OBJECT-TYPE
    SYNTAX               SnmpAdminString
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
        "Reader functional role; e.g., Pallet Verification Portal."
    ::= { epcgReaderDeviceInformation 2 }

epcgRdrDevEpc            OBJECT-TYPE
    SYNTAX               SnmpAdminString
```

```
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "Manufacturer assigned EPC URI."
    ::= { epcgReaderDeviceInformation 3 }

epcgRdrDevSerialNumber   OBJECT-TYPE
    SYNTAX              SnmpAdminString
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "Manufacturer assigned serial number."
    ::= { epcgReaderDeviceInformation 4 }

epcgRdrDevTimeUtc        OBJECT-TYPE
    SYNTAX              DateAndTime
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "Reader device wall clock time in UTC."
    ::= { epcgReaderDeviceInformation 5 }

epcgRdrDevCurrentSource OBJECT-TYPE
    SYNTAX              RowPointer
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
       "A pointer to the row in the source table(epcgSourceTable)
        that corresponds to the current source"
    ::= { epcgReaderDeviceInformation 6 }

epcgRdrDevReboot        OBJECT-TYPE
    SYNTAX              TruthValue
    MAX-ACCESS          read-write
    STATUS              current
    DESCRIPTION
       "Setting this object to true(1) causes the device to reboot.
        Reading this object always returns false(2).  Care should be taken
       that the reboot is done after the response is sent, to prevent the
       management station from resending the request and rebooting the device
       again."
    ::= { epcgReaderDeviceInformation 7 }

epcgRdrDevResetStatistics OBJECT-TYPE
    SYNTAX               TruthValue
    MAX-ACCESS           read-write
    STATUS               current
    DESCRIPTION
        " Object used to clear all measurements as follows.
          'true' - Causes the device to set all objects defined
                   with a syntax of EpcgCurrentCountSinceReset
                   to zero.

          'false' - read value.

         Reading this object always returns 'false'."
    ::= { epcgReaderDeviceInformation 8 }

epcgRdrDevResetTimestamp    OBJECT-TYPE
    SYNTAX                  DateAndTime
    MAX-ACCESS              read-only
    STATUS                  current
```

```
    DESCRIPTION
       "This object is set to the current date and time when the
        epcgRdrDevResetStatistics object is set to 'true'.
        Initially it will be set to null value."
    ::= { epcgReaderDeviceInformation 9 }

epcgRdrDevNormalizePowerLevel OBJECT-TYPE
    SYNTAX                  TruthValue
    MAX-ACCESS              read-only
    STATUS                  current
    DESCRIPTION
       " This object is used to indicate how an implementation
         presents power levels as follows.

         'true' - All power levels will be converted be normalize
                  to a range from 0-255.

         'false' - All power levels are presented in raw format
                   specific to a particular vendor.

         This controls the presentation of the following objects.
         -- epcgAntRdPntPowerLevel
        "
    ::= { epcgReaderDeviceInformation 10 }

epcgRdrDevNormalizeNoiseLevel OBJECT-TYPE
    SYNTAX                  TruthValue
    MAX-ACCESS              read-only
    STATUS                  current
    DESCRIPTION
        "This object is used to indicate how an implementation
         presents noise levels as follows.

          'true' - All noise levels will be converted be normalize
                   to a range from 0-255.

          'false' - All noise levels are presented in raw format
                    specific to a particular vendor.

          This controls the presentation of the following objects.
          -- epcgAntRdPntNoiseLevel
        "
    ::= { epcgReaderDeviceInformation 11 }


--*****************************************************************
--** Optional system wide counters
--*****************************************************************
epcgGlobalCountersTable OBJECT-TYPE
    SYNTAX            SEQUENCE OF EpcgGlobalCountersEntry
    MAX-ACCESS        not-accessible
    STATUS            current
    DESCRIPTION
        "The following table aggregates the various counters
         that are maintained per entity like read point,
         antenna read points, and source."
    ::= { epcgReaderDevice 2 }

epcgGlobalCountersEntry OBJECT-TYPE
    SYNTAX            EpcgGlobalCountersEntry
    MAX-ACCESS        not-accessible
    STATUS            current
```

```
    DESCRIPTION
        "Each entry contains information need to communicate with
         each type of server. "
    INDEX { epcgGlobalCountersIndex }
    ::= { epcgGlobalCountersTable 1 }

EpcgGlobalCountersEntry ::= SEQUENCE {
        epcgGlobalCountersIndex  INTEGER,
        epcgGlobalCountersData   EpcgCurrentCountSinceReset
        }

epcgGlobalCountersIndex OBJECT-TYPE
    SYNTAX INTEGER   { antennaTagsIdentified        (1),
                       antennaTagsNotIdentified     (2),
                       antennaMemoryReadOperations  (18),
                       antennaMemoryReadFailures    (3),
                       antennaWriteOperations       (4),
                       antennaWriteFailures         (5),
                       antennaKillOperations        (6),
                       antennaKillFailures          (7),
                       antennaEraseOperations       (8),
                       antennaEraseFailures         (19),
                       antennaLockOperations        (9),
                       antennaLockFailures          (10),
                       sourceUnknownToGlimpsed      (11),
                       sourceGlimpsedToUnknown      (12),
                       sourceGlimpsedToObserved     (13),
                       sourceObservedToLost         (14),
                       sourceLostToGlimpsed         (15),
                       sourceLostToUnknown          (16),
                       triggerMatches               (17)
             }
    MAX-ACCESS              not-accessible
    STATUS                  current
    DESCRIPTION
       "Index to access global counters as follows.  The units of
        each counter will be the same as object indicated by
        index.  For example the antennaIdentifier value references
        the epcgAntRdPntTagsIdentified and it uses 'tags' as units.

         antennaTagsIdentified - Aggregate counter for all instances
                of the epcgAntRdPntTagsIdentified object.

         antennaTagsNotIdentified - Aggregate counter for all
                instances of the epcgAntRdPntTagsNotIdentified object.

         antennaMemoryReadOperations - Aggregate counter for all instances
                of the epcgAntRdPntMemoryReadOperations object.

         antennaMemoryReadFailures - Aggregate counter for all instances
                of the epcgAntRdPntMemoryReadFailures object.

         antennaWriteOperations - Aggregate counter for all instances
                of the epcgAntRdPntWriteOperations object.

         antennaWriteFailures - Aggregate counter for all instances
                of the epcgAntRdPntWriteFailures object.

         antennaKillOperations - Aggregate counter for all instances
                of the epcgAntRdPntKillOperations object.

         antennaKillFailures - Aggregate counter for all instances
```

```
                of the epcgAntRdPntKillFailures object.

        antennaEraseOperations - Aggregate counter for all instances
                of the epcgAntRdPntEraseOperations object.

        antennaEraseFailures - Aggregate counter for all instances
                of the epcgAntRdPntEraseFailures object.

        antennaLockOperations - Aggregate counter for all instances
                of the epcgAntRdPntLockOperations object.

        antennaLockFailures - Aggregate counter for all instances
                of the epcgAntRdPntLockFailures object.

        sourceUnknownToGlimpsed - Aggregate counter for all instances
                of the epcgSrcUnknownToGlimpsedTrans object.

        sourceGlimpsedToUnknown - Aggregate counter for all instances
                of the epcgSrcGlimpsedToUnknownTrans object.

        sourceGlimpsedToObserved - Aggregate counter for all instances
                of theepcgSrcGlimpsedToObservedTrans  object.

        sourceObservedToLost - Aggregate counter for all instances
                of the epcgSrcObservedToLostTrans object.

        sourceLostToGlimpsed - Aggregate counter for all instances
                of the epcgSrcLostToGlimpsedTrans object.

        sourceLostToUnknown - Aggregate counter for all instances
                of the epcgSrcLostToUnknownTrans object.

        triggerMatches - Aggregate counter for all instances
                of the epcgTriggerMatches object.

        source - Aggregate counter for all instances
                of the  object.
      "
    ::= { epcgGlobalCountersEntry 1 }

epcgGlobalCountersData  OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The object contain aggregate count for all instances
         of object as defined by epcgGlobalCountersIndex."
    ::= { epcgGlobalCountersEntry 2 }

--*****************************************************************
--** Reader Device - Operation Objects
--*****************************************************************
epcgReaderDeviceOperation
     OBJECT IDENTIFIER ::= { epcgReaderDevice 3 }

epcgRdrDevOperStatus     OBJECT-TYPE
    SYNTAX               EpcgOperationalStatus
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
       "Current status of the reader device."
    ::= { epcgReaderDeviceOperation 1}
```

```
epcgRdrDevOperStatusPrior OBJECT-TYPE
    SYNTAX                EpcgOperationalStatus
    MAX-ACCESS            accessible-for-notify
    STATUS                current
    DESCRIPTION
       "Status of reader prior to status change that triggered
        the epcgReaderDeviceOperationState notification."
    ::= { epcgReaderDeviceOperation 2 }

epcgRdrDevOperStateEnable OBJECT-TYPE
    SYNTAX                TruthValue
    MAX-ACCESS            read-write
    STATUS                current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderDeviceOperationState notification
        as follows.

        'true'  Indicates that the notification of
                operation state changes has been
                enabled.

        'false' Indicates that the notification of
                operation state changes has been
                disabled."
     DEFVAL { false }
    ::= { epcgReaderDeviceOperation 3 }

epcgRdrDevOperNotifFromState OBJECT-TYPE
    SYNTAX                EpcgOperationalEnable
    MAX-ACCESS            read-write
    STATUS                current
    DESCRIPTION
       "The source state used to control the generation of
        the epcgReaderDeviceOperationState notification."
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgReaderDeviceOperation 4 }

epcgRdrDevOperNotifToState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The destination state used to control the generation of
        the epcgReaderDeviceOperationState notification."
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgReaderDeviceOperation 5 }

epcgRdrDevOperNotifStateLevel  OBJECT-TYPE
    SYNTAX              EpcgNotifLevel
    MAX-ACCESS          read-write
    STATUS              current
    DESCRIPTION
        "The severity level assigned to the
         epcgReaderDeviceOperationStatenotification"
    DEFVAL { error }
        ::= { epcgReaderDeviceOperation 6 }

epcgRdrDevOperStateSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
```

```
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderDeviceOperationState change notifications.

         Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgReaderDeviceOperation 7 }

epcgRdrDevOperStateSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS              "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of device operation state change
         notifications that have been suppressed."
     ::= { epcgReaderDeviceOperation 8 }

--*****************************************************************
--** Reader Device Memory Objects                                 **
--*****************************************************************
epcgReaderDeviceMemory
     OBJECT IDENTIFIER ::= { epcgReaderDevice 4 }

epcgRdrDevFreeMemory     OBJECT-TYPE
    SYNTAX               Gauge32(0..4294967295)
    UNITS                "bytes"
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
        "Reader device free memory, in bytes"
    ::= { epcgReaderDeviceMemory 1 }

epcgRdrDevFreeMemoryNotifEnable OBJECT-TYPE
    SYNTAX                  TruthValue
    MAX-ACCESS              read-write
    STATUS                  current
    DESCRIPTION
       "This object controls the generation of the
        epcgRdrDevMemoryState notification as follows.

        'true'  Indicates that the notification of
                epcgRdrDevMemoryState changes has been enabled.

        'false' Indicates that the notification of
                epcgRdrDevMemoryState changes has been disabled."
    ::= { epcgReaderDeviceMemory 2 }

epcgRdrDevFreeMemoryNotifLevel  OBJECT-TYPE
    SYNTAX                      EpcgNotifLevel
    MAX-ACCESS                  read-write
    STATUS                      current
    DESCRIPTION
       "The severity level assigned to the epcgRdrDevMemoryState
        notification"
    DEFVAL { critical }
    ::= { epcgReaderDeviceMemory 3 }

epcgRdrDevFreeMemoryOnsetThreshold  OBJECT-TYPE
```

```
    SYNTAX                          EpcgThreshold
    MAX-ACCESS                      read-write
    STATUS                          current
    DESCRIPTION
        "The threshold value used to detect low memory situations."
    ::= { epcgReaderDeviceMemory 4 }

epcgRdrDevFreeMemoryAbateThreshold  OBJECT-TYPE
    SYNTAX                          EpcgThreshold
    MAX-ACCESS                      read-write
    STATUS                          current
    DESCRIPTION
        "The threshold value used to detect end of low memory
         situations."
    ::= { epcgReaderDeviceMemory 5 }

epcgRdrDevFreeMemoryStatus  OBJECT-TYPE
    SYNTAX                  INTEGER { shortage (1),
                                      normal   (2)
                            }
    MAX-ACCESS              read-only
    STATUS                  current
    DESCRIPTION
        "The current memory state."
    ::= { epcgReaderDeviceMemory 6 }

epcgRdrDevMemStateSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgRdrDevMemoryState change notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgReaderDeviceMemory 7 }

epcgRdrDevMemStateSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS              "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of device memory state change
         notifications that have been suppressed."
     ::= { epcgReaderDeviceMemory 8 }


--*****************************************************************
--** Reader Device - Server Table
--*****************************************************************
epcgReaderServerTable OBJECT-TYPE
    SYNTAX            SEQUENCE OF EpcgReaderServerEntry
    MAX-ACCESS        not-accessible
    STATUS            current
    DESCRIPTION
        "The following table provide information about the various
         servers used by reader device.  Entries are added and removed
         from this table via epcgReaderServerRowStatus in accordance
```

```
         with the RowStatus convention."
    ::= { epcgReaderDevice 5 }

epcgReaderServerEntry OBJECT-TYPE
    SYNTAX            EpcgReaderServerEntry
    MAX-ACCESS        not-accessible
    STATUS            current
    DESCRIPTION
        "Each entry contains information need to communicate with
         each type of server. "
    INDEX { epcgReaderServerType,
            epcgReaderServerNumber
     }
    ::= { epcgReaderServerTable 1 }

EpcgReaderServerEntry ::= SEQUENCE {
    epcgReaderServerType              INTEGER,
    epcgReaderServerNumber            Unsigned32,
    epcgReaderServerAddressType       InetAddressType,
    epcgReaderServerAddress           InetAddress,
    epcgReaderServerRowStatus         RowStatus
}

epcgReaderServerType         OBJECT-TYPE
    SYNTAX                   INTEGER { dhcp (1),
                                       dns  (2),
                                       time (3)
                             }
    MAX-ACCESS               not-accessible
    STATUS                   current
    DESCRIPTION
       "The type of server as follows.

          dhcp  - DHCP server.
          dns   - Domain Name System.
          time  - Time server used to synchronization Reader's
                  real-time clock.
       "
    ::= { epcgReaderServerEntry 1 }

epcgReaderServerNumber       OBJECT-TYPE
    SYNTAX                   Unsigned32(1..10)
    MAX-ACCESS               not-accessible
    STATUS                   current
    DESCRIPTION
       "The server number.  This index is assigned arbitrarily by
        the SNMP engine and is not required to be saved over
        restarts."
    ::= { epcgReaderServerEntry 2 }

epcgReaderServerAddressType  OBJECT-TYPE
    SYNTAX                   InetAddressType
    MAX-ACCESS               read-create
    STATUS                   current
    DESCRIPTION
       "The type of Internet address to be used to
        communicate with server to collect."
    ::= { epcgReaderServerEntry 3 }

epcgReaderServerAddress      OBJECT-TYPE
    SYNTAX                   InetAddress
    MAX-ACCESS               read-create
```

```
    STATUS                  current
    DESCRIPTION
       "The address of the server and the format of the address
        is specified by the epcgReaderServerAddressType object."
    ::= { epcgReaderServerEntry 4 }

epcgReaderServerRowStatus OBJECT-TYPE
    SYNTAX          RowStatus
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
       "The status of this row."
    ::= { epcgReaderServerEntry 5 }


--*********************************************************************
--** Read Points                                                     **
--*********************************************************************
epcgReadPointTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF EpcgReadPointEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "This table contains a row entry for each of the Reader's
         ReadPoints.  A ReadPoint can be any physical entity that is
         capable of acquiring (or distributing, in the case of tag
         programming) item data.  A single RF tag reader antenna is
         one example of a ReadPoint.  ReadPoints are not limited to
         antennas; for example, a bar code scanning device is another
         example of a ReadPoint.  Hence, ReadPoint is the base class
         for all physical elements over which EPC data is read or
         written.

         The AntennaReadPoint class extends this class and is
         represented by the AntennaReadPointTable.  (A later revision
         of this MIB may introduce a BarCodeReadPointTable that
         augments this table.)

         Typically the number of rows in this table will be equal to
         the number of Antenna ports on a reader"
    ::= { epcgReadPoints 1 }


epcgReadPointEntry OBJECT-TYPE
    SYNTAX      EpcgReadPointEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The properties of a read point.  SNMP operations can
         neither create nor delete rows of this table"
    INDEX { epcgReadPointIndex }
    ::= { epcgReadPointTable 1 }

EpcgReadPointEntry ::= SEQUENCE {
        epcgReadPointIndex                  Unsigned32,
        epcgReadPointName                   SnmpAdminString,
        epcgReadPointDescription            SnmpAdminString,
        epcgReadPointAdminStatus            EpcgAdministrativeStatus,
        epcgReadPointOperStatus             EpcgOperationalStatus,
        epcgReadPointOperStateNotifyEnable  TruthValue,
        epcgReadPointOperNotifyFromState    EpcgOperationalEnable,
        epcgReadPointOperNotifyToState      EpcgOperationalEnable,
```

```
        epcgReadPointOperNotifyStateLevel      EpcgNotifLevel,
       epcgReadPointOperStatusPrior           EpcgOperationalStatus,
       epcgReadPointOperStateSuppressInterval Unsigned32,
       epcgReadPointOperStateSuppressions     Counter32
          }

epcgReadPointIndex OBJECT-TYPE
    SYNTAX      Unsigned32(1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
       "Index used for uniquely identifying a read point within
        the scope of a given RFID Reader.  This index is assigned
        arbitrarily by the SNMP engine and is not required to be
        saved over restarts."
    ::= { epcgReadPointEntry 1 }

epcgReadPointName        OBJECT-TYPE
    SYNTAX               SnmpAdminString
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
       "Unique name assigned to read point by device."
    ::= { epcgReadPointEntry 2 }

epcgReadPointDescription OBJECT-TYPE
    SYNTAX               SnmpAdminString
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
       "A textual description of the read point."
    ::= { epcgReadPointEntry 3 }

epcgReadPointAdminStatus OBJECT-TYPE
    SYNTAX               EpcgAdministrativeStatus
    MAX-ACCESS           read-write
    STATUS               current
    DESCRIPTION
       "The adminstratively specified status of the read point."
    ::= { epcgReadPointEntry 4 }

epcgReadPointOperStatus  OBJECT-TYPE
    SYNTAX               EpcgOperationalStatus
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
       "The read point's current operational status"
    ::= { epcgReadPointEntry 5 }

epcgReadPointOperStateNotifyEnable OBJECT-TYPE
    SYNTAX               TruthValue
    MAX-ACCESS           read-write
    STATUS               current
    DESCRIPTION
       "This object controls the generation of the
        epcgReadPointOperationState notification
        for this read point as follows.

        'true'  Indicates that the notification of read point
                operation state changes has been enabled.

        'false' Indicates that the notification of read point
```

```
                operation state changes has been disable."

