ILNAS

Institut luxembourgeois de la normalisation
de l'accréditation, de la sécurité et qualité
des produits et services

ILNAS-EN 15722:2020

Intelligent transport systems - ESafety - ECall minimum set of data

Systèmes de transport intelligents - ESafety - Ensemble minimal de données (MSD) pour l'eCall

Intelligente Verkehrssysteme - eSicherheit - Minimaler Datensatz für den elektronischen Notruf eCall

08/2020

National Foreword

This European Standard EN 15722:2020 was adopted as Luxembourgish Standard ILNAS-EN 15722:2020.

Every interested party, which is member of an organization based in Luxembourg, can participate for FREE in the development of Luxembourgish (ILNAS), European (CEN, CENELEC) and International (ISO, IEC) standards:

- Participate in the design of standards
- Foresee future developments
- Participate in technical committee meetings

https://portail-qualite.public.lu/fr/normes-normalisation/participer-normalisation.html

EUROPEAN STANDARD
NORME EUROPÉENNE
EUROPÄISCHE NORM

EN 15722

August 2020

ICS 03.220.20; 13.200; 35.240.60

Supersedes EN 15722:2015

English Version

Intelligent transport systems - ESafety - ECall minimum set of data

Systèmes de transport intelligents - ESafety - Ensemble minimal de données (MSD) pour l'eCall

Intelligente Transportsysteme - ESicherheit - Minimaler Datensatz für den elektronischen Notruf eCall

This European Standard was approved by CEN on 5 July 2020.

This European Standard was corrected and reissued by the CEN-CENELEC Management Centre on 4 November 2020.

CEN members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration. Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the CEN-CENELEC Management Centre or to any CEN member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CEN member into its own language and notified to the CEN-CENELEC Management Centre has the same status as the official versions.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

Ref. No. EN 15722:2020 E

Contents

Page

European foreword

This document (EN 15722:2020) has been prepared by Technical Committee CEN/TC 278 “Intelligent transport systems”, the secretariat of which is held by NEN.

This European Standard shall be given the status of a national standard, either by publication of an identical text or by endorsement, at the latest by February 2021, and conflicting national standards shall be withdrawn at the latest by February 2021.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN shall not be held responsible for identifying any or all such patent rights.

This document supersedes EN 15722:2015.

In comparison with the previous edition, the following modifications have been made:

— Correction of some typing errors;

— Added additional clarifications to solve frequently asked questions;

— Inclusion of recent locations mandatory to support more efficient dispatch of emergency services;

— MSD field “numberOfPassengers” replaced by “numberOfOccupants”;

— The number of vehicle categories supported by this standard has been expanded through revision of the enumeration values to enable support for additional categories of vehicles, which now covers the full UNECE categorization;

— Updated privacy requirements to include EU 2016/679 GDPR.

According to the CEN-CENELEC Internal Regulations, the national standards organisations of the following countries are bound to implement this European Standard: Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.

Introduction

The pan-European in-vehicle emergency call, 'eCall', is estimated to have the potential to save up to 2 500 fatalities annually in the EU when fully deployed, and furthermore to reduce the severity of injuries, to bring significant savings to the society in and to reduce human suffering.

Emergency calls made from vehicles or mobile telephones using wireless technologies, can assist with the objectives of significantly reducing road deaths and injuries, but drivers often have poor (imprecise) location awareness, especially on interurban roads or abroad. Additionally, in many situations the car occupants may not be in a position to call using a normal mobile phone.

The situation is worse for those travelling abroad. A high (and increasing) number of vehicles travelling outside their home country is thus also contributing to the need for automated emergency call system in vehicles. In EU there are over 100 million trips to another EU country per year, 65 % of the people feel less protected while abroad and most do not know which number to call in an emergency (in some countries over 60 %). Language problems are pertinent and may render proper communication difficult. Yet, in the most crucial cases, the victim(s) may not be able to call because they have been injured/trapped, do not know the local number to call, and in many cases, particularly in rural situations and late at night, there may be no witnesses who happen to have a mobile phone and a sense of community.

eCall, in the context of "Intelligent Transport Systems" or "ITS", (previously known as "Road Traffic and Transport Telematics") can be described as a "user instigated or automatic system to provide notification to public safety answering points, by means of wireless communications, that a vehicle has crashed, and to provide coordinates and a defined minimum set of data, and where possible a voice link to the PSAP".

The objective of implementing the pan-European in-vehicle emergency call system (eCall) is to automate the notification of a traffic accident, wherever in the European Union and associated countries, with the same technical standards and the same quality of services objectives of other emergency services (for example the TS12 emergency call of GSM/UMTS).

