INTERNATIONAL STANDARD

ISO 13184-2

First edition
2016-05-15

Intelligent transport systems (ITS) — Guidance protocol via personal ITS station for advisory safety systems —

Part 2: Road guidance protocol (RGP) requirements and specification

Systèmes intelligents de transport — Protocole d'orientation par station ITS personnelle pour systèmes à avis de sécurité —

Partie 2: Spécifications et exigences du protocole d'orientation routière

Reference number
ISO 13184-2:2016(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

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 on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT), see the following URL: Foreword — Supplementary information.

The committee responsible for this document is ISO/TC 204, *Intelligent transport systems*.

ISO 13184 consists of the following parts, under the general title *Intelligent transport systems (ITS) — Guidance protocol via personal ITS station for advisory safety systems*:

— *Part 1: General information and use cases definition*

— *Part 2: Road guidance protocol (RGP) requirements and specification*

Additional part dealing with road guidance protocol (RGP) conformance test specification is under preparation.

Introduction

This part of ISO 13184 specifies the use cases implementation of a real-time decision support system for guidance information, designed to enhance mobility and vehicle safety and to provide a parking guide service using the Personal ITS Station (P-ITS-S). The purpose of the system is to transmit guidance or warning messages to drivers and pedestrians in real-time, enhance the user's convenience and avoid congestion in parking facilities by preventing accidents and enabling easy parking.

This part of ISO 13184 implements the road guidance protocol (RGP) requirements (derived from the use cases defined in ISO 13184–1) based on the Data eXchange Message (DXM) at the application level regarding the safety warning and parking guide services between the Roadside ITS Station (R-ITS-S) installed at the roadside and the user's Personal ITS Station (P-ITS-S), e.g. Nomadic Device.

This part of ISO 13184 covers subjects related to traffic safety, including pedestrians besides vehicle drivers. Therefore, this DXM implementation describes how the safety-related services are provided using the P-ITS-S.

This system is based on the following assumptions.

— Based on the fact that the P-ITS-S has limited resources considers these limitations.

— Use cases related to the safety warning and parking guide service can be classified in various ways. These use cases can be added or deleted frequently depending on the specific circumstances of roads and parking spaces. Therefore, the DXM implementation design needs to be flexible and extendable, which enables to add or delete the use cases conveniently.

— The DXM implementation of road guidance contains data elements to configure the message transmitted between the ITS Stations.

— The major use cases include safety warnings at roads and parking guide services to be used between the R-ITS-S and the P-ITS-S.

Intelligent transport systems (ITS) — Guidance protocol via personal ITS station for advisory safety systems —

Part 2: Road guidance protocol (RGP) requirements and specification

Systèmes intelligents de transport -- Protocole d'orientation par station ITS personnelle pour systèmes à avis de sécurité -- Partie 2

1 Scope

This part of ISO 13184 specifies the road guidance use cases on the DXM to provide the real-time decision support system to drivers or pedestrians using P-ITS-S. The road guidance protocol (RGP) is an instantiation of the data exchange message (DXM), which represents a generic message to exchange data between ITS stations.

The RGP defines an interoperable service protocol between P-ITS-S and R-ITS-S for exchanging data elements. This part of ISO 13184 specifies the following:

— Reference architecture for real-time decision support system.

  This reference architecture provides a general structure for the real-time decision support system and the method of message exchange between the P-ITS-S and the R-ITS-S. This reference architecture is used to build the interconnections between the P-ITS-S and the R-ITS-S.

— Technique of application protocol design for various use cases on a P-ITS-S.

  This technique adopts a flexible and extendable protocol design. In many cases, the application protocol for the ITS is designed to provide a set of messages that is dependent on the use cases and the message exchange method. However, it is not easy to enumerate all use cases for some applications. The use cases can be changed or enhanced frequently. For this type of application, the protocol design, depending on the use cases, is not appropriate. This part of ISO 13184 provides a general technique of designing the road guidance application protocol based on the use cases.

— Primitive data element.

  The primitive data element will be commonly used to configure the safety warning and parking guide service in the form of speed, location and time.

— Use cases at the road and parking spaces for warning and parking guide.

  This part of ISO 13184 describes the use cases applicable to the communication services between the P-ITS-S and the R-ITS-S for the purposes of providing safety warning and parking guidance.

ISO 13184 (all parts) have been aligned according to the requirements specified in ISO 21217, ISO/TS 17419 and ISO/TS 17423.

This part of ISO 13184 only specifies the RGP messages based on the DXM definition (see Annex B and Annex C) at real-time. The content of the RGP messages are based on the definition of road guidance use cases as documented in ISO 13184–1.

This part of ISO 13184 implements ITS-SU objects, which is a general reference to ITS application objects, ITS message sets and other objects which may require globally unique identification and registration.

The management of ITS-SU objects is many-fold, e.g. specified in ISO 24102–4, ISO 24102–5, ISO 24102–6, ISO 24102–7, ISO 24102–8 and ISO 24102–9, and in CEN/ISO/TS 17423. This part of ISO 13184 implements authorized and controlled operation of ITS-SU objects, which requires considerations of ITS-SU object identifiers, i.e. ITS-AID, ITS-MsgSetID, ITS-SUID, ITS-SCUID, addresses and protocol identifiers used in the communication protocol stack of an ITS-S, and others.

NOTE The accuracy of the navigation and positioning system as input to the Road Guidance application is important for road guidance but is not part of the ISO 13184 series. Detailed information about crossroads is needed for implementation of Road Guidance applications.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/TS 17419, *Intelligent transport systems — Cooperative systems — Classification and management of ITS applications in a global context*

ISO/TS 17423, *Intelligent transport systems — Cooperative systems — ITS application requirements and objectives for selection of communication profiles*

ISO 21217, *Intelligent transport systems — Communications access for land mobiles (CALM) — Architecture*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/TS 17419, ISO/TS 17423, ISO 21217 and the following apply.

3.1
FA-SAP
service access point between facilities and application layer

3.2
GPS coordinates
collection of GPS position and time

3.3
GPS position
collection of GPS latitude, longitude and altitude

3.4
ITS-S capability (ITS-S capabilities)
uniquely addressable protocol functionality

3.5
ITS station
ITS-S
entity in a communication network, comprised of application, facilities, networking and access layer components specified in ISO 21217 that operate within a bounded secure management domain

3.6
ITS-S application process
ITS-S AP
element in an ITS station that performs information processing for a particular application, and uses ITS-S services to transmit and receive information

3.7
ITS-S application process provisioner
ITS-S APP
functionality in an ITS-SU offering *ITS-S application processes* (3.6) for download and installation to other ITS-Ss

3.8
personal ITS station
P-ITS-S
implementation of an ITS station as personal ITS subsystem

Note 1 to entry: P-ITS-S is used to send the information of each user (drivers and pedestrians) to the roadside ITS station, and receives the safety warning and parking guide service and transfers them to the users.

3.9
roadside ITS station
R-ITS-S
system that receives and processes vehicular and pedestrian information within a certain zone and determines the situation, in order to provide the safety warning and parking guide service to vehicles and pedestrians

Note 1 to entry: The system is installed at the roadside.

3.10
sensor
device designed to collect general information (e.g., road surface state, potential hazard vehicle's speed) within the server's zone

3.11
WGS-84 coordinate system
WGS-84 is the reference system used in the satellite-based positioning system NAVSTAR Global Positioning System (GPS)

Note 1 to entry: The World Geodetic System (WGS) is a standard for use in cartography, geodesy, and navigation. The latest revision is WGS-84.

4 Abbreviated terms

ADU	application data unit
AP	application process
ASN.1	abstract syntax notation one
BT	Bluetooth
C	conditional
C-ITS-S	central–intelligent transport system–station
CRC	cyclic redundancy check
Cvt	convention (M, O, C)
DER	distinguished encoding rules
DTC	diagnostic trouble code
DXM	data exchange message
FA-SAP	facility application–service access point
GPS	global positioning system
HTML	hypertext mark-up language
ITS	intelligent transport systems
ITS-AID	intelligent transport systems–application identifier
ITS-MsgSetID	intelligent transport systems–message set identifier

ITS-S	intelligent transport systems–station
ITS-SU	intelligent transport systems–station unit
ITS-SCUID	intelligent transport systems–station communication unit identifier
ITS-SUID	intelligent transport systems–station unit identifier
ITS-S AP	intelligent transport systems–ITS-S application process
ITS-S APP	intelligent transport systems–ITS-S application process provisioner
L2CAP	logical link control and adaptation protocol
M	mandatory
ND	nomadic device
O	optional
OBEX	object exchange
OSI	open systems interconnection
PER	packed encoding rules
P-ITS-S	personal–intelligent transport system-station
R-ITS-S	roadside–intelligent transport system-station
RGP	road guidance protocol
SDP	service discovery protocol
SGML	standard generalized mark-up language
UCDF	use case description format
UGP	unified gateway protocol
VIN	vehicle identification number
V-ITS-SG	vehicle–intelligent transport system-station gateway
WGS-84	World Geodetic System 1984

5 Conventions

This part of ISO 13184 is based on the conventions discussed in the OSI Service Conventions (ISO/IEC 10731) as they apply for communication services. The vehicle data transfer protocol is applicable to OSI layers 5, 6 and 7.

6 Road guidance implementation overview

The Road Guidance will be implemented in Data eXchange Message (DXM) which is another way of transmitting data. Other standards directly define ASN.1 elements for every parameter (see ETSI EN 302 637–2, SAE J2735 or ETSI/TS 101 539–1). However, DXM defines data parameters, its types, units, etc. DXM defines flexible messages because every message consists of a message ID (iTSmsID) and a list of registered value identifiers (rvIds) referencing data parameters. New messages can be defined by adding rvIds to the list, new data parameters can be defined by referring data types.

The road guidance DXM implementation considers conditions such as limited resources because the user receives the service from a P-ITS-S. It is undesirable that all use cases be implemented and installed in a light-weight nomadic device, as elements of the road environment, such as crossroads and the parking environment, are very diverse in their forms. In addition, the service environment at crossroads and parking spaces varies, depending on the time and the area, and undergoes a relatively large number of changes. Therefore, use cases may need to be added/modified/deleted. As a result, the nomadic device can work as an obstacle to smooth service provisioning and service expansion, as it has the burden of updating the use cases manually according to its necessity. This part of ISO 13184 proposes a light-weight P-ITS-S that can resolve these challenges.

The DXM handler allows sending of information between two ITS stations [i.e. a personal ITS station sends its user type (pedestrian, vehicle, etc.) and GPS position to a roadside ITS station; a roadside ITS station sends a collision possible message, including its position to the personal ITS station]. The DXM

handler is an ITS-S facilities layer ITS-S capability and an ITS-S application process (see Figure 1). The relation between the APPs on a personal ITS station or the application on a roadside ITS station and a DXM handler is realized through FA-SAP (API). FA-SAP primitives defined in ISO/TS 17429 can be used for this purpose. APPs request the use of the DXM ITS-S capability at time of flow registration. DXM ITS-S capability formats the packets based on RGP defined use cases. As per conformance with ISO/TS 17419 and ISO/TS 17423, add in the overview the process by which a road guidance application (ITS-S application process) gets installed in an ITS station and registered with the ITS station management entity, together with its communication requirements, defined for each type of messages exchanged between the Personal ITS-S, Roadside ITS-S and Central ITS-S should be prepared by ITS-S application process(s) (ITS-S AP).

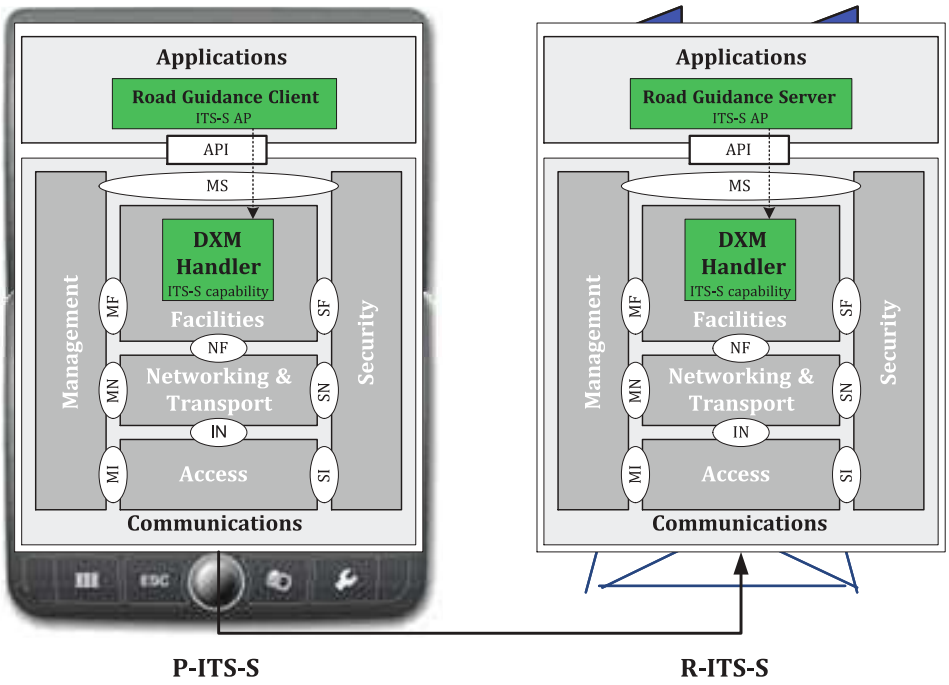
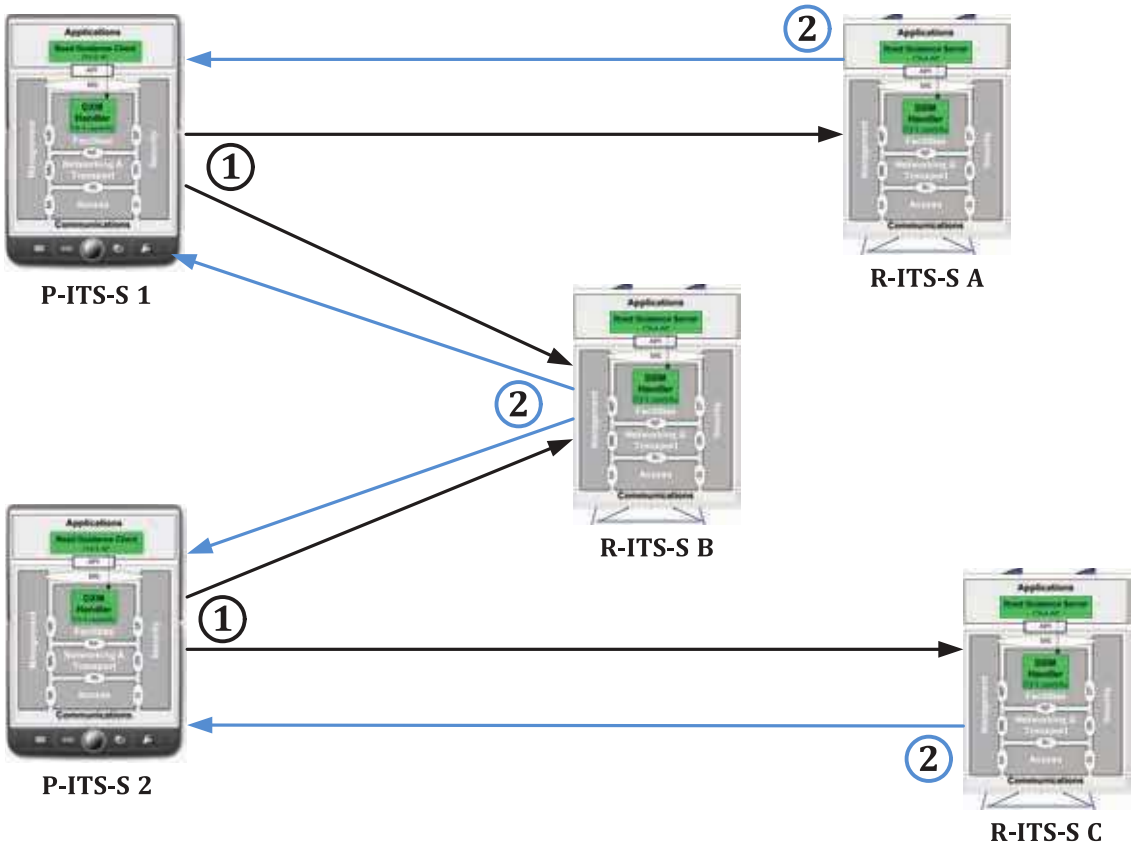


Figure 1 — Communication flow

Figure 2 shows the communication of the road guidance implementation using the DXMessage, i.e. the P-ITS-S starts with "App" handling only messages from and to the R-ITS-Ss. The P-ITS-S is pre-loaded with the DXM configuration using case(s)-specific configuration information from the C-ITS-S (1) to support road guidance messages and data parameters.

After a road user (vehicle, pedestrian) enters a road guidance zone of an R-ITS-S (3), the P-ITS-S immediately reports its position and motion parameters with the necessary message to the R-ITS-S (4). If a situation occurs in the road guidance zone, which matches a supported use case, the R-ITS-S notifies the situation (i.e. safety warning, parking guidance) information to the P-ITS-S (5). The P-ITS-S shows the situation on the display and/or does some acoustic signals. On leaving the road guidance zone, the P-ITS-S stops the communication to the R-ITS-S (6).



Key

1 P-ITS-S sends a broadcast — to inform all possible R-ITS-Ss: “I am interested”

2 R-ITS-Ss send possible problems to all interested stations

Figure 3 — Communication between P-ITS-Ss and R-ITS-Ss

Every RGP message is encoded as DXM. The DXM will be transmitted with an Application Data Unit (ADU) by the road guidance Application Process (AP) using the ITS-S layer facilities.

7 Use cases implementation into the road guidance protocol

7.1 Use case clusters overview

Table 1 provides an overview of the different use cases. The use cases are grouped into use case clusters.

Table 1 — Overview of use case clusters and associated use cases

# Title of use case cluster	Brief description
1 Crossroads with a traffic signal	This cluster is separated into five use cases by taking the signal violating vehicle as a risk factor. A vehicle at a crossroad is controlled by the signalling system of the traffic signal. The use cases presented in this cluster are designed to provide a service for smooth crossroad traffic control while complying with the signalling system and protecting pedestrians on a pedestrian crossing. UC 1.1 — Vehicle violates a signal without stopping UC 1.2 — Violating vehicle is inside the crossroad UC 1.3 — Guiding the pedestrian on a pedestrian crossing UC 1.4 — Pedestrians violate the traffic signal on a pedestrian crossing UC 1.5 — The traffic is bumper-to-bumper on the crossroad
2 Crossroads	This use case considers the scenario that the crossroad is not equipped with a traffic signal, which may cause the traffic congestion if several vehicles enter into the crossroad simultaneously. This use case is designed to provide smooth traffic flow by preventing heavy traffic congestion or reducing waiting time. UC 2.1 — Crossroads without a traffic signal
3 Parking space	This cluster describes two use cases for the parking guide. UC 3.1 — Parking guide when the vehicle enters a parking area UC 3.2 — Searching the path and parking space on demand The parking path guide in the parking space use case refers to simple transmission of the path to the user's personal ITS station, not to the vehicle navigation system. When a vehicle enters into the parking space, it is difficult for the vehicle driver to check how many parking lots are available at which space. Therefore, if the parking space is full or if a parking space is available but cannot be identified conveniently, the driver will waste time to park the vehicle or even unable to find a parking space for the vehicle. In addition, the driver may not be able to park the vehicle at the convenient space.
4 Risky environment alarm	This cluster describes two use cases that consider the frequent accidents area at the curved road. UC 4.1 — Vehicle strays into the path of an oncoming vehicle UC 4.2 — Vehicle approaches the curved road with excessive speed The cluster considers the oncoming vehicle and speed limit regulation. The use cases focus on reducing and preventing the accident which can be caused by the geometric structure of the road. The following two use cases consider the speed limit of the vehicles with some special cases that contain school zone and severe weather condition. UC 4.3 — Risky environments alarm in severe weather condition UC 4.4 — Risky environments alarm in the areas of speed limit enforcement The use cases help safe driving by informing of the presence of the school zone and the severe weather condition.