    ::= { epcgReadPointEntry 6 }

epcgReadPointOperNotifyFromState OBJECT-TYPE
    SYNTAX              EpcgOperationalEnable
    MAX-ACCESS          read-write
    STATUS              current
    DESCRIPTION
       "The source state used to control the generation of
        the epcgReadPointOperationState notification."
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgReadPointEntry 7 }

epcgReadPointOperNotifyToState OBJECT-TYPE
    SYNTAX              EpcgOperationalEnable
    MAX-ACCESS          read-write
    STATUS              current
    DESCRIPTION
       "The destination state used to control the generation of
        the epcgReadPointOperationState notification."
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgReadPointEntry 8 }

epcgReadPointOperNotifyStateLevel  OBJECT-TYPE
    SYNTAX              EpcgNotifLevel
    MAX-ACCESS          read-write
    STATUS              current
    DESCRIPTION
        "The severity level assigned to the
         epcgReadPointOperationState notification."
    DEFVAL { error }
    ::= { epcgReadPointEntry 9 }

epcgReadPointOperStatusPrior OBJECT-TYPE
    SYNTAX                  EpcgOperationalStatus
    MAX-ACCESS              accessible-for-notify
    STATUS                  current
    DESCRIPTION
         "The past status of read-point prior to the new status change
          that triggered the epcgReadPointOperationState
          notifictation."
    ::= { epcgReadPointEntry 10 }

epcgReadPointOperStateSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReadPointOperationState change notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgReadPointEntry 11 }

epcgReadPointOperStateSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS              "notifications"
     MAX-ACCESS         read-only
```

```
     STATUS            current
     DESCRIPTION
        "The number of read point operational state change
         notifications that have been suppressed."
     ::= { epcgReadPointEntry 12 }


--*********************************************************************
--** Antenna Read Points                                             **
--*********************************************************************
epcgAntReadPointTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF EpcgAntReadPointEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table extends the epcgReadPointTable.  It contains
         one row for each physical antenna, or antenna port, on
         a Reader."
    ::= { epcgAntennaReadPoints 1 }

epcgAntReadPointEntry OBJECT-TYPE
    SYNTAX      EpcgAntReadPointEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The properties of an antenna read point.  SNMP
         operations can neither create nor delete
         rows of this table"

    INDEX { epcgReadPointIndex }
    ::= { epcgAntReadPointTable 1 }

EpcgAntReadPointEntry ::= SEQUENCE {
    epcgAntRdPntTagsIdentified           EpcgCurrentCountSinceReset,
    epcgAntRdPntTagsNotIdentified        EpcgCurrentCountSinceReset,
    epcgAntRdPntMemoryReadOperations     EpcgCurrentCountSinceReset,
    epcgAntRdPntMemoryReadFailures       EpcgCurrentCountSinceReset,
    epcgAntRdPntReadFailureNotifEnable   TruthValue,
    epcgAntRdPntReadFailureNotifLevel    EpcgNotifLevel,
    epcgAntRdPntReadFailureSuppressInterval Unsigned32,
    epcgAntRdPntReadFailureSuppressions  Counter32,
    epcgAntRdPntWriteOperations          EpcgCurrentCountSinceReset,
    epcgAntRdPntWriteFailures            EpcgCurrentCountSinceReset,
    epcgAntRdPntWriteFailuresNotifEnable TruthValue,
    epcgAntRdPntWriteFailuresNotifLevel  EpcgNotifLevel,
    epcgAntRdPntWriteFailureSuppressInterval Unsigned32,
    epcgAntRdPntWriteFailureSuppressions  Counter32,
    epcgAntRdPntKillOperations           EpcgCurrentCountSinceReset,
    epcgAntRdPntKillFailures             EpcgCurrentCountSinceReset,
    epcgAntRdPntKillFailuresNotifEnable  TruthValue,
    epcgAntRdPntKillFailuresNotifLevel   EpcgNotifLevel,
    epcgAntRdPntKillFailureSuppressInterval Unsigned32,
    epcgAntRdPntKillFailureSuppressions  Counter32,
    epcgAntRdPntEraseOperations          EpcgCurrentCountSinceReset,
    epcgAntRdPntEraseFailures      EpcgCurrentCountSinceReset,
    epcgAntRdPntEraseFailuresNotifEnable     TruthValue,
    epcgAntRdPntEraseFailuresNotifLevel     EpcgNotifLevel,
    epcgAntRdPntEraseFailureSuppressInterval Unsigned32,
    epcgAntRdPntEraseFailureSuppressions  Counter32,
    epcgAntRdPntLockOperations           EpcgCurrentCountSinceReset,
    epcgAntRdPntLockFailures             EpcgCurrentCountSinceReset,
    epcgAntRdPntLockFailuresNotifEnable TruthValue,
```

```
    epcgAntRdPntLockFailuresNotifLevel  EpcgNotifLevel,
    epcgAntRdPntLockFailureSuppressInterval Unsigned32,
    epcgAntRdPntLockFailureSuppressions  Counter32,
    epcgAntRdPntPowerLevel               Integer32,
    epcgAntRdPntNoiseLevel               Integer32,
    epcgAntRdPntTimeEnergized            EpcgCurrentCountSinceReset
}

epcgAntRdPntTagsIdentified OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    UNITS       "tags"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Count of tags successfully identified by this antenna read point."
    ::= { epcgAntReadPointEntry 1 }

epcgAntRdPntTagsNotIdentified OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    UNITS       "tags"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Count of tags failed to be identified by this antenna
         read point.  If the device does not implement the technology to
        unambiguously determine a failed tag singulation, it is acceptable to
        leave this value at zero."
    ::= { epcgAntReadPointEntry 2 }

epcgAntRdPntMemoryReadOperations OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    UNITS       "tags"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Count of tags successfully having their memory read by this
         antenna read point."
    ::= { epcgAntReadPointEntry 25 }

epcgAntRdPntMemoryReadFailures OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    UNITS       "tags"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Count of tags failed to have their memory read by this
         antenna read point."
    ::= { epcgAntReadPointEntry 3 }

epcgAntRdPntReadFailureNotifEnable OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderAntennaReadFailure notification as follows.

        'true'  Indicates that the notification of antenna read
                failure has been enabled.

        'false' Indicates that the notification of antenna read
                failure changes has been disabled."
```

```
    ::= { epcgAntReadPointEntry 4 }

epcgAntRdPntReadFailureNotifLevel OBJECT-TYPE
    SYNTAX      EpcgNotifLevel
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
       "The severity level assigned to the
        epcgReaderAntennaReadFailure notification."
    DEFVAL { error }
    ::= { epcgAntReadPointEntry 5 }

epcgAntRdPntReadFailureSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderAntennaReadFailure notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgAntReadPointEntry 26 }

epcgAntRdPntReadFailureSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS             "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of epcgReaderAntennaReadFailure
         notifications that have been suppressed."
     ::= { epcgAntReadPointEntry 27 }

epcgAntRdPntWriteOperations OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    UNITS       "write operations"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
       "Number of writes successfully done by this antenna read point."
    ::= { epcgAntReadPointEntry 6 }

epcgAntRdPntWriteFailures OBJECT-TYPE
    SYNTAX      EpcgCurrentCountSinceReset
    UNITS       "write operations"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Number of writes attempted and failed by this antenna
         read point."
    ::= { epcgAntReadPointEntry 7 }

epcgAntRdPntWriteFailuresNotifEnable OBJECT-TYPE
    SYNTAX                          TruthValue
    MAX-ACCESS                      read-write
    STATUS                          current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderAntennaWriteFailure notification as follows.
```

```
        'true'  Indicates that the notification of antenna write
                failure changes has been enabled.

        'false' Indicates that the notification of ntenna write
                failure changes has been disabled."
    ::= { epcgAntReadPointEntry 8 }

epcgAntRdPntWriteFailuresNotifLevel  OBJECT-TYPE
    SYNTAX                           EpcgNotifLevel
    MAX-ACCESS                       read-write
    STATUS                           current
    DESCRIPTION
        "The severity level assigned to the
         epcgReaderAntennaWriteFailure notification."
    DEFVAL { notice }
    ::= { epcgAntReadPointEntry 9 }

epcgAntRdPntWriteFailureSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderAntennaWriteFailure notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgAntReadPointEntry 28 }

epcgAntRdPntWriteFailureSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS             "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of epcgReaderAntennaWriteFailure
         notifications that have been suppressed."
     ::= { epcgAntReadPointEntry 29 }

epcgAntRdPntKillOperations OBJECT-TYPE
    SYNTAX                EpcgCurrentCountSinceReset
    UNITS                 "kill operations"
    MAX-ACCESS            read-only
    STATUS                current
    DESCRIPTION
        "Number of kill operations successfully done by this antenna read
point."
    ::= { epcgAntReadPointEntry 10 }

epcgAntRdPntKillFailures OBJECT-TYPE
    SYNTAX               EpcgCurrentCountSinceReset
    UNITS                "kill operations"
    MAX-ACCESS           read-only
    STATUS               current
    DESCRIPTION
        "Number of kill operations attempted and failed by this
         antenna read point."
    ::= { epcgAntReadPointEntry 11 }
```

```
epcgAntRdPntKillFailuresNotifEnable OBJECT-TYPE
    SYNTAX                          TruthValue
    MAX-ACCESS                      read-write
    STATUS                          current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderAntennaKillFailure notification as follows.

        'true'  Indicates that the notification of antenna kill
                failure has been enabled.

