TECHNICAL SPECIFICATION

ISO/TS 20684-4

First edition
2022-10

Intelligent transport systems — Roadside modules SNMP data interface —

Part 4: Notifications

Systèmes de transport intelligents — Interface de données SNMP pour les modules en bord de route —

Partie 4: Notifications

Reference number
ISO/TS 20684-4:2022(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

A list of all parts in the ISO 20684 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

0.1 Background

The need for standardized communication with ITS field devices is growing around the world. Several countries have adopted Simple Network Management Protocol (SNMP) based field device communication standards.

There is a growing view and empirical evidence that standardizing this activity will result in improved ITS performance, reduced cost, reduced deployment time, and improved maintainability. The ISO 20684 series extends ISO 15784-2 by defining the management information necessary to monitor, configure and control features of field devices. The data elements defined in all parts of ISO 20684 series may be used with any protocol but were designed with an expectation that they would be used with one of the ISO 15784-2 protocols.

By using this approach, agencies can specify open procurements and systems can be expanded geographically in an open and non-proprietary manner, which reduces costs, speeds up deployment, and simplifies integration.

0.2 Overview

SNMP is a collection of well-thought-out and well-proven concepts and principles. SNMP employs the sound principles of abstraction and standardization. This has led to SNMP being widely accepted as the prime choice for communication between management systems and devices on the internet and other communications networks.

The original implementation of SNMP was used to manage network devices such as routers and switches. Since then, the use of SNMP has grown into many areas of application on the internet and has also been used successfully over various serial communications networks.

This document defines management information for ITS field devices following the SNMP conventions.

0.3 Document approach and layout

This document defines:

a) the conformance requirements for this document (Clause 4);

b) a set of user needs for user-defined trigger conditions that can “fire” to initiate actions (Clause 5);

c) a set of detailed requirements for the identified user needs (Clause 6);

d) a set of custom dialogues for notification management (Clause 7);

e) security considerations for the information defined in this document (Clause 8);

f) the management information bases that define the data for the defined requirements (Annex A);

g) the requirements traceability matrix (RTM) that traces the requirements to the design elements (Annex B).

Intelligent transport systems — Roadside modules SNMP data interface —

Part 4: Notifications

1 Scope

Field devices are a key component in intelligent transport systems (ITS). Field devices include traffic signals, message signs, weather stations, traffic sensors, roadside equipment for connected ITS (C-ITS) environments, etc.

Field devices often need to exchange information with other external entities (managers). Field devices can be quite complex, necessitating the standardization of many data concepts for exchange. As such, the ISO 20684 series is divided several individual parts.

This document specifies the needs, requirements and design for the field device to send notifications to one or more managers. It relies upon the definition of triggers as defined in ISO/TS 20684-3.

NOTE 1 There are similarities between certain portions of NTCIP 1103 and this document.

NOTE 2 ISO 20684-1 provides additional details about how the ISO 20684 series relates to the overall ITS architecture.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 20684-1:2021, *Intelligent transport systems — Roadside modules SNMP data interface — Part 1: Overview*

ISO/TS 20684-7:2022, *Intelligent transport systems — Roadside modules SNMP data interface — Part 7: Support features*

IETF RFC 2578, *Structure of Management Information Version 2 (SMIv2), April 1999*

IETF RFC 2579, *Textual Conventions for SMIv2, April 1999*

IETF RFC 2580, *Conformance Statements for SMIv2, April 1999*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 20684-1 apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

4 Conformance

This clause follows the rules defined in ISO 20684-1. Table 1 traces each user need to a set of software features. Table 2 traces each feature to a set of requirements. Table 3 defines terms that are used as predicates in the conformance codes listed in Tables 1 and 2. For a full understanding of these tables and codes, see ISO 20684-1.

Table 1 — User need and feature conformance

Need	Requirement	Conformance
5.1.1: Real-time notifications user need		
	6.2: Notification channel	M
	6.3: Notification event	M
	6.4: Notification factory	M
	6.5: Notification packet	M
	20684-7 6.5: SNMP target	M
	20684-7 6.6: SNMP target parameters	M
	6.1: Notification aggregator	O
	20684-7 6.4: Object group	O

Table 2 — Requirement conformance

Feature	Requirement	Conformance
6.1: Notification aggregator		
	6.1.4.1: Aggregating logic	M
	6.1.4.2: Sending aggregated notifications	M
6.2: Notification channel		
	6.2.2.1: Determine notification channel capabilities	M
	6.2.2.2: Configure a notification channel	M
	6.2.2.3: Verify notification channel configuration	M
	6.2.2.4: Retrieve notification channel statistics	M
	6.2.2.5: Clear notification channel queue	M
	6.2.2.6: Delete a notification channel	M
	6.2.2.7: Toggle enabled status for all notifications	M
	6.2.2.8: Send notifications to target	M
	6.2.3.1: Notification packet size	M
	6.2.4.1: Process incoming data	M
	6.2.4.2: Sending non-queueable notifications	M
	6.2.4.3: Sending queueable notifications	queue:M
	6.2.4.4: Adding messages to the notification queue	queue:M
6.3: Notification event		
	6.3.2: Notification event contents	M
	6.3.3: Maximum data size	M
	6.3.4: Timestamp latency	M
	6.3.5: Timestamp resolution	M
6.4: Notification factory		
	6.4.2.1: Determine notification factory capabilities	M
	6.4.2.2: Configure a notification factory	M
	6.4.2.3: Verify notification manager factory	M

Table 2 *(continued)*

Feature	Requirement	Conformance
	6.4.2.4: Retrieve notification factory statistics	M
	6.4.2.5: Retrieve notification factory status	M
	6.4.2.6: Toggle notification factory	M
	6.4.2.7: Delete notification factory	M
	6.4.3.1.1: Support for unacknowledged notifications	M
	6.4.3.1.2: Support for acknowledged notifications	O
	6.4.3.2.1: Support for non-queueable notifications	M
	6.4.3.2.2: Support for queueable notifications	O
	6.4.3.3.1: Support for non-aggregated notifications	M
	6.4.3.3.2: Support for aggregated notifications	O
	6.4.4: Generate a notification	M
6.5: Notification packet		
	6.5.2.1: Retrieve last notification contents	M
	6.5.3: Notification packet contents	M
	6.5.4.1: Maximum packet size	M
	6.5.4.2: Maximum number of events	M

Table 3 — External standard reference

Predicate	Subclause
queue	6.4.3.2.2

5 User needs

5.1 Monitor user-defined exceptions in real-time

5.1.1 Real-time notifications user need

When user-defined triggers fire, a manager needs to receive real-time notifications containing user-defined information. This will allow a manager to immediately become aware of information that can potentially affect its operation or security without burdening the communications channel with frequent polling for data that seldom changes. Multiple triggers can need to be monitored with different systems being notified based on the type of condition.