Table 1 *(continued)*

# Title of use case cluster	Brief description
	The following two use cases consider temporary road occupation scenarios and the situation of an emergency vehicle to establish a clear path. UC 4.5 — Vehicle approaches a temporary road occupation This use case addresses the situation when a vehicle approaches a temporary road occupation such as the road construction, accident/disabled vehicles or obstacles on the road. By informing the status of temporary road occupation, the road congestion will be prevented. UC 4.6 — Emergency vehicle approaches on one's route This use case addresses the situation when an emergency vehicle is moving to establish a clear path. This cluster handles the safety messaging procedure when an emergency vehicle is approaching. By announcing the emergency vehicle approaching information, an emergency vehicle can achieve a clear path.

7.2 Use cases implementation

7.2.1 UC cluster 1 — Crossroads with a traffic signal

7.2.1.1 UC 1.1 — Vehicle violates a signal without stopping

Table 2 defines the use case handling when a vehicle violates a stop signal at the crossroads.

Table 2 — Definition of UC 1.1 — Vehicle violates a signal without stopping

Use Case	**Cluster**	**1 — Crossroads with a traffic signal**		When a vehicle enters a crossroad and cannot stop on the stop line due to the long braking distance and velocity of the vehicle, the warning message is sent to all vehicles and pedestrians. If the vehicle is able to stop on the line, the warning message may disappear.
	Name	**UC 1.1 — Vehicle violates a signal without stopping**		
	Occurrence area	Crossroads with a traffic signal.		
	Road user situation	Accessing the crossroads.		
	Provisioning phase	a) The vehicle accesses the crossroads. b) Forecasts that the vehicle can violate the signal when entering into the crossroads. c) Sends the information message with handling of risk factors as soon as any vehicle poses a risk. d) Sends the release message when the vehicle arrives at the stop line.		
	P-ITS-S control	The following references are related to "Provisioning phase": to c) Caution/Warning message. to d) Release message.		
	Hindrance factor	Signals of the traffic signal should be known in advance to forecast the signal.		
	Requirements	When the user receives the information message, the user must pay attention to it.		
	Reference	Signal violation forecasting of this system is closely related to the driver's response time and braking distance. Therefore, environmental circumstances such as the vehicle type and the state of the road surface may be considered as the major factors to apply this service.		
	Clause	**Name**	**Exe**	**Description**

Table 2 *(continued)*

Messages	9.2	notify-on-position	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	stop-notify	P/V	Stop the communication to the R-ITS-S.
	9.4	collision-possible	R	Send notification message to indicate a vehicle possibly violating a crossing signal, including crossing position and blocked lanes. Send updated message if blocked lanes have changed its status.
	9.15	release	R	Release message if problem no longer occurs.

7.2.1.2 UC 1.2 — Violating vehicle is inside the crossroad

Table 3 defines the use case handling procedure when a vehicle violates the stop signal and locates inside the crossroad. This situation is similar to the use case shown in 7.2.1.1. However, unlike 7.2.1.1, this use case considers the handling procedure when a vehicle has already violated the traffic signal.

Table 3 — Definition of UC 1.2 — Violating vehicle is inside the crossroad

Use Case	**Cluster**	**1 — Crossroads with a traffic signal**		If a vehicle crosses over the stop line and enters inside the crossroad, or the vehicle has violated the stop line and keeps moving even with the stop signal, the warning message is sent to the surrounding vehicles. The warning message is also sent to pedestrians who have received a clear signal at the pedestrian crossing.
	Name	**UC 1.2 — Violating vehicle is inside the crossroad**		
	Occurrence area	Crossroads with a traffic signal.		
	Road user situation	The clear signal is on.		
	Provisioning phase	a) Checks whether a vehicle violates the signal at the crossroad when the traffic signal has changed. b) Generates a caution message that the vehicle violating the traffic signal has been detected. As soon as the signal has changed, the vehicle in violation of the signal is identified and the caution message is sent to the surrounding vehicles and pedestrian which receives the clear signal. c) Generates a release message if the vehicle that has violated the traffic signal exits the crossroad.		
	P-ITS-S control	The following references are related to “Provisioning phase”: to c) Caution/Warning message. to d) Release message.		
	Hindrance factor	Vehicle making a right turn.		
	Requirements	Pay attention to the warning message.		
	Reference	The vehicle attempts to make a right turn and the vehicle in violation of the traffic signal cannot be clearly identified.		
	Clause	**Name**	**Exe**	**Description**
	9.2	notify-on-position	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.

Table 3 *(continued)*

Messages	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.5	`vehi-cle-blocks-road`	R	Send notification message to indicate a vehicle blocks a crossing road, including crossing position and blocked lanes. Send updated message if blocked lanes have been changed.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.1.3 UC 1.3 — Guiding the pedestrian on a pedestrian crossing

Table 4 defines the use case handling procedure if the remaining time of clear signal is short when the pedestrians cross the pedestrian crossing.

Table 4 — Definition of UC 1.3 — Guiding the pedestrian on a pedestrian crossing

Use Case	**Cluster**	**1 — Crossroads with a traffic signal**		If a pedestrian accesses the pedestrian crossing, the remaining time of the clear signal is compared with the average walking time, and the guide message is sent. In addition, if the signalling time at a pedestrian crossing is short, the remaining time of clear signal is notified to the pedestrian at the pedestrian crossing.
	Name	**UC 1.3 — Guiding the pedestrian on a pedestrian crossing**		
	Occurrence area	Crosswalks with a traffic signal.		
	Road user situation	Remaining clear signal time needed.		
	Provisioning phase	a) Checks the remaining time of clear signal. b) Checks the average walking time. c) Generates a guide message that the remaining time of clear signal is notified to the pedestrian. The remaining time of the clear signal is compared with the average walking time, and the guide message is sent. In addition, if the signalling time at a pedestrian crossing is short, the remaining time of clear signal is notified to the pedestrian at the pedestrian crossing.		
	P-ITS-S control	The following references are related to “Provisioning phase”: to b) Caution/Warning message. to c) Guide message.		
	Hindrance factor	Not applicable.		
	Requirements	Pay attention to the guide message.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	No messages can be defined to solve use case			

7.2.1.4 UC 1.4 — Pedestrians violate the traffic signal on a pedestrian crossing

Table 5 defines the use case handling procedure if pedestrians violate a traffic signal on the pedestrian crossing.

Table 5 — Definition of UC 1.4 — Pedestrians violate the traffic signal on a pedestrian crossing

Use Case	**Cluster**	**1 — Crossroads with a traffic signal**		If a pedestrian enters into the pedestrian crossing even though the traffic signal indicates "Stop" or the traffic signal is changing during a pedestrian crossing, the warning message is sent to the surrounding vehicles.
	Name	**UC 1.4 — Pedestrians violate the traffic signal on a pedestrian crossing**		
	Occurrence area	Crosswalks with a traffic signal.		
	Road user situation	Vehicle accesses the crossroads.		
	Provisioning phase	a) Check the existence of pedestrians after the green signal is expired in the pedestrian crossing immediately. b) Generates warning message which indicates the pedestrians are existing. c) Releases the warning message if the pedestrians exit the pedestrian crossing.		
Use Case	**P-ITS-S control**	The following references are related to "Provisioning phase": to b) Caution/Warning message. to c) Release message.		
	Hindrance factor	Not applicable.		
	Requirements	Pay attention to the warning message.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.6	`pedestrian-blocks-road`	R	Send notification message to indicate (a) pedestrians block(s) a crossing road, including crossing position and blocked lanes. Send updated message if blocked lanes have been changed.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.1.5 UC 1.5 — Traffic is bumper-to-bumper on the crossroad

Table 6 defines the use case handling procedure for protecting traffic congestion and improving traffic flow if vehicles try to enter into crossroads when the traffic is bumper-to-bumper.

Table 6 — Definition of UC 1.5 — The traffic is bumper-to-bumper on the crossroad

	Cluster	**1 — Crossroads with a traffic signal**	If traffic is congested and expected to affect vehicles, a guidance message will be announced to affected vehicles to enable them to seek alternative routes that may help improve the overall traffic situation. Clear signal should be changed to red, yellow, and green and absent traffic signals (see EN 12795).
	Name	**UC 1.5 — The traffic is bumper-to-bumper on the crossroad**	
	Occurrence area	Crossroads with a traffic signal.	
	Road user situation	Receiving a clear signal.	

Table 6 *(continued)*

Use Case	**Provisioning phase**	a) Check the traffic congestion along the road and crossroad. b) Generates a guide message if the traffic is bumper-to-bumper immediately. c) Release the guide message if the traffic is smooth.		
	P-ITS-S control	The following references are related to “Provisioning phase”: to b) Guide message. to c) Release message.		
	Hindrance factor	Vehicles which neglect the guide message and enter into the crossroad.		
	Requirements	If vehicles receive the guide message, the vehicles do not enter into the crossroad even though the traffic signal is clear.		
	Reference	Detecting of bumper-to-bumper status refers the share of roads and the speed of vehicles.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.7	`bumper-to-bumper`	R	Send notification message to indicate the traffic is bumper-to-bumper in the crossroads, including crossing position and blocked lanes. Send updated message if blocked lanes have been changed.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.2 UC cluster 2 — Crossroads

7.2.2.1 UC 2.1 — Crossroads without a traffic signal

Table 7 defines the use case handling procedure of the vehicles at the crossroads without a traffic signal.

Table 7 — Definition of UC 2.1 — Crossroads without a traffic signal

Use Case	**Cluster**	**2 — Crossroads**	When several vehicles enter a crossroad simultaneously, information as to which vehicle gets priority is announced to affected vehicles based on the traffic regulations, which enables the traffic flow smoothly by minimizing the waiting time and making the vehicles move consecutively. In addition, to announce the priority, the warning message where the vehicle which violates the regulation is approaching is also notified.
	Name	**UC 2.1 — Crossroads without a traffic signal**	
	Occurrence area	Crossroads without a traffic signal.	
	Road user situation	Approaching the crossroad without a traffic signal.	
	Provisioning phase	a) Checks the vehicle speed and distance between the vehicle and crossroad. b) Checks the priority and announces the priority information immediately as the vehicles cross the crossing. c) Driver receives the message. d) Pass the crossroad following the priority.	

Table 7 *(continued)*

	P-ITS-S control	The following references are related to “Provisioning phase”: to c) Guide message.		
	Hindrance factor	The vehicle which neglects the priority.		
	Requirements	Pass the crossroads depending on the priority information, and follow the updated information if the priority is changing.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.8	`crossing-priority`	R	Send notification message to indicate the crossing priority to the vehicle, including crossing position and priority. If another vehicle passes the crossing, update the priority.

7.2.3 UC cluster 3 — Parking space

7.2.3.1 UC 3.1 — Parking guide when the vehicle enters a parking area

Table 8 defines the use case handling procedure of parking guidance using the parking guide service. Using this use case, the driver can select the preferred parking space, so the vehicle is guided to the parking space.

Table 8 — Definition of UC 3.1 — Parking guide when the vehicle enters a parking area

Use Case	**Cluster**	**3 — Parking space**	When vehicles approach the parking space, available parking spaces depending on the driver’s types (handicapped, inexperienced, etc.) and vehicle types (passenger vehicles, small vehicles, SUVs, buses, etc.) are identified and notified to the driver. After the driver chooses the preferred space among the notified spaces by the guidance, the information regarding the movement path to the designated space is sent to the personal ITS station. The service will be terminated when the driver finds the target parking space and parks the vehicle according to the guided path.
	Name	**UC 3.1 — Parking guide when the vehicle enters a parking area**	
	Occurrence area	Street, parking area.	
	Road user situation	Driving to the designated parking space.	
	Provisioning phase	a) Checks the available parking spaces; chooses the algorithm for selecting the optimal space and announce the route for the specific space. b) Sends guide messages to the driver. c) Completes parking.	
	P-ITS-S control	The following references are related to “Provisioning phase”: to b) Guide message.	
	Hindrance factor	Sensors for notifying the empty parking space have to be working properly. Well defined path calculation algorithm for parking has to be established.	

Table 8 *(continued)*

	Requirements	Pay attention to the selection of preferred space among the notified spaces.		
	Reference	Not applicable.		
Messages	**Chapter**	**Name**	**Exe**	**Description**
	9.16	`search-parking-space`	P/V	Request R-ITS-S to search a parking space with the given requirements.
	9.17	`guide-parking-space`	R/C	Guide the P-ITS-S to the parking space with waypoints and navigation hints.

7.2.3.2 UC 3.2 — Searching the path and parking space on demand

Table 9 defines the use case handling procedure of parking guidance when some obstacles hinder normal movement of the vehicle on the movement path or parking space while the vehicle is travelling to the parking space under the guidance of the personal ITS station.

Table 9 — Definition of UC 3.2 — Searching the path and parking space on demand

Use Case	**Cluster**	**3 — Parking space**	If any vehicle or obstacle blocks the movement path due to abnormal situation in a parking space or if any other factor disturbs normal parking at the designated space, the new parking space and movement path will be sent to the personal ITS station on demand and sent to the personal ITS station, so that the driver can avoid the hindrance.	
	Name	**UC 3.2 — Searching the path and parking space on demand**		
	Occurrence area	Street, parking area.		
	Road user situation	Driving to the designated parking space.		
	Provisioning phase	a) Moves to the designated parking space. b) Checks the obstacle blocking the path of the vehicle. c) Searches a new route and parking slot. d) Introduces the new route and parking slot. e) Completes parking.		
	P-ITS-S control	The following references are related to “Provisioning phase”: to d) Guide message.		
	Hindrance factor	Hinder the driving route and parking position.		
	Requirements	Pay attention to the guide message for new parking slot.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.16	`search-parking-space`	P/V	Request R-ITS-S to search a parking space with the given requirements.
	9.17	`guide-parking-space`	R/C	Guide the P-ITS-S to the parking space with waypoints and navigation hints.

7.2.4 UC cluster 4 — Risky environment alarm

7.2.4.1 UC 4.1 — Vehicle strays into the path of an oncoming vehicle

Table 10 defines the use case handling procedure of warning alarm when there is an oncoming vehicle which strays into the path. Using this use case, the vehicle can pass the curved road in safety by preventing a collision.

Table 10 — Definition of UC 4.1 — Vehicle strays into the path of an oncoming vehicle

Use Case	**Cluster**	**4 — Risky environment alarm**	When vehicles approach a curved road and an oncoming vehicle strays into its path, the warning message is sent to the vehicle. If a driver receives the warning message, he/she can avoid a collision by driving the vehicle to the outside lane or reducing the vehicle speed. The service will be terminated when the driver passes through the curved road.	
	Name	**UC 4.1 — Vehicle strays into the path of an oncoming vehicle**		
	Occurrence area	Curved road.		
	Road user situation	Approaching the curved road.		
	Provisioning phase	a) Checks whether a vehicle approaches the curved road. b) Checks whether there's an oncoming vehicle which strays into the path. c) Generates the caution message that the vehicle strays into the path of an oncoming vehicle has been detected, Guide to drive outside lane and reduce the vehicle speed. d) Sends message which advise to move the vehicle to the outside lane or reduce the vehicle speed.		
	P-ITS-S control	The following references are related to "Provisioning phase": to b) Caution/Warning message.		
	Hindrance factor	Oncoming vehicle is out of control.		
	Requirements	Pay attention to the guide message for existence of the oncoming vehicle.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.9	`oncoming-vehicle`	R	Send notification message to indicate an oncoming vehicle.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.4.2 UC 4.2 — Vehicle approaches the curved road with excessive speed

Table 11 defines the use case handling procedure of warning alarm when a vehicle approaches the curved road and violates the speed limit regulation. Using this use case, the vehicle speed is predetermined and makes the vehicle pass the curved road smoothly.

Table 11 — Definition of UC 4.2 — Vehicle approaches the curved road with excessive speed

Use Case	**Cluster**	**4 — Risky environment alarm**		When vehicles approach a curved road and approaching vehicles violate the speed limit, the warning message is sent to the corresponding vehicle. If a driver receives the warning message, he/she can reduce the vehicle speed, which enables to drive the curved road following by the speed limit regulation. This results in smooth and safe driving in the curved road. The service will be terminated when the driver pass through the curved road.
	Name	**UC 4.2 — Vehicle approaches the curved road with excessive speed**		
Use Case	**Occurrence area**	Curved road.		
	Road user situation	Approaching the curved road.		
	Provisioning phase	a) Checks whether a vehicle violates the speed limit regulation when the vehicle approaches the curved road. b) Generates and sends alarm message to the vehicle. Guide to reduce the vehicle speed. c) Generates a release message when the vehicle's speed reaches the regulation speed.		
	P-ITS-S control	The following references are related to "Provisioning phase": to b) Caution/Warning message.		
	Hindrance factor	Not applicable.		
	Requirements	Pay attention to the speed limit alarm message.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.10	`overspeed`	R	Send notification message to indicate the driver that he is driving too fast. This notification message is generated immediately until the driver reduces the speed or the end of the curve is reached.

7.2.4.3 UC 4.3 — Risky environments alarm in severe weather condition

Table 12 defines the use case handling procedure of warning alarm when the unexpected weather condition occurs. Using this use case, the bad weather condition such as heavy rain, thick fog or freezing road information is sent to a driver in advance, which enables to drive in safety.

Table 12 — Definition of UC 4.3 — Risky environments alarm in severe weather condition

Use Case	**Cluster**	**4 — Risky environment alarm**		When vehicles approach an accident zone in bad weather conditions, the warning message is sent to the vehicle depending on the road and weather condition. If a driver receives the warning message, he/she can recognize the risky environments and prevent the accident by reducing the vehicle's speed or by establishing a safety distance more clearly. The service will be terminated when the driver passes through the designated areas.
	Name	**UC 4.3 — Risky environments alarm in severe weather condition**		
	Occurrence area	Areas of frequent accidents.		
	Road user situation	Approaching the unknown areas with bad road condition.		
Use Case	**Provisioning phase**	a) Checks whether a vehicle approaches the frequent accident zones due to the bad weather conditions. b) Checks the on-road and weather conditions. c) Generates and sends a caution message to the vehicle that it may skid on the ice or it has poor visibility. Guide to reduce the vehicle speed. d) Generates a release message when the vehicle passes through the designated zones.		
	P-ITS-S control	The following references are related to "Provisioning phase": to b) Caution/Warning message.		
	Hindrance factor	Not applicable.		
	Requirements	Pay attention to the guide message for road and weather conditions.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.11	`bad-weather`	R	Send notification message to indicate bad weather.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.4.4 UC 4.4 — Risky environments alarm in the areas of speed limit enforcement

Table 13 defines the use case handling procedure of warning alarm when a vehicle approaches the area of speed limit enforcement. Using this use case, the area of speed limit enforcement such as school zone and temporary speed limit reducing is informed to a driver in advance, which enables to drive in safety and prevent accidents.

Table 13 — Definition of UC 4.4 — Risky environments alarm in the areas of speed limit enforcement

Use Case	**Cluster**	**4 — Risky environment alarm**		When vehicles approach an area where a speed limit is enforced, the warning message is sent to the approaching vehicles. If a driver receives the warning message, he/she can prepare to reduce the speed of vehicle and prevent the accident. The service will be terminated when the driver passes through the designated areas.
	Name	**UC 4.4 — Risky environments alarm in the areas of speed limit enforcement**		
	Occurrence area	Areas of speed limit enforcement.		
	Road user situation	Approaching to the areas of speed limit enforcement.		
	Provisioning phase	a) Checks whether a vehicle approaches to the areas of speed limit enforcement. b) Generates and sends a caution message which indicates the speed limit zone ahead. Guide to reduce the vehicle speed. c) Generates a release message when the vehicle passes through the designated zones.		
Use Case	**P-ITS-S control**	The following references are related to “Provisioning phase”: to b) Caution/Warning message.		
	Hindrance factor	Not applicable.		
	Requirements	Pay attention to the guide message for indicating the speed limit zone.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.12	`speed-limit`	R	Send notification message to indicate driving too fast in a speed enforcement area.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.4.5 UC 4.5 — Vehicle approaches a temporary road occupation

Table 14 defines the use case handling procedure of warning alarm when the temporary road occupation occurs. Using this use case, the temporary road condition such as the road construction ahead, accident ahead or disabled vehicle ahead information is announced to the following vehicles, which enable to drive in safety.