        'false' Indicates that the notification of antenna kill
                failure has been disabled."
    ::= { epcgAntReadPointEntry 12 }

epcgAntRdPntKillFailuresNotifLevel  OBJECT-TYPE
    SYNTAX                          EpcgNotifLevel
    MAX-ACCESS                      read-write
    STATUS                          current
    DESCRIPTION
       "The severity level assigned to the
         epcgReaderAntennaKillFailure notification."
    DEFVAL { notice }
    ::= { epcgAntReadPointEntry 13 }

epcgAntRdPntKillFailureSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderAntennaKillFailure notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgAntReadPointEntry 30 }

epcgAntRdPntKillFailureSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS             "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of epcgReaderAntennaKillFailure
         notifications that have been suppressed."
     ::= { epcgAntReadPointEntry 31 }

epcgAntRdPntEraseOperations OBJECT-TYPE
    SYNTAX                  EpcgCurrentCountSinceReset
    UNITS                   "erase operations"
    MAX-ACCESS              read-only
    STATUS                  current
    DESCRIPTION
        "Number of erase operations successfully completed by this antenna
         read point. "
    ::= { epcgAntReadPointEntry 14 }

epcgAntRdPntEraseFailures OBJECT-TYPE
    SYNTAX                      EpcgCurrentCountSinceReset
    UNITS                       "erase operations"
```

```
    MAX-ACCESS                  read-only
    STATUS                      current
    DESCRIPTION
        "Number of erase operations attempted and failed by this
         antenna read point."
    ::= { epcgAntReadPointEntry 15 }

epcgAntRdPntEraseFailuresNotifEnable OBJECT-TYPE
    SYNTAX                       TruthValue
    MAX-ACCESS                   read-write
    STATUS                       current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderAntennaEraseFailure notification as follows.

        'true'  Indicates that the notification of antenna erase
                failure changes has been enabled.

        'false' Indicates that the notification of antenna erase failure
                changes has been disabled."
    ::= { epcgAntReadPointEntry 16 }

epcgAntRdPntEraseFailuresNotifLevel  OBJECT-TYPE
    SYNTAX                        EpcgNotifLevel
    MAX-ACCESS                    read-write
    STATUS                        current
    DESCRIPTION
        "The severity level assigned to the
         epcgReaderAntennaEraseFailure notification."
    DEFVAL { notice }
    ::= { epcgAntReadPointEntry 17 }

epcgAntRdPntEraseFailureSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderAntennaEraseFailure notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgAntReadPointEntry 32 }

epcgAntRdPntEraseFailureSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS             "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of epcgReaderAntennaEraseFailure
         notifications that have been suppressed."
     ::= { epcgAntReadPointEntry 33 }

epcgAntRdPntLockOperations OBJECT-TYPE
    SYNTAX                 EpcgCurrentCountSinceReset
    UNITS                  "lock operations"
    MAX-ACCESS             read-only
    STATUS                 current
    DESCRIPTION
```

```
       "Number of lock operations successfully issued by this antenna
        read point."
    ::= { epcgAntReadPointEntry 18 }

epcgAntRdPntLockFailures OBJECT-TYPE
    SYNTAX              EpcgCurrentCountSinceReset
    UNITS               "lock operations"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "Number of lock operations attempted and failed by this
         antenna read point."
    ::= { epcgAntReadPointEntry 19 }

epcgAntRdPntLockFailuresNotifEnable OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderAntennaLockFailure notification
        as follows.

        'true'  Indicates that the notification of antenna lock
                failure changes has been enabled.

        'false' Indicates that the notification of antenna lock
                failure changes has been disabled."
    ::= { epcgAntReadPointEntry 20 }

epcgAntRdPntLockFailuresNotifLevel  OBJECT-TYPE
    SYNTAX              EpcgNotifLevel
    MAX-ACCESS          read-write
    STATUS              current
    DESCRIPTION
        "The severity level assigned to the
         epcgReaderAntennaLockFailure notification."
    DEFVAL { notice }
    ::= { epcgAntReadPointEntry 21 }

epcgAntRdPntLockFailureSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderAntennaLockFailure notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgAntReadPointEntry 34 }

epcgAntRdPntLockFailureSuppressions OBJECT-TYPE
     SYNTAX             Counter32
     UNITS             "notifications"
     MAX-ACCESS         read-only
     STATUS             current
     DESCRIPTION
        "The number of epcgReaderAntennaLockFailure
         notifications that have been suppressed."
```

```
     ::= { epcgAntReadPointEntry 35 }

epcgAntRdPntPowerLevel OBJECT-TYPE
    SYNTAX             Integer32(-2147483648..2147483647)
    MAX-ACCESS         read-only
    STATUS             current
    DESCRIPTION
       "The power level of this antenna read point.
        The information provided by this object will be in
        different formats depending on the value of the
        epcgRdrDevNormalizePowerLevel object.  The different
        formats are provide in the describe for the
        epcgRdrDevNormalizePowerLevel object."
    ::= { epcgAntReadPointEntry 22 }

epcgAntRdPntNoiseLevel OBJECT-TYPE
    SYNTAX              Integer32(-2147483648..2147483647)
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
       "The noise level of this antenna read point.
        The information provided by this object will be in
        different formats depending on the value of the
        epcgRdrDevNormalizePowerLevel object.  The different
        formats are provide in the describe for the
        epcgRdrDevNormalizePowerLevel object."
    ::= { epcgAntReadPointEntry 23 }

epcgAntRdPntTimeEnergized OBJECT-TYPE
    SYNTAX              EpcgCurrentCountSinceReset
    UNITS               "milli-seconds"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The amount of time ,in milli-seconds, that this antenna
         read point since system started or value
         was cleared."
    ::= { epcgAntReadPointEntry 24 }

--*****************************************************************
--** IO Ports
--*****************************************************************
epcgIoPortTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF EpcgIoPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "There is a row entry in this table for each data IO port
         that the reader supports.  Typically these data IO ports
         send and/or receive event triggers."
    ::= { epcgIoPorts 1 }

epcgIoPortEntry OBJECT-TYPE
    SYNTAX      EpcgIoPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The properties of a single IO port.  SNMP operations
         can neither create nor delete rows in the epcgIoPortTable"
    INDEX { epcgIoPortIndex }
    ::= { epcgIoPortTable 1 }
```

```
EpcgIoPortEntry ::= SEQUENCE {
         epcgIoPortIndex                      Unsigned32,
         epcgIoPortName                       SnmpAdminString,
         epcgIoPortAdminStatus                EpcgAdministrativeStatus,
         epcgIoPortOperStatus                 EpcgOperationalStatus,
         epcgIoPortOperStatusNotifEnable      TruthValue,
         epcgIoPortOperStatusNotifLevel       EpcgNotifLevel,
         epcgIoPortOperStatusNotifFromState   EpcgOperationalEnable,
         epcgIoPortOperStatusNotifToState     EpcgOperationalEnable,
         epcgIoPortDescription                SnmpAdminString,
        epcgIoPortOperStatusPrior            EpcgOperationalStatus,
        epcgIoPortOperStateSuppressInterval Unsigned32,
        epcgIoPortOperStateSuppressions      Counter32
}

epcgIoPortIndex OBJECT-TYPE
    SYNTAX          Unsigned32(1..2147483647)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Index used to uniquely identifying an IO port within
         the scope of a given RFID Reader.  This index is assigned
         arbitrarily by the SNMP engine and is not required to
         be saved over restarts."
        ::= { epcgIoPortEntry 1 }

epcgIoPortName OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "The name of this IO Port."
        ::= { epcgIoPortEntry 2 }

epcgIoPortAdminStatus OBJECT-TYPE
    SYNTAX          EpcgAdministrativeStatus
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
         "The adminstratively specified status of the IO port."
        ::= { epcgIoPortEntry 3 }

epcgIoPortOperStatus OBJECT-TYPE
    SYNTAX          EpcgOperationalStatus
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "The read point's current operational status."
        ::= { epcgIoPortEntry 4 }

epcgIoPortOperStatusNotifEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderIoPortOperationState notification
        as follows.

        'true'  Indicates that the notification of IO port operation
                state changes has been enabled.
```

```
        'false' Indicates that the notification of IO port operation
                state changes has been disabled."
        ::= { epcgIoPortEntry 5 }

epcgIoPortOperStatusNotifLevel  OBJECT-TYPE
    SYNTAX          EpcgNotifLevel
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
           "The severity level assigned to the
            epcgReaderDeviceOperationState notification"
    DEFVAL { error }
        ::= { epcgIoPortEntry 6 }

epcgIoPortOperStatusNotifFromState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
         "The source state for state transitions triggering
          epcgReaderIoPortOperationState notifications"
    DEFVAL { { unknown, other, up, down} }
        ::= { epcgIoPortEntry 7 }

epcgIoPortOperStatusNotifToState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
         "The destination state for state transitions triggering
          epcgReaderIoPortOperationState notifications"
    DEFVAL { { unknown, other, up, down} }
        ::= { epcgIoPortEntry 8 }

epcgIoPortDescription OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "The operator's textual description of the IO Port."
        ::= { epcgIoPortEntry 9 }

epcgIoPortOperStatusPrior OBJECT-TYPE
    SYNTAX          EpcgOperationalStatus
    MAX-ACCESS      accessible-for-notify
    STATUS          current
    DESCRIPTION
         "Status of IO port prior to status change that
          triggered THIS epcgReaderIoPortOperationState."
        ::= { epcgIoPortEntry 10 }

epcgIoPortOperStateSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderIoPortOperationState change notifications.

        Notification will not be suppressed when zero value is
        specified."
```

```
    DEFVAL { 0 }
    ::= { epcgIoPortEntry 11 }

epcgIoPortOperStateSuppressions OBJECT-TYPE
     SYNTAX            Counter32
     UNITS            "notifications"
     MAX-ACCESS        read-only
     STATUS            current
     DESCRIPTION
        "The number of IO port state change
         notifications that have been suppressed."
     ::= { epcgIoPortEntry 12 }

--*****************************************************************
--** (Logical) Sources (of tag data)
--*****************************************************************
epcgSourceTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF EpcgSourceEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
         "This table contains a row for each Source configured
          within a Reader.  A Source is a logical entity with
          zero, one, or more ReadPoints assigned to it.  A
          Source object may be associated with zero, one or
          more ReadTriggers and zero, one or more Notification
          Channels."
        ::= { epcgSources 1 }

epcgSourceEntry OBJECT-TYPE
    SYNTAX          EpcgSourceEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
         "The properties of a tag data source. SNMP operations can
          neither create nor delete rows of this table."