EXAMPLE 1 A manager wants to be immediately notified when the cabinet door opens so that an appropriate response can be initiated if the access is unauthorized.

EXAMPLE 2 A manager wants to have the maintenance system notified when the cabinet door opens and have the traffic management system notified when a new message is displayed on a sign.

5.1.2 Monitor user-defined exceptions in real-time overview

5.1.2.1 Required features

In the simplest case, the “real-time notifications” user need shall support the following features

a) A mechanism to call a specific notification factory to generate a new notification event. This document is written based on the assumption that the call will be made by one of the mechanisms specified in ISO/TS 20684-3, which may conclude with a specific action calling a specific notification factory, but this document does not prohibit calling a notification factory by other mechanisms.

b) Notification factory, as defined in this document, which specifies details about the specific notification event and the type of notification packet that should be used to send the notification event.

c) Notification event, which represents the information to be reported to a manager when a call is made to the notification factory.

d) Notification packet, as defined in this document, which represents the contents of a single SNMP notification message, which contains one or more notification events. There are four types of notification packet: one-off traps (not acknowledged), one-off informs (acknowledged), aggregated traps and aggregated informs.

e) Notification channel, as defined in this document, which constructs, manages, and transits notification packets.

f) SNMP target, as specified in ISO/TS 20684-7 and RFC 3413, which defines the SNMP manager to which the notification channel should send notification packets.

g) SNMP target parameters, as specified in ISO/TS 20684-7 and RFC 3413, which define the parameters used to communicate with the target.

5.1.2.2 Aggregate notification option

An implementation may support the notification aggregator, as defined in this document, which manages the aggregation of individual notification events into a single notification packet. Without this feature, all notification packets contain a single notification event and are passed to the notification channel immediately.

5.1.3 Graphical relationships

The relationships among these features are depicted in Figure 1.