This document specifies the "Minimum Set of Data" (MSD) to be transferred by such an in-vehicle eCall system in the event of a crash or emergency.

NOTE The communications media and means of transferring the eCall MSD are not defined in this document. See list of referenced standards.

1 Scope

This document specifies the standard data concepts that comprise the "Minimum Set of Data" (MSD) to be transferred from a vehicle to a 'Public Safety Answering Point' (PSAP) in the event of a crash or emergency via an 'eCall' communication transaction.

Optional additional data concepts may also be transferred as part of the MSD.

The communications media protocols and methods for the transmission of the eCall message are not specified in this document.

2 Normative references

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

EN 16062, *Intelligent transport systems — ESafety — eCall high level application requirements (HLAP) using GSM/UMTS circuit switched networks*

EN 16102, *Intelligent transport systems — eCall — Operating requirements for third party support*

ISO/IEC 8825-2, *Information technology — ASN.1 encoding rules: Specification of Packed Encoding Rules (PER) — Part 2:*

NOTE Communications standards required for transmission of eCall using GSM/UMTS wireless communications networks are referenced in EN 16062 and EN 16072 [6].

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

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

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

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

3.1
ASN.1
Abstract Syntax Notation One
notation that describes rules and structures for representing, encoding, transmitting, and decoding data enabling representation of objects that are independent of machine-specific encoding techniques; see Annex B

3.2
eCall
emergency call generated either automatically via activation of in-vehicle sensors or manually by the vehicle occupants; when activated it provides notification and relevant location information to the most appropriate 'Public Safety Answering Point', by means of mobile wireless communications networks, carries a defined standardized 'Minimum Set of Data' notifying that there has been an incident that requires response from the emergency services, and establishes an audio channel between the occupants of the vehicle and the most appropriate 'Public Safety Answering Point'

3.3
MSD
minimum set of data
direct, timely data content of an eCall message to the PSAP operator receiving the emergency call containing information about the location of the incident, providing detail characterising the vehicle, and potentially sometimes also providing additional data that is deemed relevant

3.4
PSAP
public safety answering point
'first level' responder to whom an emergency call/eCall is directed

4 Symbols and abbreviated terms

For the purposes of this document, the following symbols and abbreviated terms apply.

ASN.1	abstract syntax notation one (ISO/IEC 8824, ISO/IEC 8825)
3G	third generation mobile cellular network system, defined by 3GPP standards
3GPP	third generation partnership project
BCD	binary coded decimal
BER	basic encoding rules (ASN.1)
CNG	compressed natural gas
CXER	Canonical XML encoding rules
ETSI	European telecommunications standards institute
EC	European Commission
EU	European Union
EXER	extended XML encoding rules
GSM	global system for mobile communications
GNSS	global navigation satellite system
ID	identity
IP	Internet protocol
ISO	international organization for standardization
ITS	intelligent transport system(s)
ITU	international telecommunication union
IVS	in-vehicle system
LPG	liquid propane gas
M	mandatory
MSD	minimum set of data
O	optional
OID	object identifier (ISO/IEC 8824) - see Annex E
P2WV	powered 2-wheel vehicles

PDU	protocol data unit (ASN.1)
PER	packed encoding rules (ASN.1)
PSAP	public safety answering point
TPSP	third party service provider
UMTS	universal mobile telecommunications system
UPER	unaligned packet encoding rules (ASN.1)
VDS	vehicle type descriptor (part of VIN)
VIN	vehicle identification number
VIS	vehicle identification sequence (part of VIN)
WMI	world manufacturer index (part of VIN)
WGS84	world geodetic system
XER	XML encoding rules
XML	extensible markup language
XSD	XML Schema Definition

5 Requirements

5.1 Concepts and formats

5.1.1 MSD data concepts

NOTE The minimum set of data is important information to assist the provision of the most appropriate services to the crash or emergency site and to speed up the response. The minimum set of data makes it possible for the PSAP operator to respond to the eCall even without the voice connection.

The "Minimum Set of Data" shall be a direct, timely message to the PSAP operator receiving the emergency call.

The information elements in the MSD have been selected on the basis of their relevance in an emergency rescue situation.

The MSD has an ‘optional additional data’ block that can be used to add information elements that are relevant to a specific situation. See 5.1.5.

5.1.2 Representation of MSD data concepts

The message shall be sent in the sequence defined within the ASN.1 definition defined in Annex A.

The transferred MSD for Pan-European eCall shall be represented in Abstract Syntax Notation (ASN.1) using the ‘Unaligned Packed Encoding Rules’ (UPER) as defined in ISO/IEC 8825-2, using the ASN1 definitions found in Annex A. See also 5.1.4.