        INDEX { epcgSrcIndex }
        ::= { epcgSourceTable 1 }

EpcgSourceEntry ::= SEQUENCE {
        epcgSrcIndex                    Unsigned32,
        epcgSrcName                     SnmpAdminString,
        epcgSrcReadCyclesPerTrigger     Unsigned32,
        epcgSrcReadDutyCycle            Gauge32,
        epcgSrcReadTimeout              Unsigned32,
        epcgSrcGlimpsedTimeout          Unsigned32,
        epcgSrcObservedThreshold        Unsigned32,
        epcgSrcObservedTimeout          Unsigned32,
        epcgSrcLostTimeout              Unsigned32,
        epcgSrcUnknownToGlimpsedTrans   EpcgCurrentCountSinceReset,
        epcgSrcGlimpsedToUnknownTrans   EpcgCurrentCountSinceReset,
        epcgSrcGlimpsedToObservedTrans  EpcgCurrentCountSinceReset,
        epcgSrcObservedToLostTrans      EpcgCurrentCountSinceReset,
        epcgSrcLostToGlimpsedTrans      EpcgCurrentCountSinceReset,
        epcgSrcLostToUnknownTrans       EpcgCurrentCountSinceReset,
        epcgSrcAdminStatus              EpcgAdministrativeStatus,
        epcgSrcOperStatus               EpcgOperationalStatus,
        epcgSrcOperStatusNotifEnable    TruthValue,
        epcgSrcOperStatusNotifFromState EpcgOperationalEnable,
        epcgSrcOperStatusNotifToState   EpcgOperationalEnable,
        epcgSrcOperStatusNotifyLevel    EpcgNotifLevel,
```

```
        epcgSrcSupportsWriteOperations  TruthValue,
        epcgSrcOperStatusPrior          EpcgOperationalStatus,
        epcgSrcOperStateSuppressInterval Unsigned32,
        epcgSrcOperStateSuppressions    Counter32
}

epcgSrcIndex OBJECT-TYPE
    SYNTAX          Unsigned32(1..2147483647)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
       "Index used for uniquely identifying a source within the
        scope of a given RFID reader.  This index is assigned
        arbitrarily  by the SNMP engine and is not required to
        be saved over restarts."
    ::= { epcgSourceEntry 1 }

epcgSrcName OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "Unique name assigned to a source."
    ::= { epcgSourceEntry 2 }

epcgSrcReadCyclesPerTrigger OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "cycles"
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
         "Read cycles attempted per trigger."
    ::= { epcgSourceEntry 3 }

epcgSrcReadDutyCycle OBJECT-TYPE
    SYNTAX           Gauge32(0..100)
    UNITS            "percentage"
    MAX-ACCESS       read-write
    STATUS           current
    DESCRIPTION
       "Duty cycle for this read source."
    ::= { epcgSourceEntry 4 }

epcgSrcReadTimeout OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "milli-seconds"
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "Read timeout in milli-seconds for this source."
    ::= { epcgSourceEntry 5 }

epcgSrcGlimpsedTimeout OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "milli-seconds"
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
         "Glimpsed timeout in milli-seconds for this source."
    ::= { epcgSourceEntry 6 }

epcgSrcObservedThreshold OBJECT-TYPE
```

```
    SYNTAX          Unsigned32
    UNITS           "milli-seconds"
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        "Number of milli-seconds required for tag to be considered
         observed for this source"
    ::= { epcgSourceEntry 7 }

epcgSrcObservedTimeout OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "milli-seconds"
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        "Number of milli-seconds required for tag to be considered
         observed for this source"
    ::= { epcgSourceEntry 8 }

epcgSrcLostTimeout OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           "milli-seconds"
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        "Number of milli-seconds required for tag to be considered lost
         for this source"
    ::= { epcgSourceEntry 9 }

epcgSrcUnknownToGlimpsedTrans OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    UNITS           "transitions"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "Number of tags transitioning from 'unknown' status to
       'glimpsed' status for this source."
    ::= { epcgSourceEntry 10 }

epcgSrcGlimpsedToUnknownTrans OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    UNITS           "transitions"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "Number of tags transitioning from 'glimpsed' status to
        'unknown' status for this source."
    ::= { epcgSourceEntry 11 }

epcgSrcGlimpsedToObservedTrans OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    UNITS           "transitions"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of tags transitioning from 'glimpsed' status
         to 'observed' status for this source."
    ::= { epcgSourceEntry 12 }

epcgSrcObservedToLostTrans OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    UNITS           "transitions"
```

```
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "Number of tags transitioning from 'observed' status
         to 'lost' status for this source."
    ::= { epcgSourceEntry 13 }

epcgSrcLostToGlimpsedTrans OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    UNITS           "transitions"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "Number of tags transitioning from 'lost' status to
        'glimpsed' status for this source"
    ::= { epcgSourceEntry 14 }

epcgSrcLostToUnknownTrans OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "Number of tags transitioning from 'lost' status to
        'unknown' status for this source"
    ::= { epcgSourceEntry 15 }

epcgSrcAdminStatus OBJECT-TYPE
    SYNTAX          EpcgAdministrativeStatus
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
         "The adminstratively specified status of the source."
    ::= { epcgSourceEntry 16 }

epcgSrcOperStatus OBJECT-TYPE
    SYNTAX          EpcgOperationalStatus
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "The source's current operational status."
    ::= { epcgSourceEntry 17 }

epcgSrcOperStatusNotifEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderSourceOperationState notification
        as follows.

        'true'  Indicates that the notification of Source state
                changes has been enabled.

        'false' Indicates that the notification of Source state
                changes has been disabled."
    ::= { epcgSourceEntry 18 }

epcgSrcOperStatusNotifFromState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
```

```
    DESCRIPTION
       "The source state for state transitions triggering
        epcgReaderSourceOperationState notifications"
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgSourceEntry 19 }

epcgSrcOperStatusNotifToState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The destination state for state transitions triggering
        epcgReaderSourceOperationState notifications"
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgSourceEntry 20 }

epcgSrcOperStatusNotifyLevel  OBJECT-TYPE
    SYNTAX          EpcgNotifLevel
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The severity level assigned to the
        epcgReaderSourceOperationState notification."
    DEFVAL { error }
    ::= { epcgSourceEntry 21 }

epcgSrcSupportsWriteOperations OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object provides an indication of the capability of the
         source as follows.
         'true'  - source supports read and write operations.
         'false' - source supports only read operations."
    ::= { epcgSourceEntry 22 }

epcgSrcOperStatusPrior OBJECT-TYPE
    SYNTAX          EpcgOperationalStatus
    MAX-ACCESS      accessible-for-notify
    STATUS          current
    DESCRIPTION
         "Status of EPC data source prior to status change that
          triggered the epcgReaderSourceOperationState
          notification."
    ::= { epcgSourceEntry 23 }

epcgSrcOperStateSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderSourceOperationState change notifications.

        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgSourceEntry 24 }

epcgSrcOperStateSuppressions OBJECT-TYPE
```

```
     SYNTAX            Counter32
     UNITS             "notifications"
     MAX-ACCESS        read-only
     STATUS            current
     DESCRIPTION
        "The number of source operational state change
         notifications that have been suppressed."
    ::= { epcgSourceEntry 25 }

--*****************************************************************
--** Notification Channels
--*****************************************************************
epcgNotificationChannelTable OBJECT-TYPE
    SYNTAX                  SEQUENCE OF EpcgNotificationChannelEntry
    MAX-ACCESS              not-accessible
    STATUS                  current
    DESCRIPTION
         "This table contains one row per notification channel.
          A notification channel is associated with zero, one
          or more Sources, and zero, one Notification Triggers"
    ::= { epcgNotificationChannels 1 }

epcgNotificationChannelEntry OBJECT-TYPE
    SYNTAX                  EpcgNotificationChannelEntry
    MAX-ACCESS              not-accessible
    STATUS                   current
    DESCRIPTION
         "The properties of a notification channel.  SNMP operations
          can neither create nor delete rows of this table"
        INDEX { epcgNotifChanIndex }
    ::= { epcgNotificationChannelTable 1 }

EpcgNotificationChannelEntry ::= SEQUENCE {
          epcgNotifChanIndex              Unsigned32,
          epcgNotifChanName               SnmpAdminString,
          epcgNotifChanAddressType        InetAddressType,
          epcgNotifChanAddress            InetAddress,
          epcgNotifChanLastAttempt        DateAndTime,
          epcgNotifChanLastSuccess        DateAndTime,
          epcgNotifChanAdminStatus        EpcgAdministrativeStatus,
          epcgNotifChanOperStatus         EpcgOperationalStatus,
          epcgNotifChanOperNotifEnable    TruthValue,
          epcgNotifChanOperNotifLevel     EpcgNotifLevel,
          epcgNotifChanOperNotifFromState EpcgOperationalEnable,
          epcgNotifChanOperNotifToState   EpcgOperationalEnable,
         epcgNotifChanOperStatusPrior           EpcgOperationalStatus,
         epcgNotifChanOperStateSuppressInterval Unsigned32,
         epcgNotifChanOperStateSuppressions     Counter32
}

epcgNotifChanIndex OBJECT-TYPE
    SYNTAX          Unsigned32(1..2147483647)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Index used for uniquely identifying a notification channel
         within the scope of a given RFID Reader.  This index is
         assigned arbitrarily by the SNMP engine and is not required
         to be saved over restarts."
    ::= { epcgNotificationChannelEntry 1 }

epcgNotifChanName OBJECT-TYPE
```

```
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "Unique name assigned to Notification Channel."
    ::= { epcgNotificationChannelEntry 2 }

epcgNotifChanAddressType OBJECT-TYPE
    SYNTAX          InetAddressType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "The type of Internet address used to identify the
        notification channel's destination"
    ::= { epcgNotificationChannelEntry 3 }

epcgNotifChanAddress OBJECT-TYPE
    SYNTAX          InetAddress
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "The Internet address identifying the notification
        channel's destination."
    ::= { epcgNotificationChannelEntry 4 }

epcgNotifChanLastAttempt OBJECT-TYPE
    SYNTAX          DateAndTime
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "Wall-clock time of last attempted notification by this
          notification channel."
    ::= { epcgNotificationChannelEntry 5 }

epcgNotifChanLastSuccess OBJECT-TYPE
    SYNTAX          DateAndTime
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "Wall-clock time of last successful notification sent by
          this notification channel."
    ::= { epcgNotificationChannelEntry 6 }

epcgNotifChanAdminStatus OBJECT-TYPE
    SYNTAX          EpcgAdministrativeStatus
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The adminstratively specified status of the
        notification channel."
    ::= { epcgNotificationChannelEntry 7 }

epcgNotifChanOperStatus OBJECT-TYPE
    SYNTAX          EpcgOperationalStatus
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "The notification channel's current operational status."
    ::= { epcgNotificationChannelEntry 8 }

epcgNotifChanOperNotifEnable OBJECT-TYPE
    SYNTAX          TruthValue
```

```
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "This object controls the generation of the
        epcgReaderNotificationChanOperState notification
        as follows.