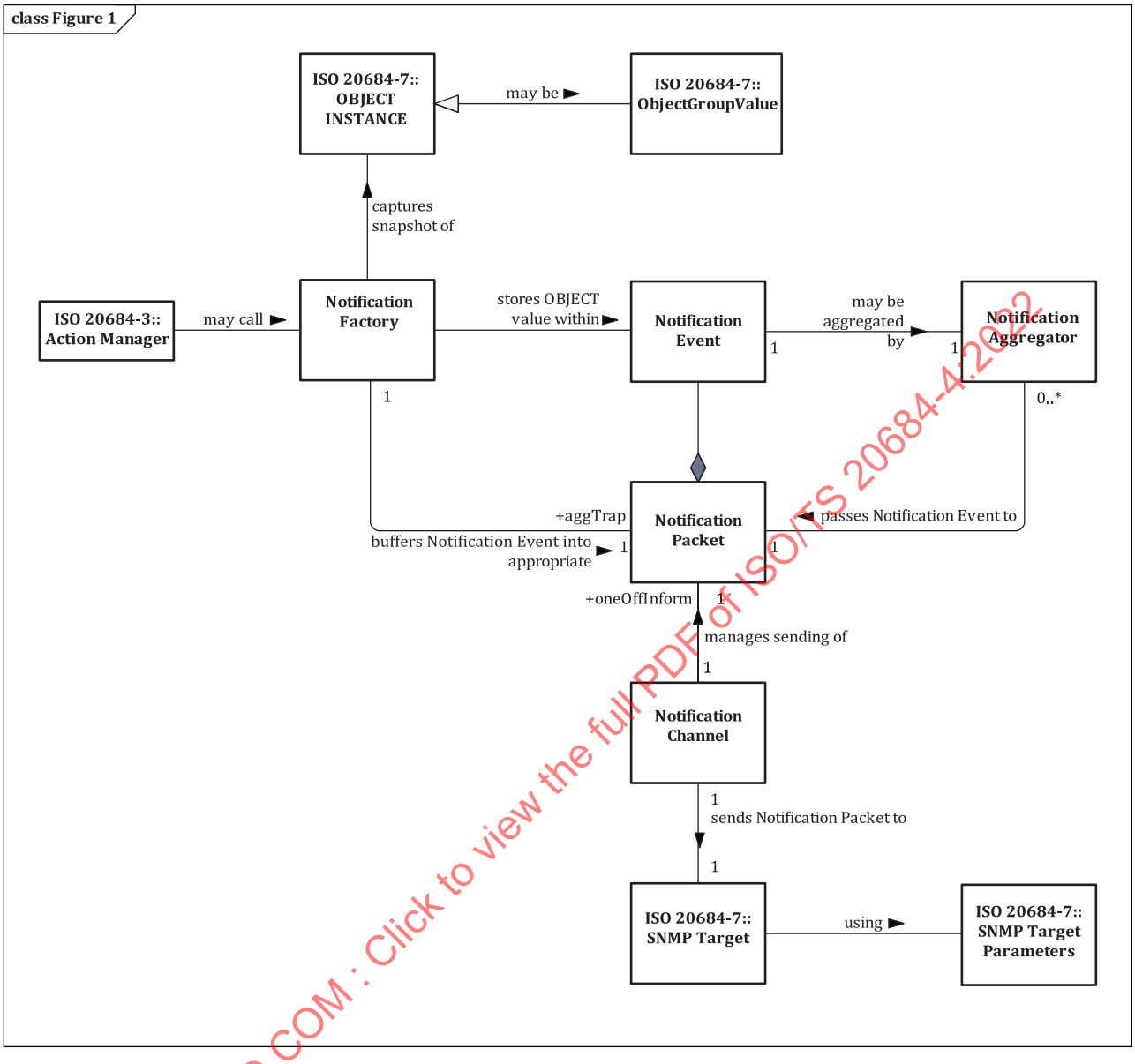

Figure 1 — Conceptual overview of notifications

When a trigger (specified in ISO/TS 20684-3) fires, it calls an action (ISO/TS 20684-3), which may direct the call to the notification factory. When called, the notification factory captures the current value of a specified object and stores this value in a notification event data structure along with an identifier of the condition that initiated the event and timestamp. The object value recorded may be an fdCompositeObjectValue (ISO/TS 20684-7), which can package the values of multiple subordinate objects in a compressed format. The notification factory then passes the Notification Event to the notification channel along with information about how the notification event is to be handled. Multiple notification factories may use the same notification channel.

If the notification event is flagged for aggregation, it is passed to the notification aggregator; otherwise the notification channel assigns the notification event to an appropriate one-off notification packet based on the rules defined in 6.1.4.1.

The optional notification aggregator combines notification events into a single notification packet, which serves to reduce overall communications overhead. The notification aggregator conceptually manages two aggregate notification packets: one that requires acknowledgements (i.e. an SNMP inform) and another that does not (i.e. a SNMP trap). Notification events from different notification

factories can be combined into a single notification packet. Once a notification packet is completed, it is sent back to the notification channel for transmission to the target.

Once a notification packet is ready to send, the notification channel sends the notification packet to the SNMP target while ensuring that the anti-streaming rate is not exceeded, which allows temporary queueing of notifications so that they do not overwhelm the communications channel.

Finally, as per the rules of SNMPv3, trap messages are unacknowledged and inform messages are acknowledged. The acknowledgement process is handled according to standard SNMPv3 rules (complete with timeout and retry logic).

6 Requirements

6.1 Notification aggregator

6.1.1 Notification aggregator definition

The notification aggregator combines multiple notification events into a single packet to reduce overall communication overhead.

6.1.2 Notification aggregator data exchange requirements

No data exchange requirements are defined for the notification aggregator.

6.1.3 Notification aggregator capability requirements

There are no explicit capability requirements for the notification aggregator.

6.1.4 Notification aggregator logic

6.1.4.1 Aggregating logic

Each notification channel shall be associated with its own notification aggregator. The notification aggregator shall store acknowledged and unacknowledged notification events in separate buffers. The aggregation logic for each buffer is independent of the other, but identical as follows.

a) If the number of events in the selected buffer (before adding the new event) is equal to or greater than the maximum configured value, the notification aggregator shall immediately send the aggregated notification according to the rules of 6.1.4.2 and then proceed to proceed to step c) of this subclause. If the number of events is zero (0) the logic proceeds to step c). If the number of events is greater than zero but less than the maximum configured value, the logic proceeds to step b).

b) The notification channel shall determine the length of a serialized notification packet containing the new notification event. If the length of the message exceeds the length of the maximum notification size, the notification aggregator shall immediately send the aggregated notification (omitting the new event) according to the rules of 6.1.4.2 and then proceed to proceed to step c) of this subclause. Otherwise, the logic simply proceeds to step c).

c) The notification aggregator shall add the new notification event to the selected buffer.

d) The notification aggregator shall update its local maximum number of events to aggregate to be the lesser of the previous value and the value associated with the new event.

e) If the number of notification events in the selected buffer (including the newly added event) is equal to or greater than the current local maximum, the notification aggregator shall immediately send the aggregated notification (including the new event) according to the rules of 6.1.4.2; the

process will then be completed. Otherwise, if the maximum configured value has not been reached, proceed to step f).

f) The notification aggregator shall initiate a countdown timer with an initial setting equal to the aggregation time associated with the notification event (and defined by the notification factory) and associate this timer with the selected buffer.

g) Upon the first countdown timer associated with the buffer reaching zero (0), the notification aggregator shall send the aggregated notification according to the rules of 6.1.4.2.

An implementation may choose to implement a single countdown timer that always reflects the shortest duration of any of the theoretical timers mentioned above.

6.1.4.2 Sending aggregated notifications

The notification aggregator shall send an aggregated notification for a selected buffer by completing the following steps.

a) Prepare and send the notification packet according to the rules of 6.2.4.2.

b) Clear all timers associated with the selected buffer.

c) Restore the local maximum number of aggregate events to buffer to the global maximum value.

6.2 Notification channel

6.2.1 Notification channel definition

A notification channel manages the packaging, queueing and transmission of notification packets.

When the notification channel receives a notification event from a notification factory, it determines the appropriate notification packet to use (see 6.2.4.1) and either immediately creates a one-off notification packet or sends the notification event to the notification aggregator.

When a notification packet is ready to send, the notification channel sends the packet while ensuring that the configured anti-streaming rate is not exceeded.

At the top of every minute, the notification channel attempts to send any notification packets that have been queued due to the anti-streaming logic.

Acknowledgements and retries of "inform" notification packets are handled by internal SNMP logic as defined in RFC 3413.

6.2.2 Notification channel data exchange requirements

6.2.2.1 Determine notification channel capabilities

The field device shall allow an SNMP manager to determine the global maximum size of a notification packet that can be transmitted.

6.2.2.2 Configure a notification channel

The field device shall allow a manager to configure the notification channel by specifying the following points.

a) Target that identifies the manager to which the notification channel will send notification packets.

b) Identifier for the manager to recognize the channel.

c) Maximum rate at which notification messages can be sent on the channel (i.e. the anti-streaming rate).

d) Maximum number of notification messages that can be stored in the queue for the channel.

e) Maximum size of a notification packet for the channel.

6.2.2.3 Verify notification channel configuration

The field device shall allow a manager to verify the configuration of the notification channel.

6.2.2.4 Clear notification channel queue

The field device shall allow a manager to clear the queue of notification messages on a notification channel.