Table 14 — Definition of UC 4.5 — Vehicle approaches a temporary road occupation

Use Case	**Cluster**	**4 — Risky environment alarm**		When an unexpected situation related to road conditions occurs, the warning message is sent to affected vehicles. The message announces information about ongoing road constructions, accidents and disabled vehicles ahead, etc. If a driver receives the warning message, he/she can perceive the road status and prevent the congestion/accident by reducing the vehicle speed or shifting the driving lane. The service will be terminated when the driver passes through the designated areas.
	Name	**UC 4.5 — Vehicle approaches a temporary road occupation**		
	Occurrence area	Congested area.		
	Road user situation	Approaching to the congested unknown areas.		
	Provisioning phase	a) Checks whether a vehicle approaches to the congested unknown area. b) Checks the road conditions such as reported accidents, road construction information and presence of disabled vehicle. c) Generates and sends a guide message to the vehicle depending on the remaining time to be cleared. Guide to avoid road congestion. d) Generates a release message when the occupation is cleared.		
	P-ITS-S control	The following references are related to “Provisioning phase”: to b) Caution/Warning message.		
	Hindrance factor	Not applicable.		
Use Case	**Requirements**	Pay attention to the guide message for the reason of the temporary road occupation.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.13	`road-occupation`	R	Send notification message to indicate a road occupation.
	9.15	`release`	R	Release message if problem no longer occurs.

7.2.4.6 UC 4.6 — Emergency vehicle approaches on one’s route

Table 15 defines the use case handling procedure of warning alarm when an emergency vehicle approaches on one’s route. Using this use case, the vehicle ensures the path of emergency vehicle by following the warning message.

Table 15 — Definition of UC 4.6 — Emergency vehicle approaches on one's route

Use Case	**Cluster**	**4 — Risky environment alarm**		When an emergency vehicle approaches, the warning message is sent to vehicles on its route. If a driver receives the warning message, he/she can prepare to clear the path for the emergency vehicle. This use case enables an emergency vehicle to drive clearly by informing emergency vehicle approaching information in advance. The Service will be terminated after an emergency vehicle passes along the driver's vehicle.
	Name	**UC 4.6 — Emergency vehicle approaches on one's route**		
	Occurrence area	Road.		
	Road user situation	Approaching the emergency vehicle to one's route.		
	Provisioning phase	a) Checks whether an emergency vehicle approaches to one's route. b) Generates and sends a guide message for the designated path to be cleared. Guide for fire lane to be cleared. c) Generates a release message when the emergency vehicle passes along one's route.		
	P-ITS-S control	The following references are related to "Provisioning phase": to b) Caution/Warning message.		
	Hindrance factor	Not applicable.		
	Requirements	Pay attention to the guide message whether the emergency vehicle has passed along or not.		
	Reference	Not applicable.		
Messages	**Clause**	**Name**	**Exe**	**Description**
	9.2	`notify-on-position`	P/V	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S) on entering R-ITS-Ss communication area. After the first contact, send only the P-ITS-S position in a predefined time interval.
	9.3	`stop-notify`	P/V	Stop the communication to the R-ITS-S.
	9.14	`emergency-vehicle`	R	Send notification message to indicate an/some emergency vehicle(s) on the road.
	9.15	`release`	R	Release message if problem no longer occurs.

8 RGP messages

8.1 Overview

Figure 4 shows an overview of the RGP messages.

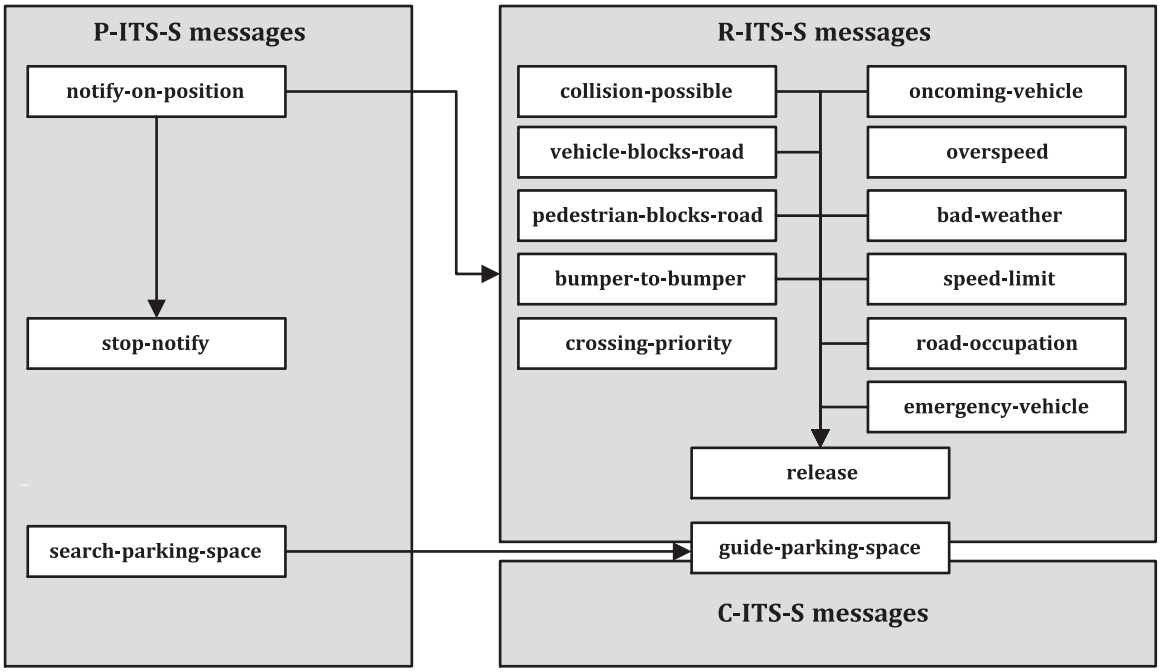


Figure 4 — RGP message overview

8.2 Real-time RGP "Data eXchange Message" communication

Figure 5 shows the real-time RGP communication.

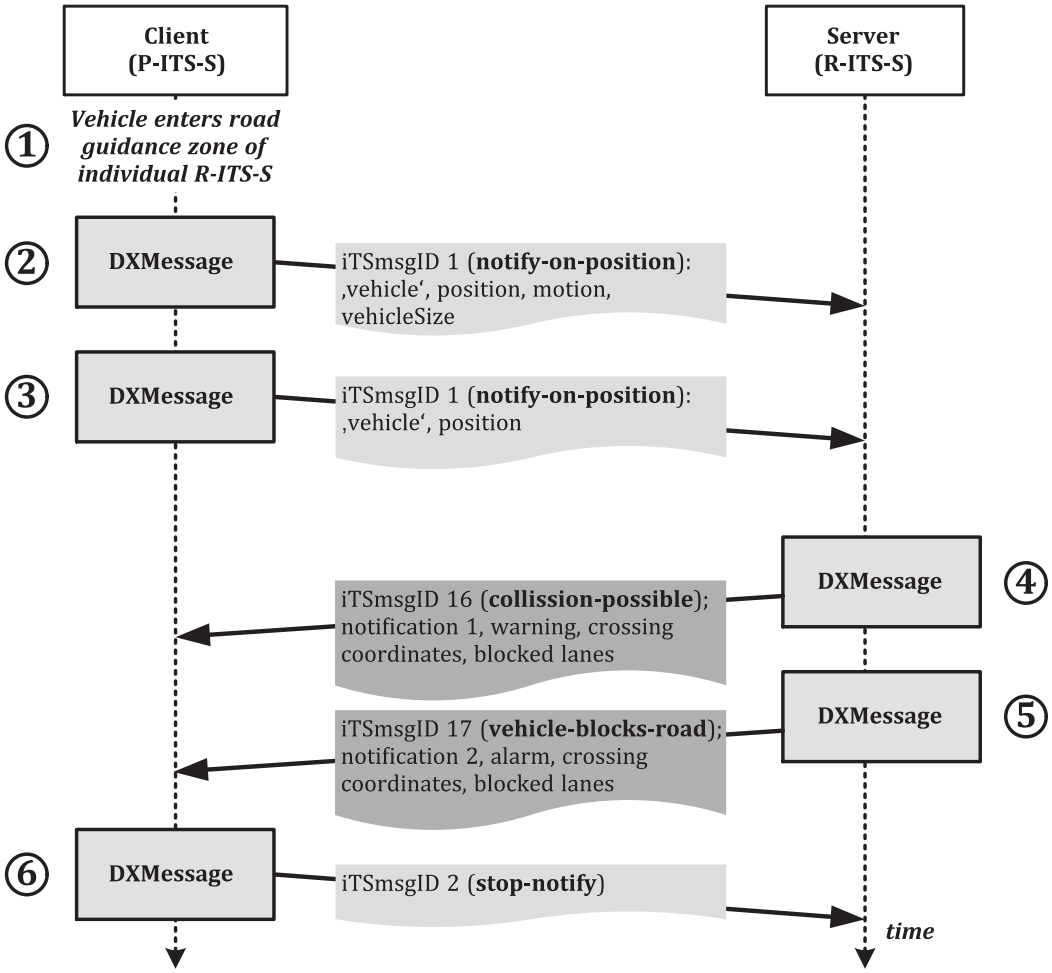


Figure 6 — RGP real-time message flow

9 RGP defined data exchange messages

9.1 General Data eXchange Message definition

The Road Guidance Protocol (RGP) is not a protocol by itself. RGP uses the Data eXchange Message (DXM) format and configuration defined in Annex B. The DXM configuration (DXMConfig) (see B.2) contains the subject (DXMSubject) (see B.3) and all supported DXMsgs (see B.4) of this configuration and uses the Vehicle Interface Data Format (VIDF) to define its data (data parameters, data types, etc.). For a detailed description of the VIDF, see Annex A. The DXMessage (see B.1) contains the runtime data, which only consists of the message identifier (iTmsID), the optional time stamp of the message (timeInMillis), the list of the message parameter values (value) and an optional list of DTC information (dtcInfo). To interpret the DXMessage, the iTmsID must be searched in the DXMConfigs DXMsgs. The corresponding DXMsg contains the list of the data parameters (registered value identifiers). The definition of the data parameters can be found in the VIDF configuration.

In the following subclauses, each RGP use case is defined and combined with an example. The definition contains the DXMsg (DXMConfig) and data parameter (VIDFConfig) definition. The example contains a corresponding runtime message.

9.2 notify-on-position

9.2.1 Definition

Table 16 defines the Data eXchange Message `notify-on-position` to send the P-ITS-S position and motion parameter to the road side server (R-ITS-S).

Table 16 — Definition of notify-on-position

Msg	id	221	Send P-ITS-S position, motion parameter and vehicle size to the road side server (R-ITS-S).	
	name	`notify-on-position`		
	exec	P-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`road-user`	`enumString`	Type of the road user with values `vehicle`, `pedestrian`	M
	`position`	`structure`	Current position of the road user containing:	M
	– `latitude`	– `lnumeric [°N]`	— Geographic longitude of the position, expressed in 1/8th integer micro degrees	M
	– `longitude`	– `lnumeric [°E]`	— Geographic latitude of the position, expressed in 1/8th integer micro degrees	M
	– `elevation`	– `lnumeric [m]`	— Elevation of the position, expressed in metres above the reference ellipsoid (unsigned), offset by 1 km; 0 = 1 km below the reference ellipsoid	O
	`motion`	`structure`	Current motion parameters containing:	C[a]
	– `speed`	– `numeric [m/s]`	— Vehicle speed expressed in signed units of metres per second. Negative values imply the vehicle in moving in reverse.	M
	– `heading`	– `lnumeric [°]`	— Current heading of the road user, expressed in signed units of 0,005 493 247° from North (such that 65,535 such degrees represent 360°). North shall be defined as the axis defined by the WGS-84 coordinate system and its reference ellipsoid. Headings "to the east" are defined as the positive direction.	M
[a] Parameter `motion` is only necessary if roadUser is `vehicle` and it is the first call to this R-ITS-S.				
	– `accel set`	– `structure`	— Acceleration parameters containing:	O
	– `long`	– `numeric [m/s²]`	— Signed acceleration of the road user along the longitude axis in units of 0,01 m/s². A range of over 2 Gs is supported. Accelerations in the directions of forward and to the right are taken as positive.	M
	– `lat`	– `numeric [m/s²]`	— Signed acceleration of the road user along the lateral axis in units of 0,01 m/s².	M
	– `vert`	– `numeric [m/s²]`	— Signed vertical acceleration of the road user along the vertical axis in units of 0,080 m/s². This provides a range of over 1 G in each direction in a one byte value.	M
	– `yaw`	– `lnumeric [°/s]`	— Yaw Rate of the road user, a signed value (to the right being positive) and expressed in 0,01°/s.	M
	`vehicleSize`	`structure`	Current motion parameters containing:	C[a]
	– `length`	– `numeric [m]`	— Vehicle length expressed in metres	M

Table 16 *(continued)*

Data Parameter	— width	— numeric [m]	— Vehicle width expressed in metres	M
	— height	— numeric [m]	— Vehicle height, measured from the ground to the highest surface, excluding any antenna(s); expressed in units of 5 cm. In cases of vehicles with adjustable ride heights, camper shells, and other devices which may cause the overall height to vary, the largest possible height will be used.	M
	— mass	— numeric [kg]	— Vehicle mass with an LSB of 25 kg. Mass should reflect current gross mass of vehicle and contents if known, otherwise an average laden value should be established.	M
[a] Parameter `motion` and `vehicleSize` are only necessary if roadUser is `vehicle` and it is the first call to this R-ITS-S.				
DXMConfig	`msg { { iTSmsID 221, name { textId 10378, longname "notify-on-position" }, type info, executer '1100'B, dataParamList { 1, 2, 6, 22 } }`			
VIDFConfig extract	`dataParam { { rvId 1, name { textId 11200, longname "road user" }, dataTypeId 1, accessType '11000'B, dataParamProperty other }, { rvId 2, name { textId 11229, longname "GPS position" }, dataTypeId 2, accessType '10000'B, dataParamProperty collection }, { rvId 3, name { textId 11021, longname "GPS latitude" }, dataTypeId 3, accessType '10000'B, dataParamProperty sensor }, { rvId 4, name { textId 11022, longname "GPS longitude" }, dataTypeId 4, accessType '10000'B, dataParamProperty sensor }, { rvId 5, name { textId 11023, longname "GPS elevation" }, dataTypeId 5, accessType '10000'B, dataParamProperty sensor }, { rvId 6, name { textId 11201, longname "motion" }, dataTypeId 6, accessType '10000'B, dataParamProperty collection }, ...`			

Table 16 *(continued)*

VIDFConfig extract	{ rvId 8, name { textId 11029, longname "heading direction" }, dataTypeId 8,
	accessType '10010'B, dataParamProperty sensor },
	{ rvId 9, name { textId 11040, longname "Acceleration" }, dataTypeId 9,
	accessType '10000'B, dataParamProperty collection },
	{ rvId 10, name { textId 11041, longname "Acceleration longitudinal" },
	dataTypeId 10, accessType '10000'B, dataParamProperty sensor },
	{ rvId 11, name { textId 11042, longname "Acceleration lateral" }, dataTypeId 10,
	accessType '10000'B, dataParamProperty sensor },
	{ rvId 12, name { textId 11043, longname "Acceleration vertical" }, dataTypeId 10,
	accessType '10000'B, dataParamProperty sensor },
	{ rvId 16, name { textId 11202, longname "yaw" }, dataTypeId 12,
	accessType '10000'B, dataParamProperty sensor },
	{ rvId 22, name { textId 11203, longname "vehicle size" }, dataTypeId 21,
	accessType '10000'B, dataParamProperty collection },
	{ rvId 23, name { textId 11204, longname "length" }, dataTypeId 22,
	accessType '10000'B, dataParamProperty fix },
	{ rvId 24, name { textId 11205, longname "width" }, dataTypeId 23,
	accessType '10000'B, dataParamProperty fix },
	{ rvId 25, name { textId 11206, longname "height" }, dataTypeId 24,
	accessType '10000'B, dataParamProperty fix },
	{ rvId 26, name { textId 11207, longname "mass" }, dataTypeId 25,
	accessType '10000'B, dataParamProperty fix }
	}, dataType {
	{ dataTypeId 1, type enumString: {
	{ value 0, name { textId 10301, longname "unknown" } },
	{ value 1, name { textId 10302, longname "vehicle" } },
	{ value 2, name { textId 10303, longname "pedestrian" } }
	} },
	{ dataTypeId 2, type structure: { { 3, 4, 5 }, convention mandatory } },

Table 16 *(continued)*

VIDFConfig extract	{ dataTypeId 3, type lnumeric: {
	decimalPlaces 5, unitId 67, factor 1, quotient 8000000, addend 0,
	min -720000000, max 720000000 } },
	{ dataTypeId 4, type lnumeric: {
	decimalPlaces 5, unitId 68, factor 1, quotient 8000000, addend 0,
	min -1440000000, max 1440000000 } },
	{ dataTypeId 5, type lnumeric: {
	decimalPlaces 1, unitId 15, factor 1, quotient 10, addend -1000,
	min 0, max 16777215 } },
	{ dataTypeId 6, type structure: { { 481, 8, 9 }, convention conditional } },
	{ dataTypeId 8, type lnumeric: {
	decimalPlaces 2, unitId 42, factor 549, quotient 100000, addend 0,
	min 0, max 65535 } },
	{ dataTypeId 9, type structure: {
	{ 10, 11, 12, 50, 51 }, convention conditional } },
	{ dataTypeId 10, type lnumeric: {
	decimalPlaces 3, unitId 69, factor 1, quotient 1000, addend 0,
	min -160000, max 160000 } },
	{ dataTypeId 12, type lnumeric: {
	decimalPlaces 2, unitId 70, factor 1, quotient 100, addend 0,
	min 0, max 65535 } },
	{ dataTypeId 21, type structure: { { 23, 24, 25, 26 }, convention mandatory } },
	{ dataTypeId 22, type numeric: {
	decimalPlaces 2, unitId 15, factor 1, quotient 100, addend 0,
	min 0, max 16383 } },
	{ dataTypeId 23, type numeric: {
	decimalPlaces 2, unitId 15, factor 1, quotient 100, addend 0,
	min 0, max 1023 } },
	{ dataTypeId 24, type numeric: {
	decimalPlaces 2, unitId 15, factor 1, quotient 20, addend 0,
	min 0, max 255 } },
	{ dataTypeId 25, name { textId 10311, type numeric: {
	decimalPlaces 0, unitId 21, factor 25, quotient 1, addend 0,
	min 0, max 255 } }
	}

9.2.2 Example

Table 17 shows an example for `notify-on-position` containing the road user type “vehicle”, the GPS position of the vehicle (51,065 49 °N, 6,097 81 °E, 166,8 m), the speed of 19,44 m/s driving to the east (45°). The vehicle has the size 4,48 m × 1,78 m × 1,57 m (L × W × H) and a mass of 1 450 kg.