        'true' - Indicates that the sending of notification channel
                 operation state changes has been enabled.

        'false' - Indicates that the sending of notification channel
                  operation state changes has been disabled."
    ::= { epcgNotificationChannelEntry 9 }

epcgNotifChanOperNotifLevel  OBJECT-TYPE
    SYNTAX          EpcgNotifLevel
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The severity level assigned to the
        epcgReaderNotificationChanOperState notification."
    DEFVAL { error }
    ::= { epcgNotificationChannelEntry 10 }

epcgNotifChanOperNotifFromState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The source state for the state transitions triggering
        epcgReaderNotificationChanOperState notificatons"
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgNotificationChannelEntry 11 }

epcgNotifChanOperNotifToState OBJECT-TYPE
    SYNTAX          EpcgOperationalEnable
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
       "The destination state for the state transitions triggering
        epcgReaderNotificationChanOperState  notifications"
    DEFVAL { { unknown, other, up, down} }
    ::= { epcgNotificationChannelEntry 12 }

epcgNotifChanOperStatusPrior OBJECT-TYPE
    SYNTAX          EpcgOperationalStatus
    MAX-ACCESS      accessible-for-notify
    STATUS          current
    DESCRIPTION
        "The status of notification channel prior to the current
         status  change that triggered the
         epcgReaderNotificationChanOperState notification"
    ::= { epcgNotificationChannelEntry 13 }

epcgNotifChanOperStateSuppressInterval OBJECT-TYPE
     SYNTAX      Unsigned32 (0 | 1..3600 )
     UNITS       "seconds"
     MAX-ACCESS  read-write
     STATUS      current
     DESCRIPTION
       "The length of the interval used to suppress generation
        of epcgReaderNotificationChanOperState change notifications.
```

```
        Notification will not be suppressed when zero value is
        specified."
    DEFVAL { 0 }
    ::= { epcgNotificationChannelEntry 14 }

epcgNotifChanOperStateSuppressions OBJECT-TYPE
     SYNTAX            Counter32
     UNITS             "notifications"
     MAX-ACCESS        read-only
     STATUS            current
     DESCRIPTION
         "The number of channel operational state change
        notifications that have been suppressed."
     ::= { epcgNotificationChannelEntry 15 }

--*****************************************************************
--** Triggers
--*****************************************************************
epcgTriggerTable OBJECT-TYPE
    SYNTAX       SEQUENCE OF EpcgTriggerEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
         "This table contains one row per trigger.  A
          Trigger is associated with zero or one IO Port.
          A Trigger may be associated with zero or more Sources
          and zero or more Notification Channels"
    ::= { epcgTriggers 1 }

epcgTriggerEntry OBJECT-TYPE
    SYNTAX       EpcgTriggerEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
         "The properties of a trigger. SNMP operations can neither
          create nor delete rows in the epcgTriggerTable."
        INDEX { epcgTrigIndex }
    ::= { epcgTriggerTable 1 }

EpcgTriggerEntry ::= SEQUENCE {
        epcgTrigIndex           Unsigned32,
        epcgTrigName            SnmpAdminString,
        epcgTrigType            EpcgTriggerType,
        epcgTrigParameters      SnmpAdminString,
        epcgTriggerMatches      EpcgCurrentCountSinceReset,
        epcgTrigIoPort          RowPointer
        }

epcgTrigIndex       OBJECT-TYPE
    SYNTAX          Unsigned32(1..2147483647)
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
       "Index uniquely identifying this trigger.  This index is
        assigned arbitrarily by the SNMP engine and is not
        required to be saved over restarts."
    ::= { epcgTriggerEntry 1 }

epcgTrigName        OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-only
```

```
    STATUS          current
    DESCRIPTION
         "Name of this trigger."
    ::= { epcgTriggerEntry 2 }

epcgTrigType        OBJECT-TYPE
    SYNTAX          EpcgTriggerType
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "The trigger type."
    ::= { epcgTriggerEntry 3 }

epcgTrigParameters  OBJECT-TYPE
    SYNTAX          SnmpAdminString
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
       "The epcgTrigParameters value must comply with the constraints
        defined for the 'triggervalue' datatype in the EPCglobal
        Reader Protocol Specification, 1.1 or greater."
    REFERENCE
         "EPCglobal Reader Protocol Specification section 6.4"
    ::= { epcgTriggerEntry 4 }

epcgTriggerMatches  OBJECT-TYPE
    SYNTAX          EpcgCurrentCountSinceReset
    UNITS           "occurrences"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "Number of times this trigger has matched."
    ::= { epcgTriggerEntry 5 }

epcgTrigIoPort OBJECT-TYPE
    SYNTAX          RowPointer
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
         "A pointer to an entry in the IoPort table.  A
          trigger need not correspond to an IO port
          (e.g., a trigger of type timer has not IO port
          association), in which case this
          RowPointer object has the value zeroDotzero"
    ::= { epcgTriggerEntry 6 }

--*****************************************************************
--** Notification Trigger Table
--*****************************************************************
epcgNotifTrigTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF EpcgNotifTrigEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
       "This table associates triggers with notification channels.
        Multiple triggers may be associated with the same
        notification channel and a single trigger may be associated
        with multiple notification channels.  There will be one
        row entry for each unique pairing of trigger and notification
        channel. "
    ::= { epcgTriggers 2 }
```

```
epcgNotifTrigEntry OBJECT-TYPE
    SYNTAX          EpcgNotifTrigEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
       "Identifies a single pairing of a trigger and notification
        channel.  Entries can be added and removed from this
        table via epcgNotifTrigRowStatus in accordance with the
        RowStatus convention."
        INDEX { epcgNotifChanIndex,
                epcgTrigIndex }
    ::= { epcgNotifTrigTable 1 }

EpcgNotifTrigEntry ::= SEQUENCE {
        epcgNotifTrigRowStatus          RowStatus
        }

epcgNotifTrigRowStatus OBJECT-TYPE
    SYNTAX             RowStatus
    MAX-ACCESS         read-create
    STATUS             current
    DESCRIPTION
       "The status of this row."
    ::= { epcgNotifTrigEntry 1 }

--*********************************************************************
--** Read Trigger Table
--*********************************************************************
epcgReadTrigTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF EpcgReadTrigEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
       "This table associates Sources with Triggers.  Multiple
        triggers may be associated with the same Source and a
        single trigger may be associated with multiple Sources.
        There will be one row entry for each unique pairing of
        Trigger and Source.  Entries can be added and removed from
        this table via epcgReadTrigRowStatus in accordance with the
        RowStatus convention."
    ::= { epcgTriggers 3 }

epcgReadTrigEntry OBJECT-TYPE
    SYNTAX        EpcgReadTrigEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
       "Identifies a single pairing of a Source and Trigger."
        INDEX { epcgSrcIndex,
                epcgTrigIndex }
    ::= { epcgReadTrigTable 1 }

EpcgReadTrigEntry ::= SEQUENCE {
        epcgReadTrigRowStatus          RowStatus
        }

epcgReadTrigRowStatus OBJECT-TYPE
    SYNTAX          RowStatus
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
       "The status of this row."
```

```
    ::= { epcgReadTrigEntry 1 }

--*********************************************************************
--** ReadPoint-Source Table
--*********************************************************************
epcgRdPntSrcTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF EpcgRdPntSrcEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
       "This table associates ReadPoints with (tag) Sources.
        Multiple ReadPoints may be associated with the same
        Source and a single ReadPoint may be associated with
        multiple Sources.  There will be one row entry for
        each unique pairing of ReadPoint and Source"
    ::= { epcgSources 3 }

epcgRdPntSrcEntry OBJECT-TYPE
    SYNTAX         EpcgRdPntSrcEntry
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
       "Identifies a single pairing of a Read Point and Source."
        INDEX { epcgReadPointIndex,
                epcgSrcIndex }
    ::= { epcgRdPntSrcTable 1 }

EpcgRdPntSrcEntry ::= SEQUENCE {
        epcgRdPntSrcRowStatus           RowStatus
        }

epcgRdPntSrcRowStatus OBJECT-TYPE
    SYNTAX          RowStatus
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
       "The status of this row."
    ::= { epcgRdPntSrcEntry 1 }


--*********************************************************************
--** Notification Channel - Source Table
--*********************************************************************
epcgNotifChanSrcTable OBJECT-TYPE
    SYNTAX            SEQUENCE OF EpcgNotifChanSrcEntry
    MAX-ACCESS        not-accessible
    STATUS            current
    DESCRIPTION
       "This table associates Notification Channels with
        (tag) Sources.  Multiple Notification Channels may be
        associated with the same Source and a single Notification
        Channel may be associated with multiple Sources.  There
        will be one row entry for each unique pairing of
        Notification Channel and Source."
    ::= { epcgSources 4 }

epcgNotifChanSrcEntry OBJECT-TYPE
    SYNTAX            EpcgNotifChanSrcEntry
    MAX-ACCESS        not-accessible
    STATUS             current
    DESCRIPTION
       "Identifies a single pairing of a Notificaiton
```

```
        Channel and Source.