6.2.2.5 Retrieve notification channel statistics

The field device shall allow a manager to retrieve statistics for the notification channel, including:

a) the number of notification packets generated, and

b) the number of notification packets dropped.

6.2.2.6 Delete a notification channel

The field device shall allow a manager to delete a notification channel.

6.2.2.7 Toggle enabled status for all notifications

The field device shall allow a manager to disable all notifications or enable all active notifications defined within the fdNotificationsGroup.

6.2.2.8 Send notifications to target

The notification channel shall send notifications to the configured manager based on the logic defined in 6.2.4.

6.2.3 Notification channel capability requirements

6.2.3.1 Notification packet size

The field device shall support notification packets of up to at least 1 023 bytes unless a larger limit is otherwise specified.

6.2.4 Notification transmission logic

6.2.4.1 Process incoming data

When a notification channel receives an incoming notification event from a notification factory, it shall determine the notification mode.

a) If the mode is 'normal' (i.e. not queued and not acknowledged) and has an aggregation size of zero (0), the notification channel shall create a new fdNotificationOneOff trap message containing the NotificationEvent information and transmit the trap according to the rules in 6.2.4.2.

b) If the mode is 'ack' (i.e. not queued and acknowledged) and has an aggregation size of zero (0), the notification channel shall create a new fdNotificationOneOff inform message containing the NotificationEvent information and transmit the inform according to the rules in 6.2.4.2.

c) If the mode is 'queue' (i.e. queued and not acknowledged) and has an aggregation size of zero (0), the notification channel shall create a new fdNotificationOneOff trap message containing the NotificationEvent information and transmit the trap according to the rules in 6.2.4.3.

d) If the mode is 'q_ack' (i.e. queued and acknowledged) and has an aggregation size of zero (0), the notification channel shall create a new fdNotificationOneOff inform message containing the NotificationEvent information and transmit the inform according to the rules in 6.2.4.3.

e) If the mode is 'normal' (i.e. not queued and not acknowledged) and has an aggregation size greater than zero (0), the notification channel shall add the notification event to the channel's non-queueable fdNotificationAggr trap message according to the rules in 6.1.4.

f) If the mode is 'ack' (i.e. not queued and acknowledged) and has an aggregation size greater than zero (0), the notification channel shall add the NotificationEvent to the channel's fdNotificationAggr inform message according to the rules in 6.1.4.

Notification configurations indicating queueable with an aggregation size greater than zero are currently not allowed and shall cause a notification factory rowStatus of 'notReady'.

6.2.4.2 Sending non-queueable notifications

When attempting to send a non-queueable notification, the notification channel shall:

a) increment the notification counter;

b) finalize the serialization of the notification message;

c) clear the notification buffer;

d) delete any associated countdown timers;

e) determine the number of notification messages (traps and informs) sent during this minute;

f) if the number of notification messages sent is equal to or exceeds the anti-streaming rate defined for the notification channel, the newly serialized non-queueable notification shall be dropped and the counter of lost notifications shall be incremented by one. Otherwise, the serialized notification message shall be sent to the SNMP target using its defined parameters.

6.2.4.3 Sending queueable notifications

When attempting to send a queueable notification, the notification channel shall:

a) increment the notification counter;

b) finalize the serialization of the notification message;

c) clear the notification buffer;

d) delete any associated countdown timers;

e) determine the number of notification messages (traps and informs) sent during this minute;

f) if the number of notification messages sent is equal to or exceeds the anti-streaming rate defined for the notification channel, the newly serialized queueable notification shall be placed into the notification queue according to 6.2.4.4; otherwise, the serialized notification message shall be sent to the SNMP target using its defined parameters.

6.2.4.4 Adding messages to the notification queue

When adding a message to the transmission queue, the notification channel shall:

a) determine the number of messages in the queue;

b) if the number of messages in the queue is equal to or exceeds the configured maximum queue depth, delete the oldest notifications in the queue until the number remaining in the queue is less than the maximum queue depth;

c) add the new notification message to the end of the transmission queue;

d) at the top of every minute, determine if there are any notifications in the transmission queue. If so, it shall transmit each notification in the queue, in order from oldest to newest, until either the queue is empty or the anti-streaming rate has been met.

6.3 Notification event

6.3.1 Notification event definition

When a notification factory is called, it generates a new notification event. The notification event is a serialization of information about the event that is deemed to be of interest to a manager. Once the notification event has been serialized, the notification factory passes it to the notification channel for further processing and transmission to the target.

6.3.2 Notification event contents

A notification event shall indicate the identifier of the condition that initiated the creation of the event, a timestamp indicating when the condition was detected, the current value of a managed object that the notification factory has been configured to associate with the event, and a latency indictor indicating when the data was retrieved for the notification event.

NOTE A detected condition can potentially result in activating multiple actions, each of which can require time to perform. The notification event attempts to accurately capture the time of the original event while also indicating the latency in recording the values provided in the notification event.

NOTE 2 It is recommended for managers to be aware that the timestamp only attempts to record the time at which the condition was detected, which can be considerably different than the time at which the condition first became true. For example, an fdCondTriggerEntry (ISO/TS 20684-3) can be configured to only monitor a condition every sixty seconds and only fire after the condition reports true two times in a row. In this case, the timestamp recorded in the notification event could be nearly 2 minutes after the condition first became true (i.e. it could take up to a minute to obtain the first true result and the second true result would require an additional minute). The latency reported in the notification event reflects the change in time after the trigger fires until the data has been retrieved for this specific notification event.

6.3.3 Maximum data size

The field device shall support values of managed objects of at least 400 octets unless a larger limit is otherwise specified.

6.3.4 Timestamp latency

The timestamp recorded in the notification event shall be within one thousand milliseconds (1 s) of the actual time at which the trigger fired in 99,9 % of cases, unless otherwise specified.

6.3.5 Timestamp resolution

The timestamp recorded in the notification data shall use a resolution (from a zero basis) that is identical to or greater than the timestamp latency, with the value rounded down.