Table 17 — Example for notify-on-position

ASN.1	sendNotifyOnPosition DXMessage ::= { iTSmsID 221, dataParamValue { enumString: 0, – road user = vehicle lnumeric: 408523920, – GPS position.GPS latitude = 51.06549 °N lnumeric: 48782480, – GPS position.GPS longitude = 6.09781 °E lnumeric: 11668, – GPS position.GPS elevation = 166.8 m numeric: 1944, – motion.GPS speed = 19.44 m/s lnumeric: 8192, – motion.heading = 45° structureMissing: 1, – motion.acceleration set = NOT DEFINED (1) numeric: 448, – vehicle size.length = 4.48 m numeric: 178, – vehicle size.width = 1.78 m numeric: 32, – vehicle size.height = 1.60 m numeric: 58 – vehicle size.mass = 1450 kg } } }
U-PER	--sendNotifyOnPosition (42 bytes): 10 00 00 1B A1 64 00 00 0C C2 CC A4 80 60 BA 17 24 03 00 00 5B 28 08 79 80 C0 00 10 00 18 04 10 38 00 80 B2 04 01 00 20 0E 80

9.3 stop-notify

9.3.1 Definition

Table 18 defines the Data eXchange Message `stop-notify` to stop the communication to the road side server (R-ITS-S).

Table 18 — Definition of stop-notify

Msg	id	222		Stop communication to the road side server (R-ITS-S).	
	name	stop-notify			
	exec	P-ITS-S			
Data Parameter	**Name**		**DataType [Unit]**	**Description**	**Cvt**
	— no data parameter —				

DXMConfig	msg {
	{ iTSmsID 222, name { textId 10379, longname "stop-notify" },
	executer '1100'B, dataParamList {}
	}				

9.3.2 Example

Table 19 shows an example for `stop-notify`.

Table 19 — Example for stop-notify

ASN.1	`sendStopNotify DXMessage ::= { iTSmsID 222, dataParamValue {} } }`
U-PER	`--sendStopNotify (6 bytes):` `10 00 00 1B C0 00`

9.4 collision-possible

9.4.1 Definition

Table 20 defines the Data eXchange Message `collision-possible` to notify a possible crossing collision to the road user (P-ITS-S).

Table 20 — Definition of collision-possible

Msg	id	226	Vehicle is about to violate a stop signal without stopping.	
	name	`collision-possible`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`	`numeric []`	Independent number of the notification. The same number is used to update or release an existing notification.	M
	`messageType`	`enumString`	Type of the message with values `alarm`, `warning`, `info`	M
	`problemCoordinate`	`structure`	GPS Position of the centre of the crossing with elements latitude, longitude and elevation (see Table 16; `notify-on-position` parameter `position`).	M
	`crossingDirection {}`	`array` of `structure`	A list of structure elements containing:	M
	— `angle`	— `numeric [°]`	— Angle of the road lane, expressed in signed units of 0,005 493 247° from North (such that 65,535 such degrees represent 360°) measured from the centre of the crossing.	M
	— `lane`	— `numeric`	— Number of the lane on the crossing road. Lanes towards the crossing centre have a positive lane number, lanes off the crossing negative lane numbers. Lanes are counted from the middle of the road to the outside. So the innermost lane towards the crossing centre has the number 1.	M

Table 20 *(continued)*

DXMConfig	msg {
	{ iTSmsID 226, name { textId 10383, shortname "collision-possible",
	longname "vehicle is about to violate a stop signal without stopping" },
	executer '010'B, dataParamList { 28, 29, 2, 31 }
	}

VIDFConfig	dataParam {
	{ rvId 28, name { textId 10419, longname "notification number" }, dataTypeId 27,
	accessType '10010'B, dataParamProperty other },
	{ rvId 29, name { textId 10420, longname "message type" }, dataTypeId 28,
	accessType '10010'B, dataParamProperty other },
	{ rvId 31, name { textId 10422, longname "crossing direction" }, dataTypeId 29,
	accessType '10000'B, dataParamProperty collection },
	},
	dataType {
	{ dataTypeId 27, name { textId 10321, longname "unspecified numeric" },
	type numeric: { decimalPlaces 0, unitId 0, factor 1, quotient 1, addend 0
	} },
	{ dataTypeId 28, name { textId 10322,
	longname "message type (alarm, warning, info)" }, type enumString: {
	{ value 0, name { textId 10323, longname "alarm" } },
	{ value 1, name { textId 10324, longname "warning" } },
	{ value 2, name { textId 10325, longname "info" } }
	} },
	{ dataTypeId 29, name { textId 10326, longname "crossing direction []" },
	type array: 30 },
	{ dataTypeId 30, name { textId 10327, longname "crossing direction {}" },

Table 20 *(continued)*

	type structure: { { 32, 33 }, convention mandatory } },
	{ dataTypeId 32, name { textId 10328, longname "number" }, type numeric: {
	decimalPlaces 0, unitId 0, factor 1, quotient 1, addend 0 } },
	{ dataTypeId 33, name { textId 10329,
	longname "weather condition (rain; snow; ...)" }, type bitString: {
	{ bit 0, name { textId 10330, longname "rain" } },
	{ bit 1, name { textId 10331, longname "snow" } },
	{ bit 2, name { textId 10332, longname "ice" } },
	{ bit 3, name { textId 10333, longname "fog" } },
	{ bit 4, name { textId 10334, longname "strong wind" } }
	} }
	}

9.4.2 Example

Table 21 shows an example for `collision-possible` containing the notification number 1, the message type "warning", the crossing coordinates and one crossing direction with a possible lane blocking.

Table 21 — Example for collision-possible

ASN.1	sendCollisionPossibleNotification DXMessage ::= { version 1, iTSmsID 226, dataParamValue { numeric: 1, - notification number = 1 enumString: 0, - message type = alarm lnumeric: 406265440, - problem coordinate.latitude = 50.78318 °N lnumeric: 49017840, - problem coordinate.longitude = 6.12723 °E lnumeric: 11668, - problem coordinate.elevation = 166.8 m array: 1, - crossing direction = 1 sub element(s) lnumeric: 8192, - crossing direction.[0].angle = 45° numeric: 1 - crossing direction.[0].lane = 1 towards crossing } } }
U-PER	--sendCollisionPossibleNotification (36 bytes): 00 30 00 00 1C 41 00 80 01 20 00 00 66 0D C7 98 03 05 D7 E7 E0 18 00 02 D9 43 80 00 86 00 00 80 00 10 00 20

9.5 vehicle-blocks-road

9.5.1 Definition

Table 22 defines the Data eXchange Message `vehicle-blocks-road` to notify a vehicle blocking the crossing to the road user (P-ITS-S).

Table 22 — Definition of vehicle-blocks-road

Msg	id	227	Vehicle violated traffic signal and is inside the crossroads.	
	name	`vehicle-blocks-road`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`	`numeric []`	Independent number of the notification. The same number is used to update or release an existing notification.	M
	`messageType`	`enumString`	Type of the message with values `alarm`, `warning`, `info`	M
	`problemCoordinate`	`structure`	GPS position of the centre of the crossing with elements latitude, longitude and elevation (see Table 16; `notify-on-position` parameter `position`).	M
	`crossingDirection {}`	`array` of `structure`	A list of crossing directions (see Table 20).	M
DXMConfig	`msg { { iTSmsID 227, name { textId 10384, shortname "vehicle-blocks-road", longname "vehicle violated traffic signal and is inside the crossroads" }, executer '010'B, dataParamList { 28, 29, 30, 31 } }`			

9.5.2 Example

Table 23 shows an example for `vehicle-blocks-road` containing the notification number 1, the message type "alarm", the crossing coordinates and two crossing directions with blocked lanes.

Table 23 — Example for vehicle-blocks-road

ASN.1	sendVehicleBlocksRoadNotification DXMessage ::= { version 1, iTSmsID 227, dataParamValue { numeric: 2, - notification number = 2 enumString: 0, - message type = alarm lnumeric: 406265440, - problem coordinate.latitude = 50.78318 °N lnumeric: 49017840, - problem coordinate.longitude = 6.12723 °E lnumeric: 11668, - problem coordinate.elevation = 166.8 m array: 2, - crossing direction = 2 sub element(s) lnumeric: 8192, - crossing direction.[0].angle = 45° numeric: 1, - crossing direction.[0].lane = 1 towards crossing lnumeric: 24576, - crossing direction.[1].angle = 135° numeric: 3 - crossing direction.[1].lane = 3 towards crossing } } }
U-PER	--sendVehicleBlocksRoadNotification (43 bytes): 00 30 00 00 1C 61 40 80 02 20 00 00 66 0D C7 98 03 05 D7 E7 E0 18 00 02 D9 43 80 01 06 00 00 80 00 10 00 21 80 00 60 00 04 00 18

9.6 pedestrian-blocks-road

9.6.1 Definition

Table 24 defines the Data eXchange Message `pedestrian-blocks-road` to notify a pedestrian blocking the crossing to the road user (P-ITS-S).

Table 24 — Definition of pedestrian-blocks-road

Msg	id	228	Pedestrians violated the traffic signal on a pedestrian crossing.	
	name	pedestrian-blocks-road		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType**	**Description**	**Cvt**
	see collision-possible			

DXMConfig	msg {
	{ iTSmsID 228, name { textId 10385, shortname "pedestri-an-blocks-road",
	longname "pedestrians violated the traffic signal on a pedestrian crossing" },
	executer '010'B, dataParamList { 28, 29, 30, 31 }
	}			

9.6.2 Example

Table 25 shows an example for `pedestrian-blocks-road` analogous to `collision-possible`.

Table 25 — Example for pedestrian-blocks-road

ASN.1	`sendPedestrianBlocksRoadNotification DXMessage ::= { version 1, iTSmsID 228,`
	`  dataParamValue {`
	`  numeric: 3, - notification number = 3`
	`  enumString: 0, - message type = alarm`
	`  lnumeric: 406265440, - problem coordinate.latitude = 50.78318 °N`
	`  lnumeric: 49017840, - problem coordinate.longitude = 6.12723 °E`
	`  lnumeric: 11668, - problem coordinate.elevation = 166.8 m`
	`  array: 2, - crossing direction = 2 sub element(s)`
	`  lnumeric: 8192, - crossing direction.[0].angle = 45°`
	`  numeric: 1, - crossing direction.[0].lane = 1 towards crossing`
	`  lnumeric: 24576, - crossing direction.[1].angle = 135°`
	`  numeric: 3 - crossing direction.[1].lane = 3 towards crossing`
	`} } }`

U-PER	`--sendPedestrianBlocksRoadNotification (43 bytes):`
	`00 30 00 00  1C 81 40 80  03 20 00 00  66 0D C7 98  03 05 D7 E7  E0 18 00 02`
	`D9 43 80 01  06 00 00 80  00 10 00 21  80 00 60 00  04 00 18`

9.7 bumper-to-bumper

9.7.1 Definition

Table 26 defines the Data eXchange Message `bumper-to-bumper` to notify the traffic being bumper-to-bumper on the crossroads to the road user (P-ITS-S).

Table 26 — Definition for bumper-to-bumper

Msg	id	229	Traffic is bumper-to-bumper on the crossroads.	
	name	`bumper-to-bumper`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType**	**Description**	**Cvt**
	see `collision-possible`			

DXMConfig	`msg {`
	`  { iTSmsID 229, name { textId 10386, shortname “bumper-to-bumper”,`
	`  longname “traffic is bumper to bumper on the crossroads” },`
	`  executer ‘010’B, dataParamList { 28, 29, 30, 31 }`
	`}`			

9.7.2 Example

Because the parameters are the same as for `collision-possible`, no example is shown here.

9.8 crossing-priority

9.8.1 Definition

Table 27 defines the Data eXchange Message `crossing-priority` to notify the priority to cross a crossing without traffic signs to the road user (P-ITS-S).

Table 27 — Definition of crossing-priority

Msg	id	231		You can cross the crossing with the given priority.	
	name	`crossing-priority`			
	exec	R-ITS-S			
Data Parameter	**Name**		**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`		`numeric []`	Independent number of the notification. The same number is used to update or release an existing notification.	M
	`messageType`		`enumString`	Type of the message with values `alarm`, `warning`, `info`	M
	`problemCoordinate`		`structure`	GPS Position of the centre of the crossing with elements latitude, longitude and elevation (see Table 16; `notify-on-position` parameter `position`).	M
Data Parameter	`priority`		`numeric []`	A number representing the priority to cross the crossing. 1 = you are the first/next to cross the crossing. 2 = you are the second to cross the crossing.	M
DXMConfig	`msg { { iTSmsID 231, name { textId 10387, longname "crossing-priority" }, executer '010'B, dataParamList { 28, 29, 30, 34 } }`				
VIDFConfig	`dataParam { ... { rvId 34, name { textId 10425, longname "priority" }, dataTypeId 32, accessType write-internal, dataParamProperty other } }, dataType { ... }`				

9.8.2 Example

Table 28 shows an example for `crossing-priority` containing the notification number 1, the message type "info", the crossing coordinates and the priority "2" to cross the crossing.

Table 28 — Example for crossing-priority

ASN.1	`sendCrossingPriorityNotification DXMessage ::= { version 1, iTSmsID 231,`
	`  dataParamValue {`
	`  numeric: 3, - notification number = 3`
	`  enumString: 2, - message type = info`
	`  lnumeric: 406286240, - problem coordinate.latitude = 50.78578 °N`
	`  lnumeric: 49076560, - problem coordinate.longitude = 6.13457 °E`
	`  lnumeric: 11697, - problem coordinate.elevation = 169.7 m`
	`  numeric: 2 - priority = 2`
	`} } }`

U-PER	`--sendCrossingPriorityNotification (29 bytes):`
	`00 30 00 00 1C E0 C0 80 03 20 00 10 66 0D DB E8 03 05 D9 B2 A0 18 00 02`
	`DB 10 40 01 00`

9.9 oncoming-vehicle

9.9.1 Definition

Table 29 defines the Data eXchange Message oncoming-vehicle to notify an oncoming vehicle on the road to the road user (P-ITS-S).

Table 29 — Definition of oncoming-vehicle

Msg	id	232	An oncoming vehicle strays into the path. Please reduce speed and drive to the outermost lane to prevent a collision.	
	name	`oncoming-vehicle`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`	`numeric []`	Independent number of the notification. The same number is used to update or release an existing notification.	M
	`messageType`	`enumString`	Type of the message with values `alarm`, `warning`, `info`	M
	`problemCoordinate`	`structure`	GPS Position of the oncoming vehicle with latitude, longitude and elevation (see Table 16; `notify-on-position` parameter `position`).	

DXMConfig	`msg {`
	`{ iTSmsID 232, name { textId 10388, longname "oncoming-vehicle" },`
	`  executer '010'B, dataParamList { 28, 29, 30 }`
	`}`			

9.9.2 Example

Table 30 shows an example for `oncoming-vehicle` containing the notification number 4, the message type "alarm" and the GPS position of the oncoming vehicle.

Table 30 — Example of oncoming-vehicle

ASN.1	`sendOncomingVehicleNotification DXMessage ::= { version 1, iTSmsID 232,`
	`  dataParamValue {`
	`  numeric: 4,  -- notification number  = 4`
	`  enumString: 0,  -- message type  = alarm`
	`  lnumeric: 406274960, -- problem coordinate.latitude  = 50.78437 °N`
	`  lnumeric: 49033920,  -- problem coordinate.longitude = 6.12924 °E`
	`  lnumeric: 11645  -- problem coordinate.elevation = 164.5 m`
	`} } }`

U-PER	`--sendOncomingVehicleNotification1 (26 bytes):`
	`00 30 00 00  1D 00 A0 80  04 20 00 00  66 0D D0 E4  03 05 D8 65  80 18 00 02`
	`D7 D0`

9.10 overspeed

9.10.1 Definition

Table 31 defines the Data eXchange Message `overspeed` to notify the road user that he is driving too fast (P-ITS-S).

Table 31 — Definition of overspeed

Msg	id	233	You're violating the speed limit regulation. Reduce the speed and be careful in the curve.
	name	`overspeed`	
	exec	R-ITS-S	

Table 31 *(continued)*

	Name	DataType [Unit]	Description	Cvt
Data Parameter	notificationNumber	numeric []	Independent number of the notification. The same number is used to update or release an existing notification.	M
	messageType	enumString	Type of the message with values alarm, warning, info	M
	speedLimit	numeric [m/s]	Speed limit at this position of the road.	M

DXMConfig	msg {
	{ iTSmsID 233, name { textId 10389, longname "overspeed" },
	executer '010'B, dataParamList { 28, 29, 35 }
	}			

VIDFConfig	dataParam { ... { rvId 35, name { textId 10426, longname "speed limit" }, dataTypeId 7, accessType write-internal, dataParamProperty other } }, dataType { ... { dataTypeId 7, name { textId 10309, longname "speed in cm/s" }, type numeric: { decimalPlaces 2, unitId 60, factor 1, quotient 100, addend 0, min -32765, max 32765 } } }			

9.10.2 Example

Table 32 shows an example for overspeed containing the notification number 5, the message type "alarm" and an allowed speed limit of 70 km/h (19,44 m/s).

Table 32 — Example of overspeed

ASN.1	sendOverspeedNotification DXMessage ::= { version 1, iTSmsID 233, dataParamValue { numeric: 5, -- notification number = 5 enumString: 0, -- message type = alarm numeric: 1944 -- speed limit = 19.44 m/s } } }
U-PER	--sendOverspeedNotification (15 bytes): 00 30 00 00 1D 20 60 80 05 20 00 00 21 E6 00

9.11 bad-weather

9.11.1 Definition

Table 33 defines the Data eXchange Message bad-weather to notify a bad weather situation to the road user (P-ITS-S).

Table 33 — Definition of bad-weather

Msg	id	234	Bad weather affects the driving characteristics. Drive slower and be careful.	
	name	bad-weather		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	notificationNumber	numeric []	Independent number of the notification. The same number is used to update or release an existing notification.	M
Data Parameter	messageType	enumString	Type of the message with values alarm, warning, info	M
	problemCoordinate	structure	GPS Position of the next point to the road user having the bad weather problem with elements latitude, longitude and elevation (see Table 16; notify-on-position parameter position).	M
	weatherCondition	bitString	Bitmask supportint following weatherConditions: rain, snow, ice, fog, strong wind	M
	speedSuggestion	numeric [m/s]	Recommended speed limit for the notified weather condition.	O
	remainingDistance	numeric [m]	Remaining distance on same road having the bad weather problem.	O

Table 33 *(continued)*

DXMConfig	msg {
	{ iTSmsID 234, name { textId 10390, longname "bad-weather" },
	executer '010'B, dataParamList { 28, 29, 30, 36, 37, 38 }
	}

VIDFConfig	dataParam { ... { rvId 36, name { textId 10427, longname "weather condition" }, dataTypeId 33, accessType write-internal, dataParamProperty other }, { rvId 37, name { textId 10428, longname "speed suggestion" }, dataTypeId 7, accessType write-internal, dataParamProperty other }, { rvId 38, name { textId 10429, longname "remaining distance" }, dataTypeId 34, accessType write-internal, dataParamProperty other } }, dataType { ... { dataTypeId 7, name { textId 10309, longname "speed in cm/s" }, type numeric: { decimalPlaces 2, unitId 60, factor 1, quotient 100, addend 0, min -32765, max 32765 } }, { dataTypeId 33, name { textId 10329, longname "weather condition" },

VIDFConfig	type bitString: {
	{ bit 0, name { textId 10330, longname "rain" } },
	{ bit 1, name { textId 10331, longname "snow" } },
	{ bit 2, name { textId 10332, longname "ice" } },
	{ bit 3, name { textId 10333, longname "fog" } },
	{ bit 4, name { textId 10334, longname "strong wind" } }
	} },
	{ dataTypeId 34, name { textId 10335, longname "remaining distance in dm" },
	type lnumeric: { decimalPlaces 2, unitId 15, factor 1, quotient 10,
	addend 0, min 0, max 65535 } }
	}

9.11.2 Example

Table 34 shows an example for `bad-weather` containing the notification number 6, the message type "alarm", the next GPS coordinates having bad weather, the weather condition "snow and ice", a speed suggestion of 50 km/h and a remaining weather problem distance of 200 m.