        Entries can be added and removed from
        this table via epcgNotifChanSrcRowStatus in accordance with the
        RowStatus convention."
        INDEX { epcgNotifChanIndex,
                epcgSrcIndex }
    ::= { epcgNotifChanSrcTable 1 }

EpcgNotifChanSrcEntry ::= SEQUENCE {
        epcgNotifChanSrcRowStatus        RowStatus
        }

epcgNotifChanSrcRowStatus OBJECT-TYPE
    SYNTAX               RowStatus
    MAX-ACCESS           read-create
    STATUS               current
    DESCRIPTION
       "The status of this row."
    ::= { epcgNotifChanSrcEntry 1 }

--*****************************************************************
--* Conformance definitions
--*****************************************************************
epcgReaderGroups      OBJECT IDENTIFIER ::= { epcgReaderConformance 1 }
epcgReaderCompliances OBJECT IDENTIFIER ::= { epcgReaderConformance 2 }


--*****************************************************************
--* Compliances statements
--*****************************************************************
epcgReaderStandardCompliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION
        "This define the initial standard implementation requirements
         for an EPCglobal-compliant Reader.

         The following groups are optional.
            epcgReaderGlobalMeasurementsGroup
            epcgReaderAntennaSuppressNotifyGroup
            epcgReaderNotifGroup
            epcgReaderMemoryGroup
            epcgReaderServerInfoGroup
            epcgReaderIOPortGroup
            epcgReaderNotifChannelGroup
            epcgReaderTriggerGroup
            epcgReaderTableAssocGroup
         "
    MODULE      -- This module
        MANDATORY-GROUPS    { epcgReaderDeviceInformationGroup,
                              epcgReaderOperationGroup,
                              epcgReadPointGroup,
                              epcgReaderAntennaReadPointGroup,
                              epcgReaderSourceGroup,
                            epcgReaderMandatoryTableAssocGroup
                              }

--  *** epcgSourceTable ************************************
    OBJECT           epcgSrcReadCyclesPerTrigger
    MIN-ACCESS       read-only
    DESCRIPTION      "Write access is not required."
```

```
    OBJECT          epcgSrcReadDutyCycle
    MIN-ACCESS      read-only
    DESCRIPTION     "Write access is not required."

    OBJECT          epcgSrcReadTimeout
    MIN-ACCESS      read-only
    DESCRIPTION     "Write access is not required."

    OBJECT          epcgSrcGlimpsedTimeout
    MIN-ACCESS      read-only
    DESCRIPTION     "Write access is not required."

    OBJECT          epcgSrcObservedThreshold
    MIN-ACCESS      read-only
    DESCRIPTION     "Write access is not required."

    OBJECT          epcgSrcObservedTimeout
    MIN-ACCESS      read-only
    DESCRIPTION     "Write access is not required."

    OBJECT          epcgSrcLostTimeout
    MIN-ACCESS      read-only
    DESCRIPTION     "Write access is not required."

    ::= { epcgReaderCompliances 1 }

--*****************************************************************
--** Units of conformance
--*****************************************************************
epcgReaderNotifGroup  NOTIFICATION-GROUP
    NOTIFICATIONS { epcgReaderDeviceOperationState,
                    epcgRdrDevMemoryState,
                    epcgReadPointOperationState,
                    epcgReaderAntennaReadFailure,
                    epcgReaderAntennaWriteFailure,
                    epcgReaderAntennaKillFailure,
                    epcgReaderAntennaEraseFailure,
                    epcgReaderAntennaLockFailure,
                    epcgReaderIoPortOperationState,
                    epcgReaderSourceOperationState,
                    epcgReaderNotificationChanOperState
                  }
    STATUS      current
    DESCRIPTION
       "Reader device notifications."
    ::= { epcgReaderGroups 1 }


epcgReaderGlobalMeasurementsGroup OBJECT-GROUP
    OBJECTS     { epcgGlobalCountersData }
    STATUS      current
    DESCRIPTION
        "Reader device global measurements objects."
    ::= { epcgReaderGroups 2 }


epcgReaderDeviceInformationGroup OBJECT-GROUP
    OBJECTS     { epcgRdrDevDescription,
                  epcgRdrDevRole,
                  epcgRdrDevEpc,
                  epcgRdrDevSerialNumber,
                  epcgRdrDevTimeUtc,
```

```
                  epcgRdrDevCurrentSource,
                  epcgRdrDevReboot,
                  epcgRdrDevResetStatistics,
                  epcgRdrDevResetTimestamp,
                  epcgRdrDevNormalizePowerLevel,
                  epcgRdrDevNormalizeNoiseLevel
                }
    STATUS      current
    DESCRIPTION
       "Reader device information objects"
    ::= { epcgReaderGroups 3 }


epcgReaderOperationGroup OBJECT-GROUP
    OBJECTS     { epcgRdrDevOperStatus,
                  epcgRdrDevOperStatusPrior,
                  epcgRdrDevOperStateEnable,
                  epcgRdrDevOperNotifFromState,
                  epcgRdrDevOperNotifToState,
                  epcgRdrDevOperNotifStateLevel,
                  epcgRdrDevOperStateSuppressInterval,
                  epcgRdrDevOperStateSuppressions
                }
    STATUS      current
    DESCRIPTION
       "Reader device operation objects."
    ::= { epcgReaderGroups 4 }


epcgReaderMemoryGroup   OBJECT-GROUP
    OBJECTS     { epcgRdrDevFreeMemory,
                  epcgRdrDevFreeMemoryNotifEnable,
                  epcgRdrDevFreeMemoryNotifLevel,
                  epcgRdrDevFreeMemoryOnsetThreshold,
                  epcgRdrDevFreeMemoryAbateThreshold,
                  epcgRdrDevFreeMemoryStatus,
                  epcgRdrDevMemStateSuppressInterval,
                  epcgRdrDevMemStateSuppressions
                }
    STATUS      current
    DESCRIPTION
        "This mandatory group defines Device-level objects"
    ::= { epcgReaderGroups 5 }


epcgReaderServerInfoGroup OBJECT-GROUP
    OBJECTS     { epcgReaderServerAddressType,
                  epcgReaderServerAddress,
                  epcgReaderServerRowStatus
                }
    STATUS      current
    DESCRIPTION
       "Reader device Server objects."
    ::= { epcgReaderGroups 6 }


epcgReadPointGroup OBJECT-GROUP
    OBJECTS     { epcgReadPointName,
                  epcgReadPointDescription,
                  epcgReadPointAdminStatus,
                  epcgReadPointOperStatus,
                  epcgReadPointOperStateNotifyEnable,
```

```
                  epcgReadPointOperNotifyFromState,
                  epcgReadPointOperNotifyToState,
                  epcgReadPointOperNotifyStateLevel,
                  epcgReadPointOperStatusPrior,
                  epcgReadPointOperStateSuppressInterval,
                  epcgReadPointOperStateSuppressions
                }
    STATUS      current
    DESCRIPTION
        "Reader - Read Point Objects."
    ::= { epcgReaderGroups 7 }

epcgReaderAntennaReadPointGroup  OBJECT-GROUP
    OBJECTS     {  epcgAntRdPntTagsIdentified,
                   epcgAntRdPntTagsNotIdentified,
                   epcgAntRdPntMemoryReadFailures,
                   epcgAntRdPntReadFailureNotifEnable,
                   epcgAntRdPntReadFailureNotifLevel,
                   epcgAntRdPntWriteOperations,
                   epcgAntRdPntWriteFailures,
                   epcgAntRdPntWriteFailuresNotifEnable,
                   epcgAntRdPntWriteFailuresNotifLevel,
                   epcgAntRdPntKillOperations,
                   epcgAntRdPntKillFailures,
                   epcgAntRdPntKillFailuresNotifEnable,
                   epcgAntRdPntKillFailuresNotifLevel,
                   epcgAntRdPntEraseOperations,
                   epcgAntRdPntEraseFailures,
                   epcgAntRdPntEraseFailuresNotifEnable,
                   epcgAntRdPntEraseFailuresNotifLevel,
                   epcgAntRdPntLockOperations,
                   epcgAntRdPntLockFailures,
                   epcgAntRdPntLockFailuresNotifEnable,
                   epcgAntRdPntLockFailuresNotifLevel,
                   epcgAntRdPntPowerLevel,
                   epcgAntRdPntNoiseLevel,
                   epcgAntRdPntTimeEnergized,
                   epcgAntRdPntMemoryReadOperations
                }
    STATUS      current
    DESCRIPTION
       "Reader - Antenna Read Point objects."
    ::= { epcgReaderGroups 8 }


epcgReaderAntennaSuppressNotifyGroup OBJECT-GROUP
    OBJECTS     { epcgAntRdPntReadFailureSuppressInterval,
                epcgAntRdPntReadFailureSuppressions,
                epcgAntRdPntWriteFailureSuppressInterval,
                epcgAntRdPntWriteFailureSuppressions,
                epcgAntRdPntKillFailureSuppressInterval,
                epcgAntRdPntKillFailureSuppressions,
                epcgAntRdPntEraseFailureSuppressInterval,
                epcgAntRdPntEraseFailureSuppressions,
                epcgAntRdPntLockFailureSuppressInterval,
                epcgAntRdPntLockFailureSuppressions
                 }
    STATUS      current
    DESCRIPTION
       "Reader - Antenna Read Point suppress notifications."
    ::= { epcgReaderGroups 9 }
```