EXAMPLE If the timestamp feature has a latency of 200 ms (0,2 s), a trigger that fires at 15:00:00,690 UTC (according to the internal clock) can be timestamped with a value anywhere between 15:00:00,690 and 15:00:00,890. For example, normal processing delays can mean that the function that reads the time reports back 15:00:00,792. This subclause requires that the recorded timestamp uses a resolution that reflects the latency; in other words, in this case, the value is required to be reported in 200 ms steps from an even second (0 point). In this case, it would be reported as 15:00:00,600 (i.e. rounding down to the nearest 200 ms step).

6.4 Notification factory

6.4.1 Notification factory definition

When a trigger fires, it may call a notification factory, which generates a notification event based on its configuration. The notification factory then passes the notification event, along with some of the configuration parameters, to the configured notification channel for additional processing and transmission to the target.

6.4.2 Notification factory data exchange requirements

6.4.2.1 Determine notification factory capabilities

A field device shall allow a manager to determine the capabilities of the notification factory, including:

a) the maximum size of a notification packet;

b) the notification modes supported.

6.4.2.2 Configure a notification factory

The field device shall allow a manager to configure the notification manager by specifying the following:

a) the notification channel to be associated with this factory;

b) the data to be included within generated notification events;

c) whether the generated notification events should be acknowledged;

d) whether the generated notification events should be queued if the channel is blocked due to anti-streaming logic;

e) whether the event can be aggregated and the maximum number of events to allow within the aggregation; and

f) maximum time to wait to aggregate notifications before sending a notification message.

6.4.2.3 Verify notification manager factory

The field device shall allow a manager to verify the configuration of a notification manager.

6.4.2.4 Retrieve notification factory statistics

The field device shall allow a manager to retrieve the number of notification events processed.

6.4.2.5 Retrieve notification factory status

The field device shall allow a manager to retrieve the status of a notification manager.

6.4.2.6 Toggle notification factory

The field device shall allow a manager to toggle the enabled status of a notification manager.

6.4.2.7 Delete notification factory

The field device shall allow a manager to delete a notification manager.

6.4.3 Notification factory capabilities

6.4.3.1 Acknowledgement

The field device shall support the normal delivery notification mode (i.e. queued until link is ready or pending).

6.4.3.1.1 Support for unacknowledged notifications

The field device shall support unacknowledged notifications (i.e. traps).

6.4.3.1.2 Support for acknowledged notifications

The field device shall support acknowledged notifications (i.e. informs).

6.4.3.2 Queueing

6.4.3.2.1 Support non-queueable notifications

The field device shall support non-queueable notifications.

6.4.3.2.2 Support for queueable notifications

The field device shall support queueing non-aggregated notifications.

6.4.3.3 Aggregation

6.4.3.3.1 Support for non-aggregated notifications

The field device shall support non-aggregated notifications.

6.4.3.3.2 Support for aggregated notifications

The field device shall support aggregated notifications with a specified number of queue depth.

6.4.4 Generate a notification

Upon being called, the notification factory shall generate a new instance of a notification event and send it to the configured notification factory for further processing and transmission to the target SNMP manager.

6.5 Notification packet

6.5.1 Notification packet definition

A notification packet is a serialization of one or more notification events to be sent to a target in a single protocol data unit.

6.5.2 Notification packet data exchange requirements

6.5.2.1 Retrieve last notification contents

The field device shall allow a manager to retrieve the contents of the last notification sent.

6.5.3 Notification packet contents

A notification packet shall indicate the notification channel that is sending the event, a concise sequence number for the notifications on that channel so that the manager is able to identify lost notifications, and the notification events that have been assigned to the packet.

6.5.4 Notification packet capability requirements

6.5.4.1 Maximum packet size

The field device shall support notification packets of at least 1 023 octets unless a larger limit is otherwise specified.

6.5.4.2 Maximum number of events

A field device that supports aggregation shall support at least 64 notification events per notification packet unless a larger limit is otherwise specified.

7 Dialogues

7.1 Sending notifications

7.2 Clear notification channel queue

This dialogue provides a process by which the fields within an object group can be cleared. The standardized dialogue shall be as follows.

a) The dialogue shall be initiated by the manager; the logic used by the manager to initiate the dialogue is outside the scope of this document. Prior to initiating the dialogue, the manager shall be aware of which entry (*x.y*) in the table it wishes to use.

b) The manager shall send a GetRequest-PDU for fdNotifyChannelRowStatus.x.y. If the response indicates that fdNotifyChannelRowStatus.x.y does not exist, the dialogue shall terminate.

c) The manager shall send a SetRequest-PDU to set fdNotifyChannelClearQueue.x to 'true'.

8 Security vulnerabilities

There are data elements defined in this document with a MAX-ACCESS clause of read-write and/or read-create. These and other data elements are sensitive and need to be protected from malicious and inadvertent manipulation and/or disclosure. The support for requests in a non-secure environment without proper protection can have a negative effect on network operations. A sampling of the vulnerabilities includes:

a) the ability to change when triggers are fired

b) the ability to delete triggers and thereby disable alerts;

c) the ability to create additional triggers that may flood the network or processor; and

d) the ability to monitor current configurations.

To overcome these vulnerabilities, it is highly recommended that SNMPv3 with TLS support, as defined in RFC 6353, is used to exchange the data.

Annex A
(normative)

Management information base (MIB)

This annex provides definitions which it is useful to import into other management information base (MIB) modules of this document.