Table 34 — Example of bad-weather

ASN.1	sendBadWeatherNotification DXMessage ::= { version 1, iTSmsID 234, dataParamValue { numeric: 6, -- notification number = 6 enumString: 0, -- message type = alarm lnumeric: 406276320, -- problem coordinate.latitude = 50.78454 °N lnumeric: 49036640, -- problem coordinate.longitude = 6.12958 °E lnumeric: 11647, -- problem coordinate.elevation = 164.7 m bitString: 6, -- weather condition = 6 numeric: 1389, -- speed suggestion = 13.89 m/s lnumeric: 200 -- remaining distance = 20.00 m } } }
U-PER	--sendBadWeatherNotification (36 bytes): 00 30 00 00 1D 41 00 80 06 20 00 00 66 0D D2 38 03 05 D8 7A C0 18 00 02 D7 F2 80 03 02 15 B4 30 00 00 19 00

9.12 speed-limit

9.12.1 Definition

Table 35 defines the Data eXchange Message `speed-limit` to notify the violation of the speed limit to the road user (P-ITS-S).

Table 35 — Definition of speed-limit

Msg	id	235	You are violating the speed limit enforcement. Drive slower and bee careful.	
	name	speed-limit		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	notificationNumber	numeric []	Independent number of the notification. The same number is used to update or release an existing notification.	M
	messageType	enumString	Type of the message with values alarm, warning, info	M
	speedLimit	numeric [m/s]	Speed limit at this position of the road.	M
	remainingDistance	numeric [m]	Remaining distance on same road with the speed limit.	O

DXMConfig	msg {
	{ iTSmsID 235, name { textId 10391, longname "speed-limit" },
	executer '010'B, dataParamList { 28, 29, 35, 38 }
	}			

9.12.2 Example

Table 36 shows an example for `speed-limit` containing the notification number 7, the message type "info", a speed limit of 70 km/h and a remaining distance for this speed limit of 250 m.

Table 36 — Example of speed-limit

ASN.1	`sendSpeedLimitNotification DXMessage ::= { version 1, iTSmsID 235, dataParamValue {`
	`  numeric: 7,  -- notification number  = 7`
	`  enumString: 2,  -- message type  = info`
	`  numeric: 1944,  -- speed limit  = 19.44 m/s`
	`  lnumeric: 250  -- remaining distance  = 25.00 m`
	`} } }`

U-PER	`--sendSpeedLimitNotification (19 bytes):` `00 30 00 00  1D 60 80 80  07 20 00 10  21 E6 03 00  00 01 F4`

9.13 road-occupation

9.13.1 Definition

Table 37 defines the Data eXchange Message `road-occupation` to notify a road occupation to the road user (P-ITS-S).

Table 37 — Definition of road-occupation

Msg	id	236	A road occupation is on the road. Please reduce speed and use the suggested lanes.	
	name	`road-occupation`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`	`numeric []`	Independent number of the notification. The same number is used to update or release an existing notification.	M
	`messageType`	`enumString`	Type of the message with values `alarm`, `warning`, `info`	M
	`problemCoordinate`	`structure`	GPS Position of the road occupation with elements latitude, longitude and elevation (see Table 16; `notify-on-position` parameter `position`).	M
	`occupationType`	`enumString`	Type of the occupation with values `road-construction`, `accident`, `disabled-vehicle`	M
	`speedLimit`	`numeric [m/s]`	Speed limit at the occupation.	O
	`remainingDistance`	`numeric [m]`	Remaining distance of the occupation on same road.	O

Table 37 *(continued)*

	occupiedLane {}	numeric []	Numbers of occupied lanes. Lanes in the same direction as the road user have a positive lane number. On the other hand, lanes in the opposite direction have negative lane numbers. Lanes are counted from the middle of the road to the outermost lane.	O

DXMConfig	msg {
	{ iTSmsID 236, name { textId 10392, longname "road-occupation" },
	executer '010'B, dataParamList { 28, 29, 30, 39, 35, 38, 45 }
	}			

VIDFConfig	dataParam {...
	{ rvId 39, name { textId 10430, longname "occupation type" }, dataTypeId 35,
	accessType write-internal, dataParamProperty other },
	{ rvId 45, name { textId 10436, longname "occupied lane" }, dataTypeId 40,
	accessType write-internal, dataParamProperty other }
	}, dataType {...
	{ dataTypeId 35, name { textId 10336, longname "" }, type enumString: {
	{ value 0, name { textId 10337, longname "road construction" } },
	{ value 1, name { textId 10338, longname "accident" } },
	{ value 2, name { textId 10339, longname "disabled vehicle" } }
	} },
	{ dataTypeId 40, name { textId 10354, longname "occupied lane []" },
	type array: 32 }
	}			

9.13.2 Example

Table 38 shows an example for road-occupation containing the notification number 8, the message type "warning", the position of the road occupation, the type of the road occupation ("road-construction"), the speed limit inside the road occupation of 70 km/h, the remaining distance of the road occupation and the occupied lane 2.

Table 38 — Example of road-occupation

ASN.1	`sendRoadOccupationNotification DXMessage ::= { version 1, iTSmsID 236,`
	`  dataParamValue {`
	`  numeric: 8, -- notification number = 8`
	`  enumString: 1, -- message type = warning`
	`  lnumeric: 406276320, -- problem coordinate.latitude = 50.78454 °N`
	`  lnumeric: 49036640, -- problem coordinate.longitude = 6.12958 °E`
	`  lnumeric: 11647, -- problem coordinate.elevation = 164.7 m`
	`  enumString: 1, -- occupation type = accident`
	`  numeric: 1944, -- speed limit = 19.44 m/s`
	`  lnumeric: 1400, -- remaining distance = 140.00 m`
	`  array: 1, -- occupied lane = 1 sub element(s)`
	`  numeric: 2 -- occupied lane.[0] = 2`
	`} } }`

U-PER	`--sendRoadOccupationNotification (41 bytes):`
	`00 30 00 00  1D 81 40 80  08 20 00 08  66 0D D2 38  03 05 D8 7A  C0 18 00 02`
	`D7 F2 00 00  82 1E 60 30  00 00 AF 07  00 01 04 00  10`

9.14 emergency-vehicle

9.14.1 Definition

Table 39 defines the Data eXchange Message emergency-vehicle to notify (an) emergency vehicle(s) on the same road to the road user (P-ITS-S).

Table 39 — Definition of emergency-vehicle

Msg	id	237	A/Some emergency vehicle(s) is/are on the road. Be careful and drive to the suggested direction.	
	name	`emergency-vehicle`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`	`numeric []`	Independent number of the notification. The same number is used to update or release an existing notification.	M
	`messageType`	`enumString`	Type of the message with values `alarm`, `warning`, `info`	M
	`problemCoordinate`	`structure`	GPS Position of the emergency vehicle(s) with elements latitude, longitude and elevation (see Table 16; `notify-on-position` parameter `position`).	O
	`noOfVehicles`	`numeric []`	Number of emergency vehicles. If number is positive, vehicle(s) drive(s) in the same direction (want to overtake). If number is negative, vehicles drive in opposite direction.	M
	`vehicleType`	`enumString`	Type of the biggest emergency vehicle with values: `unknown`, `bicycle`, `motorcycle`, `car`, `truck`, `bus`, `military-vehicle`.	O

Table 39 *(continued)*

	drivingDirection	enumString	Direction to drive to, not to hinder the emergency vehicle(s).	O

DXMConfig	msg {
	{ iTSmsID 237, name { textId 10393, longname "emergency-vehicle" },
	executer '010'B, dataParamList { 28, 29, 30, 46, 47, 48 }
	}			

VIDFConfig	dataParam { ...
	{ rvId 46, name { textId 10437, longname "no of vehicles" }, dataTypeId 32,
	accessType write-internal, dataParamProperty other },
	{ rvId 47, name { textId 10438, longname "vehicle type" }, dataTypeId 346,
	accessType write-internal, dataParamProperty other },
	{ rvId 48, name { textId 10439, longname "driving direction" }, dataTypeId 42,
	accessType write-internal, dataParamProperty other }
	}, dataType { ...
	{ dataTypeId 42, name { textId 10355, longname "" }, type enumString: {
	{ value 0, name { textId 10356, longname "left" } },
	{ value 1, name { textId 10357, longname "forward" } },
	{ value 2, name { textId 10358, longname "right" } }
	} },
	{ dataTypeId 346, name { textId 10021, longname "vehicle type ()" },
	type enumString: {
	{ value 0, name { textId 10022, longname "none" } },
	{ value 1, name { textId 10023, longname "unknown" } },
	{ value 2, name { textId 10024, longname "bicycle" } },
	{ value 3, name { textId 10025, longname "motorcycle" } },
	{ value 4, name { textId 10026, longname "car" } },
	{ value 5, name { textId 10027, longname "truck" } },
	{ value 6, name { textId 10028, longname "bus" } },
	{ value 7, name { textId 10029, longname "tank" } }
	} }
	}			

9.14.2 Example

Table 40 shows an example of `emergency-vehicle` containing the notification number 9, the message type "alarm" and the position of the emergency vehicle. It is one emergency car driving in the same direction which wants you to drive to the right.

Table 40 — Example of emergency-vehicle

ASN.1	`sendEmergencyVehicleNotification DXMessage ::= { version 1, iTSmsID 237,`
	`  dataParamValue {`
	`  numeric: 9, -- notification number = 9`
	`  enumString: 0, -- message type = alarm`
	`  lnumeric: 406276320, -- problem coordinate.latitude = 50.78454 °N`
	`  lnumeric: 49036640, -- problem coordinate.longitude = 6.12958 °E`
	`  lnumeric: 11647, -- problem coordinate.elevation = 164.7 m`
	`  numeric: 1, -- no of vehicles = 1`
	`  enumString: 2, -- vehicle type = bicycle`
	`  enumString: 2 -- driving direction = right`
	`} } }`

U-PER	`--sendEmergencyVehicleNotification (34 bytes):`
	`00 30 00 00  1D A1 00 80  09 20 00 00  66 0D D2 38  03 05 D8 7A  C0 18 00 02`
	`D7 F0 40 00  90 00 08 80  00 40`

9.15 release

9.15.1 Definition

Table 41 defines the Data eXchange Message `release` to release a notification to the road user (P-ITS-S).

Table 41 — Definition of release

Msg	id	225	Release an existing notification.	
	name	`release`		
	exec	R-ITS-S		
Data Parameter	**Name**	**DataType [Unit]**	**Description**	**Cvt**
	`notificationNumber`	`numeric []`	Independent number of the notification.	M

DXMConfig	`msg {`
	`  { iTSmsID 225, name { textId 10382, longname "release" },`
	`  executer '010'B, dataParamList { 28 }`
	`}`			

9.15.2 Example

Table 42 shows an example of `release` releasing the road occupation notification 8.

Table 42 — Example of release

ASN.1	`sendReleaseNotification DXMessage ::= { version 1, iTSmsID 225, dataParamValue { numeric: 8 -- notification number = 8 - release sendRoadOccupationNotification } } }`
U-PER	`--sendReleaseNotification (9 bytes):` `00 30 00 00 1C 20 20 80 08`

9.16 search-parking-space

9.16.1 Definition

Table 43 defines the Data eXchange Message `search-parking-space` to request the R-ITS-S for searching a parking space.

Table 43 — Definition of search-parking-space

Msg	id	223	Request for searching a parking space.
	name	`search-parking-space`	
	exec	P-ITS-S	

Table 43 *(continued)*

	Name	DataType [Unit]	Description	Cvt
Data Parameter	position	structure	Current position of the road user with latitude, longitude and elevation (see Table 16; notify-on-position).	M
	vehicleSize	structure	Size of the vehicle with length, width, height and mass (see Table 16; notify-on-position).	M
	parkingSpace-Type	bitString	Bit mask of the desired parking space type with bits for echolonParking, parallelPark-ing, perpendicularParking, handicapped, woman, driver-with-child, small-vehicle.	M
	parking-SpacePosition	bitString	Bit mask of the desired parking position with bits for on-street, on-parking-area, closest-to-me, closest-to-exit, closest-to-pe-destrian-exit.	M

DXMConfig	msg {
	{ iTSmsID 223, name { textId 10380, longname "search-parking-space" },
	executer '100'B, dataParamList { 2, 22, 40, 41 }
	}			

Table 43 *(continued)*

VIDFConfig	dataParam { ...
	{ rvId 40, name { textId 10431, longname "parking space type" }, dataTypeId 36,
	accessType write-internal, dataParamProperty other },
	{ rvId 41, name { textId 10432, longname "parkingSpacePosition" }, dataTypeId 37,
	accessType write-internal, dataParamProperty other }
	}, dataType { ...
	{ dataTypeId 36, name { textId 10340, longname "parking space type (;)" },
	type bitString: {
	{ bit 0, name { textId 10341, longname "echolonParking" } },
	{ bit 1, name { textId 10341, longname " parallelParking " } },
	{ bit 2, name { textId 10341, longname " perpendicularParking " } },
	{ bit 3, name { textId 10342, longname "handicapped" } },
	{ bit 4, name { textId 10343, longname "woman" } },
	{ bit 5, name { textId 10344, longname "driver with child" } },
	{ bit 6, name { textId 10345, longname "small vehicle" } }
	} },
	{ dataTypeId 37, name { textId 10346, longname "parking space position (;)" },
	type bitString: {
	{ bit 0, name { textId 10347, longname "on street" } },
	{ bit 1, name { textId 10348, longname "on parking area" } },
	{ bit 2, name { textId 10349, longname "closest to me" } },
	{ bit 3, name { textId 10350, longname "closest to exit" } },
	{ bit 4, name { textId 10351, longname "closest to pedestrian exit" } }
	} }
	}

9.16.2 Example

Table 44 shows an example for `search-parking-space` requesting the R-ITS-S to search a parking space, including the current position of the vehicle, the vehicle size, the desired parking space type "normal or woman" and the desired parking position "on the street or on a parking area closest to me".

Table 44 — Example for search-parking-space

ASN.1	sendSearchParkingSpace DXMessage ::= { version 1, iTSmsID 223, dataParamValue {
	  lnumeric: 406276320, -- GPS position.latitude = 50.78454 °N
	  lnumeric: 49036640, -- GPS position.longitude = 6.12958 °E
	  lnumeric: 11647, -- GPS position.elevation = 164.7 m
	  numeric: 448, -- vehicle size.length = 4.48 m
	  numeric: 178, -- vehicle size.width = 1.78 m
	  numeric: 32, -- vehicle size.height = 1.60 m
	  numeric: 58, -- vehicle size.mass = 1450 kg
	  bitString: 5, -- parking space type = normal, woman
	  bitString: 7 -- parkingBayPosition = on street, on parking area,
	 -- closest to me
	} } }

U-PER	--sendSearchParkingSpace (36 bytes): 00 30 00 00 1B E1 21 98 37 48 E0 0C 17 61 EB 00 60 00 0B 5F C1 03 80 08 0B 20 40 10 02 00 E8 A0 00 A5 00 07

9.17 guide-parking-space

9.17.1 Definition

Table 45 defines the Data eXchange Message `guide-parking-space` to request the R-ITS-S for searching a parking space. If the vehicle is on the street, waypoints are provided to navigate to the parking space on the street or the parking area. To navigate inside the parking area, navigation hints are provided.

Table 45 — Definition of guide-parking-space

Msg	id	224	Guidance to a parking space.
	name	`guide-parking-space`	
	exec	R-ITS-S	

Table 45 *(continued)*

	Name	DataType [Unit]	Description	Cvt
Data Parameter	waypoint {}	array of structure	A list of waypoints to navigate to the on street parking space or the parking area with latitude, longitude and elevation (see Table 16; notify-on-position).	C
	navigationHint {}	array of structure	A list of hints to navigate inside the parking area containing following elements:	O
	directionAngle	numeric [°]	Angle of the direction to drive to, expressed in signed units of 0,005 493 247° from North.	M
	directionDistance	numeric [m]	Distance into the defined direction to drive.	M
DXMConfig	msg { { iTSmsID 224, name { textId 10381, longname "guide-parking-bay" }, executer '010'B, dataParamList { 27, 42 } }			
VIDFConfig	dataParam { ... { rvId 27, name { textId 10418, longname "waypoint" }, dataTypeId 26, accessType read-only, dataParamProperty collection }, { rvId 42, name { textId 10433, longname "navigation hint" }, dataTypeId 38, accessType read-only, dataParamProperty collection }, { rvId 43, name { textId 10434, longname "direction angle" }, dataTypeId 8, accessType write-internal, dataParamProperty other }, { rvId 44, name { textId 10435, longname "direction distance" }, dataTypeId 34, accessType write-internal, dataParamProperty other } }, dataType { ... { dataTypeId 26, name { textId 10320, longname "waypoints[] as GPS position" }, type array: 2 }, { dataTypeId 38, name { textId 10352, longname "navigation hint []" }, type array: 39 }, { dataTypeId 39, name { textId 10353, longname "navigation hint {}" }, type structure: { { 43, 44 }, convention mandatory } } }			

9.17.2 Example

Table 46 shows an example for guide-parking-space guiding the driver of the vehicle to the parking space. The example contains five waypoints to navigate from the street to a parking area. Additionally, navigation hints to find the parking space are provided: Drive 90 m to the west, then 40 m to the south.

Table 46 — Example for guide-parking-space

ASN.1

```
sendGuideParkingSpace DXMessage ::= { version 1, iTSmsID 224, dataParamValue {
  array: 5,            -- waypoint                                = 5 sub element(s)
  lnumeric: 406255600, -- waypoint.[0].latitude                     = 50.78195 °N
  lnumeric: 48413120,  -- waypoint.[0].longitude                    = 6.05164 °E
  lnumeric: 0,         -- waypoint.[0].elevation                    = -1000.0 m
  lnumeric: 406197840, -- waypoint.[1].latitude                     = 50.77473 °N
  lnumeric: 48396000,  -- waypoint.[1].longitude                    = 6.04950 °E
  lnumeric: 0,         -- waypoint.[1].elevation                    = -1000.0 m
  lnumeric: 406193440, -- waypoint.[2].latitude                     = 50.77418 °N
  lnumeric: 48379360,  -- waypoint.[2].longitude                    = 6.04742 °E
  lnumeric: 0,         -- waypoint.[2].elevation                    = -1000.0 m
  lnumeric: 406206560, -- waypoint.[3].latitude                     = 50.77582 °N
  lnumeric: 48378240,  -- waypoint.[3].longitude                    = 6.04728 °E
  lnumeric: 0,         -- waypoint.[3].elevation                    = -1000.0 m
  lnumeric: 406204480, -- waypoint.[4].latitude                     = 50.77556 °N
  lnumeric: 48359920,  -- waypoint.[4].longitude                    = 6.04499 °E
  lnumeric: 0,         -- waypoint.[4].elevation                    = -1000.0 m
  array: 2,            -- navigation hint                          = 2 sub element(s)
  lnumeric: 49151,     -- navigation hint.[0].direction angle       = 270 °
  lnumeric: 900,       -- navigation hint.[0].direction distance    = 90.00 m
  lnumeric: 32768,     -- navigation hint.[1].direction angle       = 180 °
  lnumeric: 400        -- navigation hint.[1].direction distance    = 40.00 m
} } }
```

U-PER

```
--sendGuideParkingSpace (100 bytes):
00 30 00 00  1C 02 A7 00  05 0C C1 B7  BF 80 60 B8  AE 70 03 00  00 00 00 19
83 61 65 00  C1 71 3B 70  06 00 00 00  00 33 06 C0  A4 01 82 E2  35 E0 0C 00
00 00 00 66  0D 8E 18 03  05 C4 63 00  18 00 00 00  00 CC 1B 18  20 06 0B 87
A7 C0 30 00  00 00 07 00  02 0C 00 05  FF F8 60 00  00 E1 03 00  01 00 00 18
00 00 19 00
```

Annex A
(normative)

Vehicle Interface Data Format (VIDF)

A.1 VIError

Table A.1 defines the VIError.