```
epcgReaderIOPortGroup   OBJECT-GROUP
    OBJECTS     { epcgIoPortName,
                  epcgIoPortAdminStatus,
                  epcgIoPortOperStatus,
                  epcgIoPortOperStatusNotifEnable,
                  epcgIoPortOperStatusNotifFromState,
                  epcgIoPortOperStatusNotifToState,
                  epcgIoPortOperStatusNotifLevel,
                  epcgIoPortDescription,
                  epcgIoPortOperStatusPrior,
                  epcgIoPortOperStateSuppressInterval,
                  epcgIoPortOperStateSuppressions
                }
    STATUS      current
    DESCRIPTION
       "Reader IO Port objects."
    ::= { epcgReaderGroups 10 }

epcgReaderSourceGroup   OBJECT-GROUP
    OBJECTS     { epcgSrcName,
                  epcgSrcReadCyclesPerTrigger,
                  epcgSrcReadDutyCycle,
                  epcgSrcReadTimeout,
                  epcgSrcGlimpsedTimeout,
                  epcgSrcObservedThreshold,
                  epcgSrcObservedTimeout,
                  epcgSrcLostTimeout,
                  epcgSrcUnknownToGlimpsedTrans,
                  epcgSrcGlimpsedToUnknownTrans,
                  epcgSrcGlimpsedToObservedTrans,
                  epcgSrcObservedToLostTrans,
                  epcgSrcLostToGlimpsedTrans,
                  epcgSrcLostToUnknownTrans,
                  epcgSrcAdminStatus,
                  epcgSrcOperStatus,
                  epcgSrcOperStatusNotifEnable,
                  epcgSrcOperStatusNotifFromState,
                  epcgSrcOperStatusNotifToState,
                  epcgSrcOperStatusNotifyLevel,
                  epcgSrcSupportsWriteOperations,
                  epcgSrcOperStatusPrior,
                  epcgSrcOperStateSuppressInterval,
                  epcgSrcOperStateSuppressions
                }
    STATUS      current
    DESCRIPTION
        "Reader (Tag Data) Source objects."
    ::= { epcgReaderGroups 11 }

epcgReaderNotifChannelGroup      OBJECT-GROUP
    OBJECTS     { epcgNotifChanName,
                  epcgNotifChanAddressType,
                  epcgNotifChanAddress,
                  epcgNotifChanLastAttempt,
                  epcgNotifChanLastSuccess,
                  epcgNotifChanAdminStatus,
                  epcgNotifChanOperStatus,
                  epcgNotifChanOperNotifEnable,
                  epcgNotifChanOperNotifLevel,
                  epcgNotifChanOperNotifFromState,
                  epcgNotifChanOperNotifToState,
```

```
                  epcgNotifChanOperStatusPrior,
                  epcgNotifChanOperStateSuppressInterval,
                  epcgNotifChanOperStateSuppressions
                }
    STATUS      current
    DESCRIPTION
                "Notification Channel objects"
    ::= { epcgReaderGroups 12 }

epcgReaderTriggerGroup  OBJECT-GROUP
    OBJECTS     { epcgTrigName,
                  epcgTrigType,
                  epcgTrigParameters,
                  epcgTriggerMatches,
                  epcgTrigIoPort
                }
    STATUS      current
    DESCRIPTION
       "Reader Trigger objects."
    ::= { epcgReaderGroups 13 }

epcgReaderTableAssocGroup OBJECT-GROUP
    OBJECTS     { epcgNotifTrigRowStatus,
                  epcgReadTrigRowStatus,
                  epcgNotifChanSrcRowStatus
                }
    STATUS      current
    DESCRIPTION
       "Contains objects that maintain associations between rows
        in different tables."
    ::= { epcgReaderGroups 14 }

epcgReaderMandatoryTableAssocGroup OBJECT-GROUP
    OBJECTS     { epcgRdPntSrcRowStatus
                }
    STATUS      current
    DESCRIPTION
       "Contains objects that maintain associations between rows
        in different tables."
    ::= { epcgReaderGroups 15 }

END
```

11 Acknowledgements

Disclaimer

Whilst every effort has been made to ensure that this document and the information contained herein are correct, EPCglobal and any other party involved in the creation of the document hereby state that the document is provided on an "as is" basis without warranty, either expressed or implied, including but not limited to any warranty that the use of the information herein with not infringe any rights, of accuracy or fitness for purpose, and hereby disclaim any liability, direct or indirect, for damages or loss relating to the use of the document.

Below is a list of more active participants and contributors in the development of RM 1.0.1. This list does not acknowledge those who only monitored the process or those who chose not to have their name listed here. Active participants status was granted to those who generated emails, attended face-to-face meetings and conference calls that were associated with the development of this Standard.

First Name	Last Name	Company	Notable Role
Martin	Jackson	Wal-Mart	Working Group Co-Chair
Ricardo	Labiaga	Formerly of Sun Microsystems	Working Group Co-Chair
Mark	Frey	EPCglobal Inc.	Working Group Facilitator
Bud	Biswas	Polaris Networks	
Gerhard	Gangl	7iD (formerly EOSS GmbH)	
Valentina	Shkolnikov	Alien Technology	
Ken	Jett	Cisco	
Craig	Sayers	Hewlett-Packard Co.	
Mark	Ulrich	PepsiCo	
Steven	Shafer	Microsoft	
Anush	Kumar	Microsoft	
Christian	Floerkemeier	Auto-ID Labs	
John	Gallant	Paxar	
Margaret	Wasserman	ThingMagic, LLC	
Lars-Erik	Helander	Intermec	
Debottam	Chatterjee	Polaris Networks	
Charan	Singh	Alien Technology	
Software Development Team		Impinj	

Jason	Yerardi	Paxar	

The following list contains all companies that were opt'd-in to the Reader Management Working Group and have signed the EPCglobal IP Policy.

Company
7iD (formerly EOSS GmbH)
Accenture
Acer Cybercenter Service Inc.
ACNielsen
ACSIS
Adtio Group Limited
Afilias Canada Corp
Albertsons
Alien Technology
Allixon
Altria Group, Inc./Kraft Foods
AMCO TEC International Inc.
Applied Wireless (AWID)
Ark Tech Ltd
AT4 Wireless (formerly Cetecom)
Auto-ID Labs - ADE
Auto-ID Labs - Cambridge
Auto-ID Labs - Fudan University
Auto-ID Labs - ICU
Auto-ID Labs - Japan
Auto-ID Labs - MIT
Auto-ID Labs - Univerisity of St Gallen
AXWAY/formerly Cyclone
BEA Systems
Beijing Futianda Technology Co. Ltd.
British Telecom
CAEN
Cap Gemini Ernst & Young
Certus Warensichenrung-Sys GmbH
Ceyon Technology Co., Ltd

Champtek
Cheng-Loong Corporation
China Elite Technology Co. Ltd
CISC Semiconductor
Cisco
Code Plus, Inc.
Computer Network Info Cntr.
Convergence Sys Ltd
Cybernette
D&S Technology
deister electronic GmbH
Denso Wave Inc
ECO, Inc.
Electronics and Telecommunication Research Institute (ETRI)
EPCglobal Inc.
FEIG Electronic
France Telecom
Fujitsu Ltd
Georgia-Pacific
GlobeRanger
GS1 Australia EAN
GS1 China
GS1 France
GS1 Germany (CCG)
GS1 Hong Kong
GS1 International
GS1 Japan
GS1 Netherlands (EAN.nl)
GS1 South Korea
GS1 Sweden AB (EAN)
GS1 Switzerland
GS1 Taiwan (EAN)
GS1 UK
GS1 US
Hewlett-Packard Co. (HP)

IBM
ID TechEx
Impinj
Indicus Software Pvt Ltd
Industrial Technology Research Institute
Infosys Technologies Limited
Infratab
Institute for Information Industry
Intelleflex
Intermec
Internet Initiative Japan, Inc.
Johnson & Johnson
Kim Hiap Lee Co.
Kimberly-Clark
KL-NET
Korea Computer Servs, Ltd
KTNET - KOREA TRADE NETWORK
LIT (Research Ctr for Logistics Info Tech)
LXE Inc.
Lyngsoe Systems
Lynko Technologies, Inc.
Manhattan Associates
McKesson
MET Labs
Metarights
Metro
Microelectronics Technology, Inc.
Microsoft
Mitsui
Mstar Semiconductor
National Computerization Agency
NCR
NEC Corporation
Nippon Telegraph & Telephone Corp (NTT)
NXP Semiconductors
OatSystems

Omnitrol Networks, Inc.
Oracle Corporation
Panda Logistics Co.Ltd
Pango Networks, Inc.
Patni Computer Systems
Paxar
PepsiCo
Polaris Networks
Pretide Technology, Inc.
Printronix
Procter & Gamble
PSC Scanning
Psion Teklogix Inc.
Q.E.D. Systems
Regal Scan Tech
RetailTech
Reva Systems
RFIT Solutions GmbH
Rush Tracking Systems
Samsung Electronics
Sanion Co Ltd
SAP
Savi Technology
Sedna Systems, Ltd.
SeeBeyond Technology (bought by Sun)
Shipcom Wireless, Inc.
Sirit
Skandsoft Technologies Pvt.Ltd.
Sun Microsystems
Supply Insight, Inc.
T3C Incorporated
TagSys
Tata Consultancy Services
ThingMagic, LLC
Tibco
Tongfang Microelectronics Co.Ltd.

Toppan Printing Co
Toray International, Inc.
TTA Telecommunications Technology Association
Tyco / ADT
Unisys
Ussen Limited Company
Venture Research
VeriSign
Vocollect
Vue Technology
Wal-Mart
Waldemar Winckel GmbH & Co. KG
Wish Unity (formerly Track-IT RFID)
WJ Communications
Yuen Foong Yu Paper

12 References

[RP1] "EPCglobal Reader Protocol, Version 1.1," EPCglobal, http://standars.epcgloalinc.org/RP_1_1

[MIBII] "Management Information Base for Network Management of TCP/IP-based internets: MIB-II,http://www.ietf.org/rfc/rfc1213.txt?number=1213

[ISODir2] ISO, "Rules for the structure and drafting of International Standards (ISO/IEC Directives, Part2, 2001, 4th edition)," July 2002.

[SYSLOG] "The syslog Protocol" IETF draft. http://www.ietf.cnri.reston.va.us/internet-drafts/draft-ietf-syslog-protocol-15.txt