A.1 Notification MIB

```
-- ****************************************************************************
-- A.1.1 Notification Header
-- ****************************************************************************
NOTIFICATION-MIB DEFINITIONS ::= BEGIN
IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, OBJECT-IDENTITY, Counter32, Unsigned32,
NOTIFICATION-TYPE
                                                    FROM SNMPv2-SMI
                                                      -- RFC 2578

TruthValue, StorageType, RowStatus
                                                    FROM SNMPv2-TC
                                                      -- RFC 2579

MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
                                                    FROM SNMPv2-CONF
                                                      -- RFC 2580

SnmpAdminString
                                                    FROM SNMP-FRAMEWORK-MIB
                                                      -- RFC 3411

ITSUnsigned16, ITSOerString, fieldDevice, iso20684p4
                                                    FROM FIELD-DEVICE-TC-MIB
                                                      -- ISO 20684-1 Annex A

;
fdNotificationMIB MODULE-IDENTITY
  LAST-UPDATED "202001052026Z"
  ORGANIZATION "ISO TC 204 WG 9"
  CONTACT-INFO
   "name: Kenneth Vaughn
    phone: +1-571-331-5670
    email: kvaughn@trevilon.com
    postal: 6606 FM 1488 RD STE 148-503
            Magnolia, TX 77354
            USA"
  DESCRIPTION
    "This MIB defines a mechanism by which a field device can be configured to
     manage the transmission of notifications to one or more managers in an
    efficient manner."

  REVISION "202001052026Z"
  DESCRIPTION
    "Initial revision of the document as proposed for CD ballot"
  ::= {iso20684p4 1}

-- ****************************************************************************
-- A.1.2 Node Definitions
-- ****************************************************************************

fdNotificationConformance OBJECT-IDENTITY
```

```
  STATUS     current
  DESCRIPTION
    "A node containing conformance statements related to the fdNotificationMIB,
    as defined in ISO/TS 20684-4."
  ::= {fdNotificationMIB 2}

fdNotificationCompliances OBJECT-IDENTITY
  STATUS     current
  DESCRIPTION
    "A node for compliance statements for the fdNotificationMIB."
  ::= {fdNotificationConformance 1}

fdNotificationGroups OBJECT-IDENTITY
  STATUS     current
  DESCRIPTION
    "A node for group definitions related to fdNotificationMIB."
  ::= {fdNotificationConformance 2}

fdNotification OBJECT-IDENTITY
  STATUS     current
  DESCRIPTION
    "A node defining management information related to the field device's
    Notifications."
  ::= {fieldDevice 8}

fdNotificationTypes OBJECT-IDENTITY
  STATUS     current
  DESCRIPTION
    "A node defining notifications for the fdNotificationMIB."
  ::= {fdNotification 0}

-- ****************************************************************************
-- A.1.3 Notifications
-- ****************************************************************************

fdNotificationsEnabled OBJECT-TYPE
  SYNTAX     TruthValue
  MAX-ACCESS read-write
  STATUS     current
  DESCRIPTION
    "An indication of whether the filed device is allowed to generate and send
    notifications. Changing the value to false shall result in all queued
    Notification Packets on all channels to be cleared, all buffered
    Notification Events on all channels to be cleared, and will prevent the
    generation of any new Notification Events from any Notification Factory."
  ::= {fdNotification 1}

fdNotificationsModeSupport OBJECT-TYPE
  SYNTAX     BITS { queueing (1), acknowledgements (2), aggregation (3) }
  MAX-ACCESS read-only
  STATUS     current
  DESCRIPTION
    "An indication of the notification features that the field device supports."
  ::= {fdNotification 2}

fdNotificationsMaxSize OBJECT-TYPE
  SYNTAX     Unsigned32
  UNITS      "octets"
  MAX-ACCESS read-only
  STATUS     current
  DESCRIPTION
    "The maximum size, in octets, that the field device supports for a
    notification."
  ::= {fdNotification 3}

fdNotifyFactoryTable OBJECT-TYPE
  SYNTAX     SEQUENCE OF FdNotifyFactoryEntry
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "A table of notification factories that define how notifications are to be
```

```
    generated and sent."
  ::= {fdNotification 5}

fdNotifyFactoryEntry OBJECT-TYPE
  SYNTAX     FdNotifyFactoryEntry
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "The definition of a notification factory."

  INDEX      {fdNotifyFactoryOwner, fdNotifyFactoryName}
  ::= {fdNotifyFactoryTable 1}

FdNotifyFactoryEntry ::= SEQUENCE {
               fdNotifyFactoryOwner           SnmpAdminString,
               fdNotifyFactoryName            SnmpAdminString,
               fdNotifyFactoryEventID         Unsigned32,
               fdNotifyFactoryChannelOwner    SnmpAdminString,
               fdNotifyFactoryChannelName     SnmpAdminString,
               fdNotifyFactoryObjectContext   SnmpAdminString,
               fdNotifyFactoryObjectID        OBJECT IDENTIFIER,
               fdNotifyFactoryAckEnabled      TruthValue,
               fdNotifyFactoryQueueEnabled    TruthValue,
               fdNotifyFactoryAggregationTime ITSUnsigned16,
               fdNotifyFactoryEventCount      Counter32,
               fdNotifyFactoryStorageType     StorageType,
               fdNotifyFactoryRowStatus       RowStatus }


fdNotifyFactoryOwner OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "The owner of this entry. The exact semantics of this string are subject to
    the security policy defined by the security administrator."
  ::= {fdNotifyFactoryEntry 1}

fdNotifyFactoryName OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "A locally-unique, administratively assigned name for the notification
    factory within the scope of fdNotifyFactoryOwner."
  ::= {fdNotifyFactoryEntry 2}

fdNotifyFactoryEventID OBJECT-TYPE
  SYNTAX     Unsigned32
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "A administratively assigned identifier which allows the SNMP Target
    (manager) to uniquely identify this notification event from all others that
    it might receive from the same notification channel."
  ::= {fdNotifyFactoryEntry 3}

fdNotifyFactoryChannelOwner OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The owner of the fdNotifyChannelEntry to which Notification Events created
    by this Notification Factory will be sent."
  ::= {fdNotifyFactoryEntry 4}

fdNotifyFactoryChannelName OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
```

```
    "The name of the fdNotifyChannelEntry to which Notification Events created