Table A.1 — VIError definition

	Name	Description	Cvt
Attributes	errorId	Error identifier	M
	name	Display name of the error	M
	attributeCount	Number of attributes used by this error	M
Example	{ errorId 10118, name { textId 10118, longname "Invalid file activity type" }, attributeCount 0 }		

A.2 UnitType

Table A.2 defines the UnitType collecting units in groups. Typical unit types are length, weight, time, temperature, pressure, etc.

Table A.2 — UnitType definition

	Name	Description	Cvt
Attributes	unitTypeId	Identifier of the unit type	M
	name	Display name of the unit type	M
Example	{ unitTypeId 5, name { textId 9005, longname "time" } }, { unitTypeId 7, name { textId 9007, longname "temperature" } }, { unitTypeId 8, name { textId 9008, longname "pressure" } }		

A.3 Unit

Table A.3 defines the Unit and, its relation to a unit type, formula and its attributes to calculate the default unit inside a unit type.

Table A.3 — Unit definition

	Name	Description	Cvt
Attributes	unitTypeId	Identifier of the unit type as reference (see A.1)	M
	unitId	Identifier of the unit	M
	name	Display name of the unit	M
	formula	Used formula to calculate default unit of same unit type	M
	c0	C0 parameter for formula calculation	M
	c1	C1 parameter for formula calculation	M
	c2	C2 parameter for formula calculation	M
Example	{ unitTypeId 7, unitId 11, name { textId 9111, shortname "°C", longname "degree celsius" }, Formula (0), c0 0, c1 0, c2 0 }, { unitTypeId 7, unitId 39, name { textId 9139, shortname "°F", longname "fahrenheit" }, Formula (4), c0 −32, c1 10, c2 18 }		

A.4 Provider

Table A.4 defines the Provider.

Table A.4 — Provider definition

	Name	Description	Cvt
Attributes	providerId	Identifier of the provider	M
	name	Display name of the provider	M
Example	{ providerId 33, name { textId 9233, longname "Honda" } }, { providerId 34, name { textId 9234, longname "Hyundai" } }		

A.5 ECU

Table A.5 defines the ECU.

Table A.5 — ECU definition

	Name	Description	Cvt
Attributes	ecuId	Identifier of the unit type	M
	name	Display name of the unit type	M
	providerId	Identifier of a provider	M
	parentId	Identifier of a parent ECU	O
Example	{ ecuId 17, name { textId 10202, shortname "ECM", longname "engine control module" }, providerId 0 }, { ecuId 21, name { textId 10203, shortname "CTCM", longname "coolant temperature control module" }, providerId 0 }		

A.6 DataType

A.6.1 DataType attributes

Table A.6 defines the DataType attributes.

Table A.6 — DataType attributes

	Name	Description	Cvt
Attributes	dataTypeId	Identifier of the data type	M
	name	DisplayName of the data type	O
	type	Type choice between the following sub types:	M
	– numeric	— A numeric value (Numeric) containing the following sub-attributes (see A.6.2):	C0[a]
	– decimalPlaces	— Number of decimal places for the display	O
	– unitId	— Unit identifier as reference (see A.3)	M
	– factor	— Factor to multiply with; default value is 1	O
	– quotient	— Quotient to divide with; default value is 1	O
	– addend	— Addend to add to; default value is 0	O
	– min	— Minimal numeric value	O
	– max	— Maximal numeric value	O
	– lnumeric	— A long numeric value (LNumeric) containing the same attributes as numeric (see A.6.2):	C1[a]
	– string	— A limited string (LimitedString) containing the following sub-attributes (see A.6.3):	C2[a]
	– allowedCharacters	— A list (regular expression) of the allowed characters	O
	– minLen	— Minimal length of the desired string	O
	– maxLen	— Maximal length of the desired string	O
	– displayName	— An internationalizable string (contains no data for definition)	C3[a]
	– enumString {}	— A list of enumeration string item values (EnumStringItem) containing (see A.6.5):	C4[a]
	– value	— Value of the enumeration string item	M
	– name	— Display name of the string table item	M
	– bitString {}	— A list of bit string item values (BitStringItem) containing the following attributes (see A.6.6):	C5[a]
	– bit	— Value of the enumeration string item	M
	– name	— Display name of the string table item	M
	– structure	— A Structure element containing following attributes (see A.6.7):	C6[a]
	– param {}	— A list of the contained registered value identifiers (see A.7)	M
	– convention	— Convention of the structure (`mandatory`, `optional`, `conditional`); default value is `mandatory`	O
	– array	— DataTypeId of the elements of the data array	C7[a]
	– monitor {}	— A list of monitor item values (MonitorItem) containing the following attributes:	C8[a]
	– testId	— Test ID of the monitor item	M
	– decimalPlaces	— Number of decimal places for the display	O

[a] C0, C1, C2, C3, C4, C5, C6 and C7 are the choices of the dataTypes' type. Only one of C0 or C1 or C2 or C3 or C4 or C5 or C6 or C7 or C8 or C9 can be used.

Table A.6 *(continued)*

Attributes	– unitId	— Unit identifier as reference (see A.3)	M
	– factor	— Factor to multiply with; default value is 1	O
	– quotient	— Quotient to divide with; default value is 1	O
	– addend	— Addend to add to; default value is 0	O
	– octet	— An octet string	C9[a]
[a] C0, C1, C2, C3, C4, C5, C6 and C7 are the choices of the dataTypes' type. Only one of C0 or C1 or C2 or C3 or C4 or C5 or C6 or C7 or C8 or C9 can be used.			

The data types can be `numeric`, `lnumeric` (long numeric), `string`, `enumString` (an enumeration string), `bitString`, `structure`, `array` or `monitor`.

A.6.2 numeric, lnumeric

The data type `numeric` of ASN.1 type `Numeric` is used to realize small integer and floating point values (2 bytes). The data type `lnumeric` of ASN.1 type `LNumeric` is used to realize long integer and long floating point values (4 bytes). To minimize the data parameter value length and not to transport floating values, all floating values can be defined as SNUM16 (`Numeric`) or SNUM32 (`LNumeric`). To transform into the floating value, the calculation attributes factor (default = 1), quotient (default = 1) and addend (default = 0) are used. The calculation rule for the realValue, realMinValue and realMaxValue is defined in Formula (A.1). The real value is rounded to decimalPlaces with unit.

Definition of formula

$$\texttt{realValue} = \texttt{value} \times \texttt{factor} \,/\, \texttt{quotient} + \texttt{addend} \qquad \text{(A.1)}$$

Table A.7 — Numeric example

Example	`{ dataTypeId 331, name { textId 10001, longname "voltage in 1/100 Volt" },`
	`type numeric: {`
	`decimalPlaces 2, unitId 17, factor 1, quotient 100, addend 0, min 880, max 1560`
	`} },`
	`{ dataTypeId 333, name { textId 10003, longname "temperature in 1/10 °C" },`
	`type numeric: {`
	`decimalPlaces 1, unitId 11, factor 1, quotient 10, addend 0, max 1200`
	`} }`

The first example of Table A.7 (`dataTypeId 331`) defines a numeric value between 8,80 V and 15,60 V (realMinValue = 880 × 1 / 100 + 0 = 8,8; realMaxValue = 1 560 × 1 / 100 + 0 = 15,6; two decimal places; unit is V).

The second example of Table A.7 (`dataTypeId 333`) defines a numeric value with a maximum of 120,0 °C.

A.6.3 string

The data type `string` of ASN.1 type `LimitedString` realizes limited and unlimited strings. The optional attribute `allowedCharacters` contains a regular expression of the allowed characters in the string. The optional attributes `minLen` and `maxLen` define the range of the character count. The first example in Table A.8 defines a "`Vehicle Identification Number`" string with exactly 17

characters (`minLen = maxLen = 17`) and may contain only capital letters from `A-H`, `J-N`, `P`, `R-Z` or the digits `0-9`. The second example defines an unlimited string.

Table A.8 — Limited string example

Example	`{ dataTypeId 360, name { textId 10058, longname "Vehicle Identification Number" }, type string: {`
	`  allowedCharacters "A..HJ..NPR..Z0..9", minLen 17, maxLen 17`
	`} },`
	`{ dataTypeId 334, name { textId 10004, longname "unlimited string" }, type string: {`
	`} }`

A.6.4 displayName

The data type `displayName` of ASN.1 type `NULL` realizes internationalizable strings and contains no attributes. Table A.9 defines an example for DisplayName.

Table A.9 — DisplayName example

Example	`{ dataTypeId 342, name { textId 10015, longname "display name" },`
	`  type displayName: NULL`
	`}`

A.6.5 enumString

The data type `enumString` of ASN.1 type `SEQUENCE OF EnumStringItem` realizes enumerations of values in text form. Only one of the `EnumStringItems` can be selected. The example in Table A.10 defines an enumeration of the two texts "no" and "yes". If the enum value is `0`, "`no`" is selected, if the enum value is `1`, "`yes`" is selected.

Table A.10 — Enum string example

Example	`{ dataTypeId 332, name { textId 10002, longname "answer (no, yes)" },`
	`  type enumString: {`
	`  { value 0, name { textId 10059, longname "no" } },`
	`  { value 1, name { textId 10060, longname "yes" } }`
	`} }`

A.6.6 bitString

The data type `bitString` of ASN.1 type `SEQUENCE OF BitStringItem` realizes combinations of enumerations of values in text form. So, any of the `BitStringItems` can be selected. The example in Table A.11 defines a weather selection matrix with bits for rain, snow, ice, fog and strong wind. Any of the "weather bits" can be selected.

Table A.11 — Bit string example

Example	`{ dataTypeId 33, name { textId 10329, longname "weather condition" },`
	`  type bitString: {`
	`  { bit 0, name { textId 10330, longname "rain" } },`
	`  { bit 1, name { textId 10331, longname "snow" } },`
	`  { bit 2, name { textId 10332, longname "ice" } },`
	`  { bit 3, name { textId 10333, longname "fog" } },`
	`  { bit 4, name { textId 10334, longname "strong wind" } }`
	`} }`

The general value is calculated by Formula (A.2).

Definition of formula

$$\text{value} = \sum_{\text{bit}=1}^{\text{maxbit}} \begin{cases} 2^{\text{bit}}, & \text{bit is selected} \\ 0, & \text{bit is not selected} \end{cases} \tag{A.2}$$

The value for "snow and ice" is calculated by $\texttt{value}_{\texttt{snow\&ice}} = 2^{\texttt{bitsnow}} + 2^{\texttt{bitice}} = 2^1 + 2^2 = 2 + 4 = 6$. So the bit string value shall be coded as `bitString`: 6.

A.6.7 structure

The data type `structure` of ASN.1 type `Structure` realized summarizations of data parameters. The first example in Table A.12 defines a structure with the sub parameters 7, 8 and 9. A sub-parameter can be a structure as well. Recursions are not allowed, i.e. a parameter may not have itself as direct or indirect child.

Table A.12 — Structure example

Example	`{ dataTypeId 6, name { textId 10308, longname "vehicle motion {}" },`
	`  type structure: {`
	`  { 7, 8, 9 }, convention conditional`
	`} },`
	`{ dataTypeId 345, name { textId 10020, longname "vehicle info {}" },`
	`  type structure: {`
	`  { 20000, 20001, 20002, 20003, 20004, 20005, 20006, 20007,`
	`  20010, 20011, 20012, 20020, 20021, 461 }, convention mandatory`
	`} }`

A.6.8 array

The data type `array` of ASN.1 type `UNUM16` realized arrays of data types. The example in Table A.13 defines the data type 29 as array of the data type 30. The contained data type can be of any type. Recursions are not allowed, i.e. a data type may not have itself as direct or indirect child.

Table A.13 — Array example

Example	`{ dataTypeId 29, type array: 30 },`

A.6.9 monitor

The data type `monitor` of ASN.1 type `SEQUENCE OF MonitorItem` realized Monitors. Table A.14 defines a monitor with two tests. Every test must have a `testId`. The parameters `decimalPlaces`, `unit`, `factor`, `quotient` and `addend` are equal as defined in `numeric` (see A.6.2). The parameters `min` and `max` are not defined here but within the monitor value (see A.11.8).

Table A.14 — Monitor example

Example	`{ dataTypeId 335, name { textId 10005, longname "oxygen sensor monitor" },`
	`type monitor: {`
	`{ testId 1, decimalPlaces 3, unitId 17, factor 1, quotient 1000, addend 0 },`
	`{ testId 5, decimalPlaces 3, unitId 31, factor 1, quotient 1000, addend 0 }`
	`} }`

A.6.10 octet

The data type `octet` of ASN.1 type `SNUM32` realized binary data as octet strings. The example in Table A.15 defines the data type 379 as an octet of variable size and the data type 380 as an octet of 8 bytes.

Table A.15 — Octet example

Example	`{ dataTypeId 379, name { textId 10094, longname "octetByte of variable size" },`
	`type octet: 0 },`
	`{ dataTypeId 380, name { textId 10095, longname "octetByte with 8 byte" },`
	`type octet: 8 }`

A.7 DataParam

Table A.16 defines the data parameter attributes.

Table A.16 — DataParam attributes

	Name	Description	Cvt
Attributes	`rvId`	Registered value identifier	M
	`name`	DisplayName of the data parameter	M
	`dataTypeId`	Reference to the data type identifier (see A.6.1)	M

Table A.16 *(continued)*

Attributes	accessType	The access type of the data parameter is a bit string with a combination of following values: r (0) = read, w (1) = write, x (2) = execute, i (3) = internal and u (4) = user.	M
	description	Description of the data parameter	O
	dataParamProperty	The data parameter property is a enumeration with following values: ecu-supported-info, sensor, actuator, ecu-internal-signal, ecu-internal-monitor, collection, routine, fix and other.	M
Example	{ rvId 461, name { textId 10511, shortname "VIN", longname "vehicle identification number" }, dataTypeId 360, accessType '10010'B, dataParamProperty ecu-internal-signal }, { rvId 1002, name { textId 10130, shortname "ECMB+", longname "engine control module voltage" }, dataTypeId 331, accessType '10000'B, dataParamProperty sensor }, { rvId 1123, name { textId 10131, shortname "HW_PART_NUMBER", longname "hardware part number" }, dataTypeId 334, accessType '10000'B, dataParamProperty ecu-internal-signal }, { rvId 2341, name { textId 10132, shortname "ECT", longname "engine coolant temperature" }, dataTypeId 333, accessType '10000'B, dataParamProperty sensor }, { rvId 7368, name { textId 10134, shortname "AIR_RDY", longname "secondary air system monitoring ready" }, dataTypeId 332, accessType '10000'B, dataParamProperty ecu-internal-monitor }, { rvId 20025, name { textId 10114, shortname "VehInfo", longname "vehicle info" }, dataTypeId 345, accessType '10000'B, dataParamProperty collection }		

The data parameter is identified by the rvId, the registered value identifier. It is possible to have the same rvId on multiple ECUs to support, e.g. the battery voltage (ignition on) of every ECU. Further parameters are the unique name for the name of the parameter. The dataTypeId references the data type defined in A.6. The attributes accessType and dataParamProperty can be used for filtering the data parameters. The parentId is used for routines.

A.8 DataParamMapping

Table A.17 defines the DataParamMapping.

Table A.17 — DataParamMapping definition

	Name	Description	Cvt
Attributes	rvId	Registered value identifier	M
	ecuId	ECU identifier	M
	arrayIndex	Index of an array for addressing, if the data parameter corresponding data type is an array	O
Example	{ rvId 1002, ecuId 17 }, { rvId 1123, ecuId 21 }		

A.9 DtcBase

Table A.18 defines the DtcBase.

Table A.18 — DtcBase definition

	Name	Description	Cvt
Attributes	rDtcBaseId	DTC base identifier	M
	providerId	Identifier of the provider of the DTC base, if the DTC base is provider specific	C1[a]
	ecuId	Identifier of the ECU supporting the DTC base, if the DTc base is only supported in this ECU	C2[a]
	name	DisplayName of the DTC base	M
	description	Description of the DTC base	O
	dataParamList {}	List of all related data parameter rvIds	C3[b]
	dataParamMapping {}	List of mappings between data parameter and ECU (see A.8)	C4[b]
Example	{ rDtcBaseId 4, providerId 1, name { textId 20004, longname "Fuel Volume Regulator Control Circuit High" } }, { rDtcBaseId 5, ecuId 21, name { textId 20005, longname "Fuel Shutoff Valve "A" Control Circuit/Open" } dataParamMapping { { rvId 1002, ecuId 17 }, { rvId 2341, ecuId 17 } } } }, { rDtcBaseId 295, providerId 1, name { textId 20006, longname "Intake Air Temperature Too High" } }, { rDtcBaseId 49280, providerId 1, name { textId 20009, longname "Vehicle Communication Bus 'F'" } }		

[a] Either C1 or C2 must be defined.

[b] Either C3 or C4 can be defined.

A.10 DtcSymptom

Table A.19 defines the DtcSymptom.

Table A.19 — DtcSymptom definition

	Name	Description	Cvt
Attributes	rdtcSymptomId	DTC symptom identifier	M
	providerId	Identifier of the provider of the DTC symptom, if the DTC symptom is provider specific	C1[a]
	ecuId	Identifier of the ECU supporting the DTC symptom, if the DTC symptom is only supported in this ECU	C2[a]
	name	DisplayName of the DTC symptom	M
	description	Description of the DTC symptom	O

Example	`{ rDtcSymptomId 4, providerId 1,`
	`  name { textId 21004, longname "System Internal Failure" },`
	`  description { textId 22004,`
	`  longname "This sub type is used for control module Internal Failures ..." }`
	`},`
	`{ rDtcSymptomId 8, providerId 1,`
	`  name { textId 21008, longname "Bus Signal/Message Failure" } }`
	`}`		

[a] Either C1 or C2 must be defined.

A.11 DataParamValue

A.11.1 DataParamValue attributes

Table A.20 defines the DataParamValue attributes.

Table A.20 — DataParamValue attributes

	Name	Description	Cvt
Attributes	numeric	Numeric value as SNUM16	C0[c]
	lnumeric	Long numeric value as SNUM32	C1[c]
	string	String value as String	C2[c]
	displayName	DisplayName	C3[c]
	enumString	UNUM16 value; reference to the enumeration string item	C4[c]
	bitString	UNUM16 value containing the set bits of the bit string items	C5[c]
	structureMissing	Dependence level of missing structure as numeric value as UNUM8	C6[c]
	array	Number of elements in the array as UNUM16	C7[c]
	monitor {}	List of MonitorValue elements with following attributes:	C8[c]
	— testValue	— SNUM32 testValue	M

[c] C0, C1, C2, C3, C4, C5, C6, C7, C8 and C9 are the choices of the data parameter value. Only one of C0 or C1 or C2 or C3 or C4 or C5 or C6 or C7 or C8 or C9 or C10 can be used.

Table A.20 *(continued)*

Attributes	– testValueMin	— Minimal test value	O
	– testValueMax	— Maximal test value	O
	octet	Octet value with optional length and data as octet string	C9[c]
	error	Error ID, if the parameter cannot be retrieved	C10[c]

[c] C0, C1, C2, C3, C4, C5, C6, C7, C8 and C9 are the choices of the data parameter value. Only one of C0 or C1 or C2 or C3 or C4 or C5 or C6 or C7 or C8 or C9 or C10 can be used.

The data parameter values must be mapped to the data type definition.

A.11.2 numeric, lnumeric

The data type value `numeric` is an `SNUM16`. The data type `lnumeric` is an `SNUM32`. So the value is very compact. The real value is calculated by Formula (A.1) with the attributes of the data type definition. To visualize the value it must be rounded to the decimalPlaces defined in the data type. The example `numeric: 1 250` corresponding to the data type `331` defined in A.6.2 must be interpreted as 12,50 V.

A.11.3 string

The data type value `string` is a `String`. The example string: "`VHJGH11763B65I860`" can be easily identified.

A.11.4 enumString

The data type value `enumString` is a `UNUM16` referencing the value of the corresponding data type `EnumStringItem`. The `enumString: 0` mapping to the `dataTypeId 332` defined in A.6.5 must be interpreted as "`no`". An `enumString` mapping to another `dataTypeId` has a complete different meaning.