    by this Notification Factory will be sent."
  ::= {fdNotifyFactoryEntry 5}

fdNotifyFactoryObjectContext OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The management context from which to obtain fdNotifyFactoryObjectID."
  ::= {fdNotifyFactoryEntry 6}

fdNotifyFactoryObjectID OBJECT-TYPE
  SYNTAX     OBJECT IDENTIFIER
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The object identifier of the object instance who&#x27;s value is to be included
    in the NotificationEvent. When creating a Notification Event, the value
    shall be obtained using the security credentials of the SNMP Target to
    which this Notification Event is to be sent. If the object instance does
    not exist within the defined MIB view, an SNMP NULL shall be returned
    instead."
  ::= {fdNotifyFactoryEntry 7}

fdNotifyFactoryAckEnabled OBJECT-TYPE
  SYNTAX     TruthValue
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "An indication of whether the SNMP Target will be requested to acknowledge
    the notification. If the value of this object is true, the Notification
    Event shall be transmitted in an SNMP Inform message; otherwise, it will be
    transmitted in an SNMP trap message. If the field device does not support
    notification acknowledgment, any attempt to set this value to true shall
    result in an"
  ::= {fdNotifyFactoryEntry 8}

fdNotifyFactoryQueueEnabled OBJECT-TYPE
  SYNTAX     TruthValue
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "An indication of whether the Notification Channel should queue the
    notification if the channel is in anti-streaming mode when the  notification
    is generated. If the value of this object is true, a Notification Packet
    containing the event shall be transmitted according to the rules of ISO/TS
    20684-4, Section 6.2.4.3; otherwise it shall be transmitted according to the
    rules of ISO/TS 20684-4, Section 6.2.4.2."
  ::= {fdNotifyFactoryEntry 9}

fdNotifyFactoryAggregationTime OBJECT-TYPE
  SYNTAX     IISUnsigned16
  UNITS      "seconds"
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The maximum time, in seconds, that this notification can be held within the
    aggregation buffer prior to transmission. The value of zero shall cause the
    resulting aggregated notification to be immediately transmitted. The value
    of this object shall be ignored unless fdNotifyFactoryAggregationSize is
    greater than zero (0)."
  ::= {fdNotifyFactoryEntry 10}

fdNotifyFactoryEventCount OBJECT-TYPE
  SYNTAX     Counter32
  UNITS      "notification events"
  MAX-ACCESS read-only
  STATUS     current
  DESCRIPTION
    "The number of Notifications Events generated by this entry in the table
```

```
    since this entry in the table was last activated."
  ::= {fdNotifyFactoryEntry 11}

fdNotifyFactoryStorageType OBJECT-TYPE
  SYNTAX     StorageType
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The type of memory used to store this entry in the table."
  ::= {fdNotifyFactoryEntry 12}

fdNotifyFactoryRowStatus OBJECT-TYPE
  SYNTAX     RowStatus
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The status of this conceptual row. Any attempt to modify any other read-
    create object within this conceptual row while the value of this object is
    active (1) shall result in an inconsistentValue error."
  ::= {fdNotifyFactoryEntry 13}

fdNotifyChannelTable OBJECT-TYPE
  SYNTAX     SEQUENCE OF FdNotifyChannelEntry
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "A table containing information on how to generate, manage, and send
    notification packets to a specified SNMP Target."
  ::= {fdNotification 6}

fdNotifyChannelEntry OBJECT-TYPE
  SYNTAX     FdNotifyChannelEntry
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "One row in the fdNotifyChannelTable."

  INDEX      {fdNotifyChannelOwner, fdNotifyChannelName}
  ::= {fdNotifyChannelTable 1}

FdNotifyChannelEntry ::= SEQUENCE {
               fdNotifyChannelOwner           SnmpAdminString,
               fdNotifyChannelName            SnmpAdminString,
               fdNotifyChannelID              ITSUnsigned16,
               fdNotifyChannelTarget          SnmpAdminString,
               fdNotifyChannelQueueDepth      Unsigned32,
               fdNotifyChannelAntiStreamRate  Unsigned32,
               fdNotifyChannelMaxSize         Unsigned32,
               fdNotifyChannelSeqNum          Counter32,
               fdNotifyChannelDroppedCount    Counter32,
               fdNotifyChannelClearQueue      TruthValue,
               fdNotifyChannelStorageType     StorageType,
               fdNotifyChannelRowStatus       RowStatus }


fdNotifyChannelOwner OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "The owner of this entry. The exact semantics of this string are subject to
    the security policy defined by the security administrator."
  ::= {fdNotifyChannelEntry 1}

fdNotifyChannelName OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS not-accessible
  STATUS     current
  DESCRIPTION
    "A locally-unique, administratively assigned name for the notification
    channel within the scope of fdNotifyChannelOwner."
```

```
  ::= {fdNotifyChannelEntry 2}

fdNotifyChannelID OBJECT-TYPE
  SYNTAX     ITSUnsigned16
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "A administratively assigned identifier which allows the SNMP Target
    (manager) to uniquely identify this notification channel from all others
    that it might receive."
  ::= {fdNotifyChannelEntry 3}

fdNotifyChannelTarget OBJECT-TYPE
  SYNTAX     SnmpAdminString (SIZE(1..32))
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The unique identifier associated with the snmpTargetAddrEntry to which the
    notification should be sent."
  ::= {fdNotifyChannelEntry 4}

fdNotifyChannelQueueDepth OBJECT-TYPE
  SYNTAX     Unsigned32
  UNITS      "notification packets"
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The number of Notification Packets that the Notification Channel will allow
      within the queue (i.e., when the anti-streaming rate has been exceeded)."
  ::= {fdNotifyChannelEntry 5}

fdNotifyChannelAntiStreamRate OBJECT-TYPE
  SYNTAX     Unsigned32
  UNITS      "notification packets per minute"
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The maximum number of notification Packets that this Notification Channel
    will send within any minute (from local top of minute to next top of
    minute). Notification Packets received for transmission after this number
    has been reached shall be queued or dropped according to the rules defined
    in ISO/TS 20684-4, Section 6.2.4.2 and 6.2.4.3."
  ::= {fdNotifyChannelEntry 6}

fdNotifyChannelMaxSize OBJECT-TYPE
  SYNTAX     Unsigned32
  UNITS      "octets"