A.11.5 bitString

The data type value `bitString` is a `UNUM16`. It must be interpreted as bit mask as defined in Formula (A.2). The `bitString: 6` mapping to the `dataTypeId 33` defined in A.6.6 must be interpreted as "snow and ice". Of course a `bitString` mapping to another `dataTypeId` has a complete different meaning.

A.11.6 structureMissing

The data type value `structureMissing` is a `UNUM8` containing the dependence level of the missing structure as numeric value. If a structure is expected in the data parameter value list, its contents are displayed. So only if the structure is optional, a `structureMissing` is used.

A.11.7 array

The data type value `array` is a `UNUM16` defining the size of the corresponding array. The data parameter value `array: 2` mapping to the `dataTypeId 29` defined in A.6.8 identifies an array with two elements with `dataTypeId 30`.

A.11.8 monitor

The data type value `monitor` is a `SEQUENCE OF MonitorValue`. It contains the test values and optional minimum and maximum of the `MonitorItem`s of the corresponding data type definition. The example in Table A.21 mapping to the `dataTypeId 335` defined in A.6.9 provides test values `365` and `72` with minimum and maximum for `testId 1` and `5`.

Table A.21 — Monitor value example

Example	`value monitor: {`
	`  { testValue 365, testValueMin 365, testValueMax 365 },`
	`  { testValue 72, testValueMin 0, testValueMax 100 }`
	`}`

By using Formula (A.1) with the attributes of the data type definition, the example values must be interpreted as

`testId1: testValue = 0.365 V (0.365 - 0.365 V)`

`testId5: testValue = 0.072 s (0.000-0.100 s)`

A.11.8.1 octet

The data type value `octet` is an `OctetValue`. If the corresponding data type has an octet 0 (variable size), the length parameter defines the length of the octet string and data contains the data as octet string. If the corresponding data type has octet > 0 (fix size), the length parameter is not set. The example in Table A.22 defines an octet value of length 15.

Table A.22 — Octet value example

Example	`{ value octet: {`
	`  length 15, data '54686973206973206D792064617461'H }`
	`}`

A.11.9 error

The data type value `error` is a `SNUM32` defining the error retrieving the corresponding data parameter value. Every data parameter can result in an `error`. The error number references predefined errors.

A.12 VIErrorValue

Table A.23 defines the VIErrorValue.

Table A.23 — VIErrorValue definition

	Name	Description	Cvt
Attributes	`errorId`	Error identifier	M
	`attribute {}`	List of data parameter values	O
Example	`{ errorId 1 },` `{ errorId 633, attribute { numeric: 32 } }`		

A.13 DataParamValueTS

Table A.24 defines the data parameter value time stamp (value with time stamp).

Table A.24 — DataParamValueTS definition

	Name	Description	Cvt
Attributes	`value`	Data parameter value (see A.11)	M
	`timeInMillis`	Time in milliseconds since 1970	O

Example	`{ value numeric: 1250, time 0L },`
	`{ value enumString: 1, time 0L },`
	`{ value error: 422, time 0L }`		

A.14 DataParamValueMapping

Table A.25 defines the DataParamValueMapping.

Table A.25 — DataParamValueMapping definition

	Name	Description	Cvt
Attributes	`rvId`	Data parameter value identifier	M
	`ecuId`	ECU identifier	O
	`value`	Data parameter value (see A.11)	M
	`timeInMilis`	Time in milliseconds since 1970	O
Example	`{ rvId 1002, ecuId 17 },` `{ rvId 1123, ecuId 21 }`		

A.15 DtcInfo

Table A.26 defines the DtcInfo.

Table A.26 — DtcInfo definition

	Name			Description		Cvt
Attributes	`rDtcBaseId`			Base identifier of the DTC		M
	`rDtcSymptomId`			Symptom identifier of the DTC		
	`ecuId`			ECU identifier		O
	`complementary`			Bit mask with all complementary DTC information, e.g. status, severity, class using the following bits:		M
	bit	**mnemonic**	**name**	**= 0**	**= 1**	
	0	TF	testFailed	Most recent result from DTC test indicated no failure detected.	Most recent result from DTC test indicated a matured failing result.	U
	1	TFTOC	testFailed-ThisOperationCycle	testFailed: result has not been reported during the current operation cycle or after a call was made to ClearDiagnosticInformation during the current operation cycle.	testFailed: result was reported at least once during the current operation cycle.	U

Table A.26 *(continued)*

Attributes	2	PDTC	pendingDTC	This bit shall be set to 0 after completing an operation cycle during which the test completed and a malfunction was not detected or upon a call to the ClearDiagnosticInformation service.	This bit shall be set to 1 and latched if a malfunction is detected during the current operation cycle.	U
	3	CDTC	confirmed-DTC (present)	DTC has never been confirmed since the last call to ClearDiagnosticInformation or after the aging criteria have been satisfied for the DTC (or DTC has been erased due to fault memory overflow).	DTC confirmed at least once since the last call to ClearDiagnosticInformation and aging criteria have not yet been satisfied.	M
	4	TNCSLC	testNot-Completed-SinceLast-Clear	DTC test has returned either a passed or failed test result at least one time since the last time diagnostic information was cleared.	DTC test has not run to completion since the last time diagnostic information was cleared.	U
	5	TFSLC	test-FailedSin-ceLast-Clear	DTC test has not indicated a failed result since the last time diagnostic information was cleared. It is the responsibility of the vehicle manufacturer if this bit shall also be reset to zero (“0”) in case aging threshold is fulfilled or an overflow of the fault memory occurs.	DTC test returned a failed result at least once since the last time diagnostic information was cleared.	U
	6	TNCTOC	testNot-Completed-ThisOpera-tionCycle	DTC test has returned either a passed or testFailedThisOperationCycle = “1” result during the current drive cycle (or since the last time diagnostic information was cleared during the current operation cycle).	DTC test has not run to completion this operation cycle (or since the last time diagnostic information was cleared this operation cycle).	U
	7	WIR	warn-ingIndi-catorRe-quested	Server is not requesting warningIndicator to be active	Server is requesting warningIndicator to be active	U
	8	PERM	permanent	The confirmed DTC is not retained in the non-volatile memory.	The confirmed DTC is retained in the non-volatile memory of the server until the appropriate monitor for each DTC has determined that the malfunction is no longer present and is not commanding the MIL on.	M
	9	MO	mainte-nanceOnly	no maintenanceOnly severity	maintenanceOnly severity	M
	10	CHKANH	checkAt-NextHalt	do not checkAtNextHalt	checkAtNextHalt	M

Table A.26 *(continued)*

Attributes	11	CHKI	checkImmediately	do not checkImmediately	checkImmediately	M
	12	DTCClass_0	DTCClass_0	disabled for the reported DTC	enabled for the reported DTC	M
				DTCClass 0 is unclassified. This class shall be used if DTCSeverity is included in the response message but no DTC class information is reported, e.g. legacy DTCs as defined in SAE J2012-DA and ISO 14229–1.		
	13	DTCClass_1	DTCClass_1	disabled for the reported DTC	enabled for the reported DTC	M
				DTCClass_1 matches the GTR module B Class A definition. A malfunction shall be identified as Class A when the relevant OBD threshold limits (OTLs) are assumed to be exceeded. It is accepted that the emissions may not be above the OTLs when this class of malfunction occurs.		
	14	DTCClass_2	DTCClass_2	disabled for the reported DTC	enabled for the reported DTC	M
				DTCClass_2 matches the GTR module B Class B1 definition. A malfunction shall be identified as Class B1 where circumstances exist that have the potential to lead to emissions being above the OTLs but for which the exact influence on emission cannot be estimated and thus the actual emissions according to circumstances may be above or below the OTLs. Class B1 malfunctions shall include malfunctions that restrict the ability of the OBD system to carry out monitoring of Class A or B1 malfunctions.		
	15	DTCClass_3	DTCClass_3	disabled for the reported DTC	enabled for the reported DTC	M
				DTCClass_3 matches the GTR module B Class B2 definition. A malfunction shall be identified as Class B2 when circumstances exist that are assumed to influence emissions but not to a level that exceeds the OTL. Malfunctions that restrict the ability of the OBD system to carry out monitoring of Class B2 malfunctions of shall be classified into Class B1 or B2.		
Attributes	16	DTCClass_4	DTCClass_4	disabled for the reported DTC	enabled for the reported DTC	M
				DTCClass_4 matches the GTR module B Class C definition. A malfunction shall be identified as Class C when circumstances exist that, if monitored, are assumed to influence emissions but to a level that would not exceed the regulated emission limits. Malfunctions that restrict the ability of the OBD system to carry out monitoring of Class C malfunctions shall be classified into Class B1 or B2.		

Table A.26 *(continued)*

	`envData {}`	A list of data parameter value time stamps (see A.13).	O
	`timeInMillis`	Time in milliseconds since 1970	O

Example	`{ rDtcBaseId 157, rDtcSymptomId 1, ecuId 17, complementary '0010000000100000'B }`
	`{ rDtcBaseId 5, rDtcSymptomId 1, ecuId 21, complementary '0010000000000000'B,`
	`envData {`
	`{ value numeric: 1250, time 0L },`
	`{ value numeric: 910, time 0L }`
	`} }`		

A.16 VIDF example

To explain the example data, Figure A.1 shows an overview of most of the data parameters, the data types and the mapping to the ECUs.

The example contains in addition to other data parameter declarations the definition of a routine "Routine1" as data parameter. The routine itself and all of its input and output parameters have the dataParamProperty routine. The accessType of the routine is "x", of its input parameters "w" and of its output parameters "r". To map the input and output parameters to the routine the routineId is set to the rvId of the routine data parameter (8673).

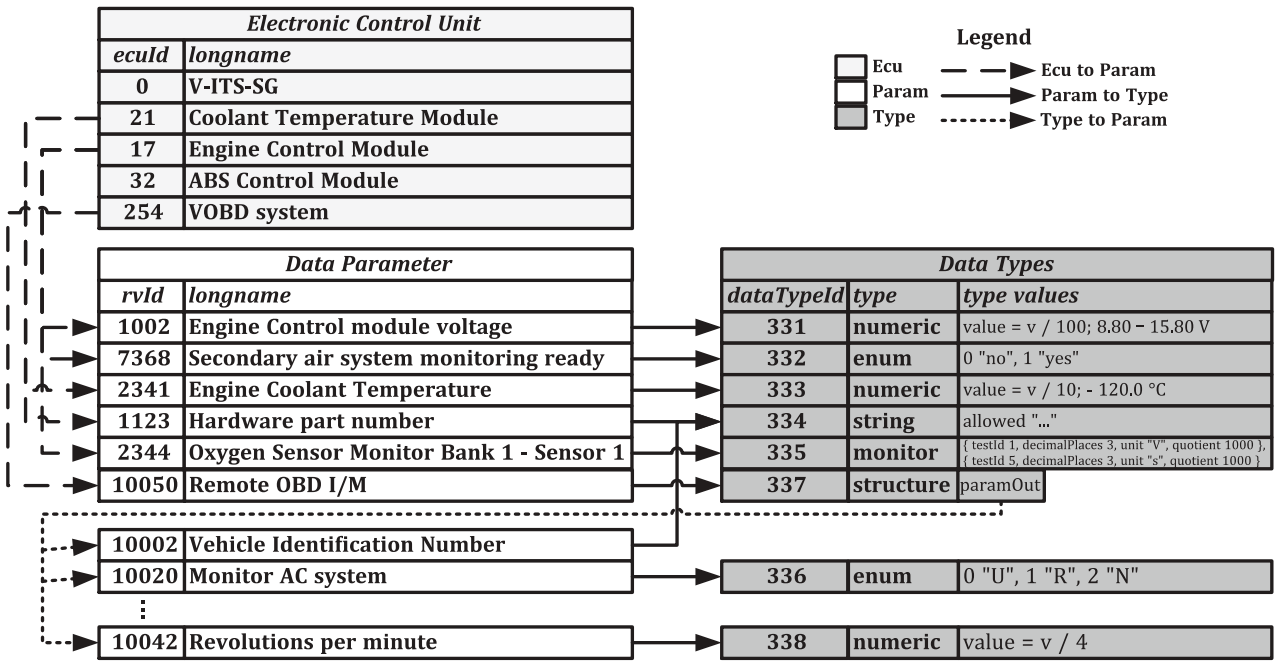


Figure A.1 — VIDF example

A.17 Internationalization

For internationalization purposes, the ASN.1 sequence `DisplayName` has been defined (see Table A.27). So, all texts, that have to be internationalized, have to be transformed into a `DisplayName`. On the

displaying device, a language pack (see A.18) for every language has to be added to the configuration. Each `textId` uniquely identifies a text and can be used to translate this text into the desired language.

Figure A.2 shows an example to demonstrate internationalization in DXM between P-ITS-S and C-ITS-SG/R-ITS-S. In the R-ITS-SG, the `text IDs` 9117, 10001 and 10130 are defined amongst others for the English text (region US as default) "volt", "power in cV" and "engine control module voltage". If the P-ITS-S is located in the US, the text could be displayed as received. For the locale `de_DE`, a language pack must be installed. With the `textId` as unique identifier, the replied texts become translated. The P-ITS-S can display the German name of the data parameter (including the unit): "Motorsteuergeräte-Spannung: 12,80 Volt".

Table A.27 — Internationalization with DisplayName

	Name	**Description**	**Cvt**
Attributes	`textId`	Unique text identifier	M
	`shortname`	Short version or abbreviation of the name to be displayed	O[a]
	`longname`	Long version of the name to be displayed (default name)	O[a]
[a] Optional text in a default language. This can be the language of the device vendor, or the country where the device is provided.			
ASN.1			

```
DisplayName :: = SEQUENCE {
    textId              Identifier,
    shortname           UTF8String                  OPTIONAL,
    longname            UTF8String                  OPTIONAL,
    ...
}
```

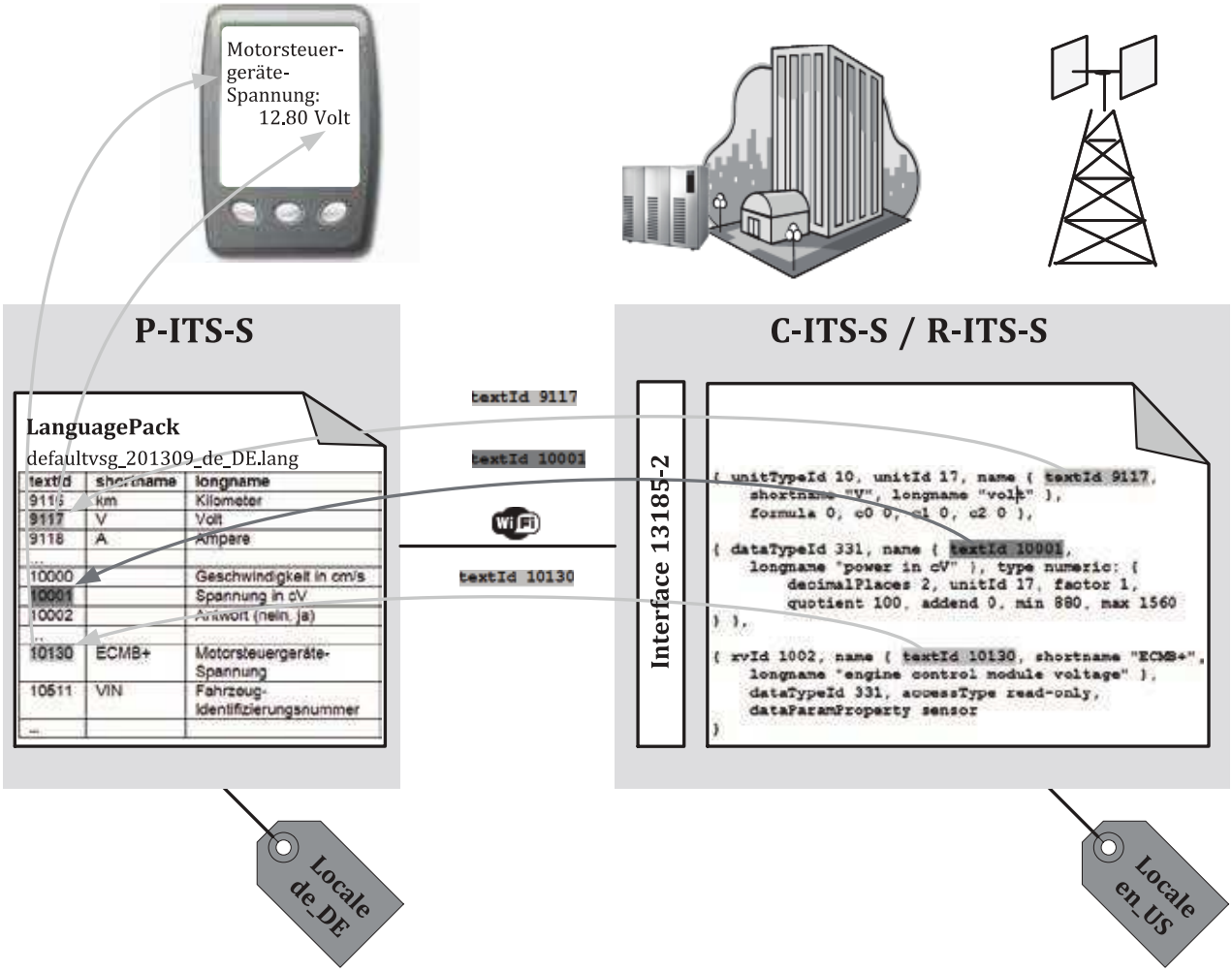


Figure A.2 — Internationalization

A.18 LanguagePack

Table A.28 defines the `LanguagePack`. The `LanguagePack` collects all internationalizable strings.

Table A.28 — `LanguagePack` definition

	Name	Description	Cvt
Attributes	`language`	Name of the language (en_US, de_DE, ...)	M
	`text {}`	List of all DisplayNames (see A.17)	M

Example	`language "en_US", text {`
	`{ textId 9117, shortname "V", longname "volt" },`
	`{ textId 10001, longname "power in cV" },`
	`{ textId 10130, shortname "ECMB+", longname "engine control module voltage" },`
	`}`		

A.19 VIDFConfig

Table A.29 defines the VIDFConfig collecting all client necessary information.

Table A.29 — VIDFConfig definition

	Name	Description	Cvt
Attributes	`configName`	Name of the configuration	M
	`configVersion`	Version of the configuration (i.e. date, number)	M
	`modelConfigName`	Name of the represented vehicle	M
	`vehicleInfo`	Vehicle info	M
	`error {}`	List of Vehicle Interface errors (see A.1)	M
	`unitType {}`	List of unit type definitions (see A.2)	M
	`unit {}`	List of unit definitions (see A.3)	M
	`provider {}`	List of providers (see A.4)	
	`ecu {}`	List of ECU definitions (see A.5)	M
	`dataType {}`	List of data type definitions (see A.6)	M
	`dataParam {}`	List of data parameter definitions (see A.7)	M
	`dataParamMapping {}`	List of data parameter mapping definitions (see A.8)	M
	`dtcBase {}`	List of DTC base definitions (see A.9)	M
	`dtcSymptom {}`	List of DTC symptom definitions (see A.10)	M
	`fixedValue {}`	List of DataParamValueMapping defining the fixed values (see A.14)	M

A.20 ComplexCondition

Table A.30 defines the ComplexCondition.

Table A.30 — ComplexCondition

	Name	Description	Cvt
Attributes	`simple`	A simple boolean condition (`true` or `false`)	C1[a]
	`paramCond`	A `DataParamCondition`	C2[a]
	`dtcCond`	A `DtcCondition` on ocurrence and on change of its complementary	C3[a]
	`not`	A `ComplexCondition` which will be negated	C4[a]
	`and {}`	A list of `ComplexConditions` which will be AND related	C5[a]
	`or {}`	A list of `ComplexConditions` which will be OR related	C6[a]
	`DataParamCondition`	A data parameter condition contains following attributes:	
	– `rvId`	— registered value identifier	M
	– `ecuId`	— registered ECU identifier	O
	– `arrayIndex`	— Index of an array for addressing, if the data parameter corresponding data type is an array	O
	– `operator`	— An operator of following types: eq, ne, gt, lt, absgt, abslt, onChange, defined, i.e. equal, not equal, greater than, less than,absolute greater than, absolute less than, on change, if defined.	M

[a] One of C1, C2, C3, C4, C5 or C6 must be defined.

[b] One of C0, C1, C2, C3, C4, C5, C6, C7 or C8 shall be defined, dependent on the parameter definition.

Table A.30 *(continued)*

	– value	— Choice of data parameter's value dependent on its data type:	M
	– numeric	— A numeric value	C0[b]
	– lnumeric	— A long numeric value	C1[b]
	– string	— A string value	C2[b]
	– displayName	— A display name value	C3[b]
	– enumString	— A numeric value referencing an enumeration string item	C4[b]
	– bitString	— A numeric value using the bits of the bit string items	C5[b]
	– array	— Number of elements in the array as UNUM16	C6[b]
	– error	— An error ID	C7[b]
	– dataParam	— A reference to another parameter for comparison	C8[b]
	DtcCondition	A DTC condition contains following attributes:	
	– rDtcBaseId	DTC base identifier	M
	– rDtcSymptomId	DTC symptom identifier	M
	– complementary	A bit combination of values (see A.15)	M

[a] One of C1, C2, C3, C4, C5 or C6 must be defined.

[b] One of C0, C1, C2, C3, C4, C5, C6, C7 or C8 shall be defined, dependent on the parameter definition.

A.21 IDs

A lot of IDs are used for referencing other data objects. Table A.31 defines the most important IDs.

Table A.31 — ID definition

	Name	Type	Description
Attributes	textId	Identifier:: = SNUM32 (4 Byte Integer value)	Unique text identifier of a DisplayName (see A.17). In most of the data objects with a name, a Display-Names is used for reference. Normally, this direct included DisplayName only contains the textId. The languagePack of the corresponding configuration contains the DisplayName, including the language specific shortname and longname.
	dataTypeId	Identifier	Identifier of the data type (see A.6.1). The dataTypeId is only referenced by a data parameter (see A.7).
	rvId	Identifier	Registered value identifier of a data parameter (see A.7). The rvId always references the data parameter, not the value. It is referenced in Dat-aParamMapping, DataParamValueMapping and DataParamCondition.

Annex B
(normative)

Data eXchange Message definition

B.1 DXMessage

A DXMessage is the ASN.1 structured container for ITS PDUs communicated between ITS stations. The DXMessage is encoded in unaligned Packed Encoding Rules (U-PER). Table B.1 defines the DXMessage. See C.2 for the DXM ASN.1 definition.

Table B.1 — Definition of DXMessage

Msg	DXMessage	Used for all messages. There is no distinction between client and server message.	
Attributes	**Name**	**Description**	**Cvt**
	iTSmsID	ITS message identifier (see ISO/TS 17419) of type Identifier	M
	timeInMillis	This parameter includes the time stamp of the DXMessage in milliseconds since 1970.	O
	value {}	List of data parameter values (see A.11)	M
	dtcInfo {}	List of DtcInfo object containing attributes (see A.15)	O

ASN.1	DXMessage :: = SEQUENCE {
	iTSmsID Identifier,
	timeInMillis UNUM32 OPTIONAL,
	value SEQUENCE OF DataParamValue,
	dtcInfo SEQUENCE OF DtcInfo OPTIONAL,
	...
	}		

B.2 DXMConfig

A DXMConfig is the ASN.1 structured container for DXM configurations. The DXMConfig is encoded in unaligned Packed Encoding Rules (U-PER). Table B.1 defines the DXMConfig. See C.2 for the ASN.1 definition.

Table B.2 — DXMConfig definition

Attributes	**Name**	**Description**	**Cvt**
	configName	Name of the DXMConfig	M
	configVersion	Version of the DXMConfig (i.e. date, number)	M
	subject	Subject of the DXMConfig (see B.3)	M
	msg {}	List of DXMsg definitions (see B.4)	M

B.3 DXMSubject

Table B.3 defines the subject of the DXM configuration.

Table B.3 — DXMSubject definition

	Name	Description	Cvt
Attributes	subjectId	Subject identifier	M
	name	Display name of the message	M
Example	{ subjectId 1, name { textId 12100, longname "road guidance" }}		

B.4 DXMsg

A DXMsg defines all possible DX messages with their name, type and executer in Table B.4.

Table B.4 — DXMsg definition

	Name	Description	Cvt
Attributes	iTSmsID	ITS message identifier	M
	name	Display name of the message	M
	type	Type of the message: unknown, info, warning, alarm.	M
	executer	Executer of the message as a bit mask of following values: p-its-s (0),v-its-s (1),r-its-s (2),c-its-s (3)	M
	dataParamList {}	List of contained data parameter rvIds	M
Example	{ iTSmsID 221, name { textId, longname "notify-on-position" }, type info, executer '1100'B, dataParamList { 1, 2, 6, 22 } }, { iTSmsID 223, name { textId 12103, longname "search-parking-space" }, type info, executer '1100'B, dataParamList { 2, 22, 40, 41 } }, { iTSmsID 226, name { textId 12106, longname "collision-possible" }, type warning, executer '0010'B, dataParamList { 28, 2, 31 } }		

Annex C
(normative)

Data eXchange Message ASN.1 definition

C.1 Vehicle to interface protocol data (VIDF) ASN.1 definition

```
VIDF { iso(1) standard(0) ugp(13185) vidf(0) version1(1) } -VehicleInterfaceDataFormat
DEFINITIONS AUTOMATIC TAGS :: = BEGIN

--EXPORTS ALL;

-- ### Basic data types ###
Boolean        :: = BOOLEAN
OctetString    :: = OCTET STRING
String         :: = VisibleString
SNUM8          :: = INTEGER(-128..127)
UNUM8          :: = INTEGER(0..255)
SNUM16         :: = INTEGER(-32768..32767)
UNUM16         :: = INTEGER(0..65535)
SNUM32         :: = INTEGER(-2147483648..2147483647)
UNUM32         :: = INTEGER(0..4294967295) -really 8 not 4 byte
--SNUM64         :: = INTEGER(-9223372036854775808..9223372036854775807)
UNUM64         :: = INTEGER(0..9223372036854775807)
Version        :: = INTEGER(0..255)
Identifier     :: = SNUM32

oidError      OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) error    (0) }
oidUnitType   OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) unittype (1) }
oidUnit       OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) unit     (2) }
oidProvider   OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) provider (3) }
oidEcu        OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) ecu      (4) }
oidDataType   OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) datatype (5) }
oidDataParam  OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) dataparam(6) }
oidDtcBase    OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) dtcbase  (7) }
oidDtcSymptom OBJECT IDENTIFIER :: = { iso(1) standard(0) ugp(13185) def(1) dtcsymptom(8)
}

-- ### data ### ------------------------------------
DisplayName :: = SEQUENCE {
    textId              Identifier,
    shortname           UTF8String                          OPTIONAL,
    longname            UTF8String                          OPTIONAL,
    ...
}
VIError :: = SEQUENCE {
    errorId             Identifier,
    name                DisplayName,
    attributeCount      UNUM16,
    ...
}
UnitType :: = SEQUENCE {
       unitTypeId          UNUM16,
       name                DisplayName
}
Unit :: = SEQUENCE {
    unitTypeId          UNUM16,
    unitId              UNUM16,
    name                DisplayName,
    formula             UNUM8,
    c0                  SNUM32,
    c1                  SNUM32,
    c2                  SNUM32,
    ...
```

```
}
Provider :: = SEQUENCE {
    providerId          Identifier,
    name                DisplayName
}
Ecu :: = SEQUENCE {
    ecuId               Identifier,
    name                DisplayName,
    providerId          Identifier,
    parentId            Identifier                              OPTIONAL,
    ...
}
DataType :: = SEQUENCE {
    dataTypeId          Identifier,
    name                DisplayName                              OPTIONAL,
    type                CHOICE {
      numeric             Numeric,
      lnumeric            LNumeric,
      string              LimitedString,
      displayName         NULL,
      enumString          SEQUENCE OF EnumStringItem,
      bitString           SEQUENCE OF BitStringItem,
      structure           Structure,
      array               UNUM16,
      monitor             SEQUENCE OF MonitorItem,
      octet               SNUM32,
      ...
    }
}
Numeric :: = SEQUENCE {
    decimalPlaces       UNUM8                                    OPTIONAL,
    unitId              UNUM16,
    factor              SNUM16                                   DEFAULT 1,
    quotient            SNUM16                                   DEFAULT 1,
    addend              SNUM16                                   DEFAULT 0,
    min                 SNUM16                                   OPTIONAL,
    max                 SNUM16                                   OPTIONAL,
    ...
}
LNumeric :: = SEQUENCE {
    decimalPlaces       UNUM8                                    OPTIONAL,
    unitId              UNUM16,
    factor              SNUM32                                   DEFAULT 1,
    quotient            SNUM32                                   DEFAULT 1,
    addend              SNUM32                                   DEFAULT 0,
    min                 SNUM32                                   OPTIONAL,
    max                 SNUM32                                   OPTIONAL,
    ...
}
LimitedString :: = SEQUENCE {
    allowedCharacters   String                                   OPTIONAL,
    minLen              UNUM16                                   OPTIONAL,
    maxLen              UNUM16                                   OPTIONAL,
    ...
}
EnumStringItem :: = SEQUENCE {
    value               UNUM16,
    name                DisplayName,
    ...
}
BitStringItem :: = SEQUENCE {
    bit                 UNUM16,
    name                DisplayName,
    ...
}
Structure :: = SEQUENCE {
    param               SEQUENCE OF Identifier,
    convention          Convention                               DEFAULT mandatory,
    ...
}
MonitorItem :: = SEQUENCE {
```

```
    testId              UNUM16,
    decimalPlaces       UNUM8,
    unitId              UNUM16,
    factor              SNUM16                                  DEFAULT 1,
    quotient            SNUM16                                  DEFAULT 1,
    addend              SNUM16                                  DEFAULT 0,
    ...
}
DataParam :: = SEQUENCE {
    rvId                Identifier,
    name                DisplayName,
    dataTypeId          Identifier,
    accessType          AccessType,
    description         DisplayName                              OPTIONAL,
    dataParamProperty   DataParamProperty,
    parentId            Identifier                              DEFAULT 0,
    ...
}
DataParamMapping :: = SEQUENCE {
    rvId                Identifier,
    ecuId               Identifier,
    arrayIndex          UNUM8                                    OPTIONAL,
    ...
}
DtcBase :: = SEQUENCE {
    rDtcBaseId          Identifier,
    providerId          Identifier                              OPTIONAL,
    ecuId               Identifier                              OPTIONAL,
    name                DisplayName,
    description         DisplayName                              OPTIONAL,
    dataParamList       SEQUENCE OF Identifier                  OPTIONAL,
    dataParamMapping    SEQUENCE OF DataParamMapping             OPTIONAL,
    ...
}
DtcSymptom :: = SEQUENCE {
    rDtcSymptomId       UNUM16,
    providerId          Identifier                              OPTIONAL,
    ecuId               Identifier                              OPTIONAL,
    name                DisplayName,
    description         DisplayName                              OPTIONAL,
    ...
}
LanguagePack :: = SEQUENCE {
    language            String,
    text                SEQUENCE OF DisplayName,
    ...
}

-- ### values ### ------------------------------------
DataParamValue :: = CHOICE {
    numeric             SNUM16,
    lnumeric            SNUM32,
    string              String,
    displayName         DisplayName,
    enumString          UNUM16,
    bitString           UNUM16,
    structureMissing    UNUM8,
    array               UNUM16,
    monitor             SEQUENCE OF MonitorValue,
    octet               OctetValue,
    error               Identifier,
    ...
}
MonitorValue :: = SEQUENCE {
    testValue           SNUM32,
    testValueMin        SNUM32                                   OPTIONAL,
    testValueMax        SNUM32                                   OPTIONAL,
    ...
}
OctetValue :: = SEQUENCE {
    length              SNUM32                                   OPTIONAL,
```

```
    data                OctetString,
       ...
}
VIErrorValue :: = SEQUENCE {
    errorId             Identifier,
    attribute           SEQUENCE OF DataParamValue             OPTIONAL,
    ...
}
DataParamValueTS :: = SEQUENCE { - time stamp
    value               DataParamValue,
    timeInMillis        UNUM64                                 OPTIONAL,
    ...
}
DataParamValueMapping :: = SEQUENCE {
    rvId                Identifier,
    ecuId               Identifier                            OPTIONAL,
    value               DataParamValue,
    timeInMillis        UNUM64                                 OPTIONAL,
    ...
}
DtcInfo :: = SEQUENCE {
    rDtcBaseId          Identifier,
    rDtcSymptomId       UNUM16,
    ecuId               Identifier                            OPTIONAL,
    complementary       DtcComplementary,
    envData             SEQUENCE OF DataParamValueTS           OPTIONAL,
    timeInMillis        UNUM64                                 OPTIONAL,
    ...
}

-- ### configuration ### ---------------------------------
VIDFConfig :: = SEQUENCE {
    configName          String,
    vehicleInfo         String                                 OPTIONAL,
    error               SEQUENCE OF VIError,
    unitType            SEQUENCE OF UnitType,
    unit                SEQUENCE OF Unit,
    provider            SEQUENCE OF Provider,
    ecu                 SEQUENCE OF Ecu,
    dataType            SEQUENCE OF DataType,
    dataParam           SEQUENCE OF DataParam,
    dataParamMapping    SEQUENCE OF DataParamMapping,
    dtcBase             SEQUENCE OF DtcBase,
    dtcSymptom          SEQUENCE OF DtcSymptom,
    fixedValue          SEQUENCE OF DataParamValueMapping,
    ...
}

-- ### conditions ### ---------------------------------
ComplexCondition :: = CHOICE {
    simple              BOOLEAN,
    paramCond           DataParamCondition,
    dtcCond             DtcCondition,
    not                 ComplexCondition,
    and                 SEQUENCE OF ComplexCondition,
    or                  SEQUENCE OF ComplexCondition,
    ...
}
DataParamCondition :: = SEQUENCE {
    rvId                Identifier,
    ecuId               Identifier                            OPTIONAL,
    arrayIndex          UNUM8                                  OPTIONAL,
    operator            OperatorType,
    value               DataParamCondValue,
    ...
}
OperatorType :: = ENUMERATED { eq, ne, gt, lt, absgt, abslt, onChange, defined, ...
}
DataParamCondValue :: = CHOICE {
    numeric             SNUM16,
    lnumeric            SNUM32,
```

```
    string              String,
    displayName         DisplayName,
    enumString          UNUM16,
    bitString           UNUM16,
    array               UNUM16,
    error               Identifier,
    dataParam           DataParamMapping,
    ...
}
DtcCondition :: = SEQUENCE {
    rDtcBaseId          Identifier,
    rDtcSymptomId       UNUM16,
    complementary       DtcComplementary                    OPTIONAL,
    ...
}

-- ### helper ### ------------------------------------
Convention :: = ENUMERATED { mandatory, optional, conditional, ...
}
AccessType :: = BIT STRING {
    r   (0),
    w   (1),
    x   (2),
    i   (3),
    u   (4)
} (SIZE(5, ...))
DataParamProperty :: = ENUMERATED { ecu-supported-info, sensor, actuator,
    ecu-internal-signal, ecu-internal-monitor, collection, routine, fix, other, ...
}
DtcComplementary :: = BIT STRING {
    testFailed                          (0),
    testFailedThisOperationCycle        (1),
    pendingDTC                          (2),
    confirmedDTC                        (3),
    testNotCompletedSinceLastClear      (4),
    testFailedSinceLastClear            (5),
    testNotCompletedThisOperationCycle  (6),
    warningIndicatorRequested           (7),
    permanent                           (8),
    maintenanceOnly                     (9),
    checkAtNextHalt                    (10),
    checkImmediately                   (11),
    dTCClass0                          (12),
    dTCClass1                          (13),
    dTCClass2                          (14),
    dTCClass3                          (15),
    dTCClass4                          (16)
} (SIZE(17, ...))

END
```

C.2 DXM ASN.1 definition

```
DXM { iso(1) standard(0) dxm(13184) message(0) version1(1) } --DataExchangeMessage
DEFINITIONS AUTOMATIC TAGS :: = BEGIN

EXPORTS dxmVersion, DXMessage;
IMPORTS UNUM64, DataParamValue, Identifier, Version, DtcInfo, DisplayName
   FROM VIDF { iso(1) standard(0) ugp(13185) vidf(0) version1(1) };

dxmVersion Version :: = 1
oidDXMSubject OBJECT IDENTIFIER :: = { iso(1) standard(0) dxm(13184) def(1) subject (0) }
oidDXMsg      OBJECT IDENTIFIER :: = { iso(1) standard(0) dxm(13184) def(1) msg     (1) }

DXMessage :: = SEQUENCE {
    iTSmsID             Identifier,
    timeInMillis        UNUM64                              OPTIONAL,
    value               SEQUENCE OF DataParamValue,
    dtcInfo             SEQUENCE OF DtcInfo                 OPTIONAL,
    ...
}
```

```
DXMConfig :: = SEQUENCE {
    configName          String,
    configVersion       String,
    subject              DXMSubject,
    msg                  SEQUENCE OF DXMsg,
    ...
}
DXMSubject :: = SEQUENCE {
    subjectId           Identifier,
    name                DisplayName,
    ...
}
DXMsg :: = SEQUENCE {
    iTSmsID             Identifier,
    name                DisplayName,
    type                DXMsgType,
    executer            DXMsgExecuter,
    dataParamList       SEQUENCE OF Identifier,
    ...
}
DXMsgType :: = ENUMERATED { unknown, info, warning, alarm, ...
}
DXMsgExecuter :: = BIT STRING {
    p-its-s   (0),
    v-its-s   (1),
    r-its-s   (2),
    c-its-s   (3)
} (SIZE(4, ...))

END
```

Annex D
(informative)

Requirements

D.1 Communication requirements

Table D.1 defines the Communication requirements.

Table D.1 — Communication requirements

	Parameter name	Value
Attributes	LogicalChannelType	CCH or SCH
	ContConnect	60 s
	NxRepeat	3 s, 2 s
	FlowType	
	PortNoInfo	?, 8082
	DestType	broadcast
	DestDomain	itsNWlocal, gobal
	CommDistance	500 m
	Directivity	fixed, ?, ...
	Resilience	true
	MinThP	3/sec
	MaxLat	0
	MaxADU	1 500
	DataConfidentiality	100
	DataIntegrity	150
	ReqNonRepudiation	70
	SourceAuthentication	255
	ProtocolReq	

Bibliography

[1] ISO/IEC 10731:1994, *Information technology — Open Systems Interconnection — Basic Reference Model — Conventions for the definition of OSI services*

[2] ISO 13185-2, *Intelligent transport systems — Vehicle interface for provisioning and support of ITS services — Part 2: Unified gateway protocol (UGP) requirements and specification for vehicle ITS station gateway (V-ITS-SG) interface*

[3] ISO/TS 17429, *Intelligent transport systems (ITS) — Co-operative systems — ITS station facilities for the transfer of information between ITS stations*

[4] ISO 24102–6, *Intelligent Transport Systems (ITS) — Communications Access for Land Mobiles (CALM) — ITS station management — Part 6: Path and flow management*

[5] ETSI EN 302 637-2, *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*

[6] SAE J2735, *Dedicated Short Range Communications (DSRC) Message Set Dictionary*

[7] ETSI/TS 101 539-1, *Intelligent Transport Systems (ITS); V2X Applications; Part 1: Road Hazard Signalling (RHS), application requirements specification*

ICS 03.220.01; 35.240.60

Price based on 83 pages