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The maximum size of any notification allowed to be sent on this channel.
    This value should normally be set to the same value as
    fdNotificationsMaxSize, unless the target to which the notifications are to
    be sent supports a lower threshold."
  ::= {fdNotifyChannelEntry 7}

fdNotifyChannelSeqNum OBJECT-TYPE
  SYNTAX     Counter32
  UNITS      "notification packets"
  MAX-ACCESS read-only
  STATUS     current
  DESCRIPTION
    "The number of Notification Packets generated by this Notification Channel
    since the channel was last activated. The sequence number encoded into the
    fdNotificationPacketOneOff or fdNotificationPacketAggr message shall equal
    the lower two octets of this value once it is incremented for the new
    notification."
  ::= {fdNotifyChannelEntry 8}

fdNotifyChannelDroppedCount OBJECT-TYPE
  SYNTAX     Counter32
```

```
  UNITS      "notification packets"
  MAX-ACCESS read-only
  STATUS     current
  DESCRIPTION
    "The number of Notification Packets generated by this Notification Channel
    that have been dropped prior to transmission for any reason since the
    channel was last activated."
  ::= {fdNotifyChannelEntry 9}

fdNotifyChannelClearQueue OBJECT-TYPE
  SYNTAX     TruthValue
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "A command that, when set to true, will clear the anti-streaming queue of
    notifications that are awaiting transmission. The
    fdNotifyChannelDroppedCount shall be incremented for each notification
    cleared from the buffer when this occurs. Clearing the queue will not affect
    the buffers managed by the Notification Aggregator."
  ::= {fdNotifyChannelEntry 10}

fdNotifyChannelStorageType OBJECT-TYPE
  SYNTAX     StorageType
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The storage type used to store the definition of this Notification Channel."
  ::= {fdNotifyChannelEntry 11}

fdNotifyChannelRowStatus OBJECT-TYPE
  SYNTAX     RowStatus
  MAX-ACCESS read-create
  STATUS     current
  DESCRIPTION
    "The status of this conceptual row. Any attempt to modify any read-create
    object within this conceptual row, other than this object and
    fdNotifyChannelClearQueue, while the value of this object is active (1)
    shall result in an inconsistentValue error."
  ::= {fdNotifyChannelEntry 12}

fdNotificationData OBJECT-TYPE
  SYNTAX     ITSOerString
  MAX-ACCESS read-only
  STATUS     current
  DESCRIPTION
    "The data contained in the most recent notification generated by any
    Notification Channel. This object packages data related to (potentially)
    multiple events into a single compact structure. The contents of this OCTET
    STRING shall be defined by the following ASN.1 structure encoded with the
    Octet Encoding Rules (OER).

      FdNotificationPacket ::= SEQUENCE {
        fdNotifyChannelID       INTEGER (0..65535),
        fdNotifyChannelSeqNum   INTEGER (0..65535),
        fdNotifyEvents          FdNotificationEvents }

      FdNotificationEvents ::= SEQUENCE OF FdNotificationEvent

      FdNotificationEvent  ::= SEQUENCE {
        fdNotifyFactoryEventId  INTEGER (0..65535),
        eventTimestamp          ITSDailyTimestamp,
        notificationLatency     ITSUnsigned8,
        data                    FdNotificationData }

      FdNotificationData ::= CHOICE {
        dataValue               OCTET STRING,
        dataError               ITSPduErrorStatus (-128..127)}

    The values for fdNotifyChannelID, fdNotifyChannelSeqNum, and
    fdNotifyFactoryEventId represent the objects with the corresponding names,
    with the sequence number representing the value of the object after the new
```

```
    notification packet is generated.

    The eventTimestamp represents the UTC time-of-day at which the trigger that
    initiated the notification event was fired. The notificationLatency
    represents the base 2 logarithm of the number of milliseconds that elapsed
    between the eventTimeStamp and the time at which the collection of data
    finished, multiplied by 10 and rounded to the nearest integer value. For
    example, if exactly one second (1000 ms) elapsed between the detection of
    the event and the collection of the notification data, the notification
    latency value would be round(log2(1000) * 10) = round(9.966 * 10) =
    round(99.66) = 100. A latency value of 255 represents a latency in excess of
    12.73 hours.

    The dataValue represents the OER encoded value of the object referenced by
    fdNotifyFactoryObjectID as obtained using the fdNotifyFactoryObjectContext
    and the security details associated with the SNMP Target of the Notification
    Channel identified by fdNotifyChannelOwner and fdNotifyChannelName. If the
    request for this data fails using this security information, the data field
    shall contain the dataError value instead to indicate the type of failure."
  ::= {fdNotification 7}

fdNotificationPacket NOTIFICATION-TYPE
    OBJECTS        { fdNotificationData }
    STATUS          current
    DESCRIPTION
        "The notification used by each Notification Channel. The fdNotificationData
    object  shall be populated with the data to be sent with the notification
    immediately prior to generating the notification."
::= { fdNotificationTypes 1 }


-- ****************************************************************************
-- A.1.4 Conformance Information
-- ****************************************************************************

fdNotificationMIBCompliance MODULE-COMPLIANCE
  STATUS     current
  DESCRIPTION
    "The conformance statement for the field device notification MIB."
  MODULE     -- this module
    MANDATORY-GROUPS {
                fdNotifyFactoryCapabilitiesGroup,
                fdNotifyFactoryManagementGroup,
                fdNotifyFactoryConfigurationGroup,
                fdNotifyFactoryStatisticsGroup,
                fdNotifyChannelCapabilitiesGroup,
                fdNotifyChannelConfigurationGroup,
                fdNotifyChannelStatisticsGroup,
                fdNotifyChannelClearGroup,
                fdNotificationEnableGroup,
                fdNotificationDataGroup,
                fdNotificationGroup
                }
  ::= {fdNotificationCompliances 1}

fdNotifyFactoryCapabilitiesGroup OBJECT-GROUP
  OBJECTS    {
             fdNotificationsModeSupport
             }
  STATUS     current
  DESCRIPTION
    "Management information that identifies the capabilities of each
    fdNotifyFactoryEntry."
  REFERENCE "Clause 6.4.2.1"
  ::= {fdNotificationGroups 1}

fdNotifyFactoryManagementGroup OBJECT-GROUP
  OBJECTS    {
             fdNotifyFactoryRowStatus
             }
  STATUS     current
```