The transfer of the MSD for Pan-European eCall using other wireless communications media (for example E- UTRAN) may be specified in future standards for ‘high level application protocols’ for that wireless media.

NOTE 1 An XML encoding of the MSD data representation can be used in TPSP-to-PSAP applications (EN 16102). Annex C contains the derived XSD for such encoding.

NOTE 2 In order to implement presentation in ASN.1 UPER, readers are advised to also read Annex B "ASN.1 Data Representation PER and BER explained"; and also the relevant normative referenced documents.

5.1.3 Different versions of MSD data

It is foreseen that, over time, new versions of the MSD data definition will occur. Wherever possible, later versions of the MSD shall be backwards compatible with existing versions.

If a future version of the MSD is defined which is not backwards compatible (i.e. cannot be automatically interpreted by receiving systems) then its deployment shall be coordinated to ensure that all receiving systems are ready before IVS adopt this new MSD format.

The main structure of an MSD shall, in any version, contain two elements, the first of which is known as the MSD version (msdVersion) which designates the encoding rules that have been used to create the second element.

Systems receiving an MSD shall support all standardized MSD versions, which are each uniquely identified using this msdVersion parameter.

5.1.4 Distribution of MSD data

The MSD shall be transmitted using one or more communications media as defined in other eCall Standards.

In order to enable interpretation by the PSAP, the MSD shall always be presented in an ASN.1 encoded module: either ASN.1 'Unaligned Packet Encoding Rules' (UPER) or ASN.1 'Extended XML Encoding Rules' (EXER) encoding shall be used.

The ASN.1 module shall contain the MSD as defined in this document plus none or more 'optional additional data' concepts presented as defined in 5.1.5 and whose name, content and presentation has been made available in a data registry as required by this document (See 5.1.5).

In the case of an MSD for pan-European eCall it shall be encoded using 'Unaligned Packed Encoding Rules' (UPER) as defined in ISO/IEC 8825-2. The length of this encoded MSD (including any 'optional additional data') shall not exceed 140 bytes. Any payload bytes received outside of the ASN.1 message length shall be ignored by the receiving entity.

NOTE 1 It is assumed that the integrity of the transmitted data is assured by the underlying communication interface standard used. For example, EN 16072 [6]which defines the operating requirements for the transmission of Pan-European eCall and EN 16062 (eCall high level application protocols for GSM/UMTS) which provide the high level application protocols for sending a Pan-European eCall via a circuit switched GSM/UMTS wireless phone network.

EN 16102 defines provisions for Third Party supported eCall.

NOTE 2 If the MSD is transferred using another means of communication that has no, or less stringent, data limits, XML encoding rules can be used if preferred. Annex C contains the derived XSD for such encoding.

5.1.5 Additional data

The MSD message has a provision for 'optional' additional data. This document specifies the presentation of any such data within an MSD message. The nature and content of such additional data is not part of this document.

EXAMPLE Additional data may contain a reference to an external source of relevant information (such as a phone number, a website URL, etc.) where further information may be found, or additional data specific to the vehicle or incident (e.g. battery temperature in the case of an electric or hybrid vehicle; number of rollovers; URL to the technical specifications to a particular vehicle model; etc.)

Optional additional data shall not include any data concerning or identifying a person (personal data) unless the transfer of such data has been explicitly and expressly prior instructed and authorized by the person who is identified by the data and its provision shall in any event only be provided only in accordance with European Union and National privacy regulations pertaining at the time of the transfer of any such personal data and in accordance with the provisions of EU 2016/679 'General Data Protection Requirements' [8].

Any additional data element(s) shall each consist of two parts:

1. a relative 'object identifier' (OID);

2. the data content.

The relative OID shall be allocated by CEN TC278 WG15 or a body nominated by it. For further information see Annex E.

CEN/TC 278/WG15 or a body nominated by it shall allocate an 'Object Identifier' (OID) for each 'oad'-optional additional data'-concept. Within the MSD the 'optional additional data'-concept used shall be identified by a 'relative OID', i.e. it will only contain the arcs of the object identifier of the concept starting below the eCall MSD 'optional additional data'-concept object identifier. See 5.2 below.

Additional data shall be represented using an ASN.1representation definition that itself is made available to emergency services/PSAPs.

When sending an MSD containing this additional data, using GSM/UMTS (EN 16062), the addition of such data shall never cause the total (UPER encoded) MSD message length to exceed the maximum available number of bytes (total message length = 140 bytes).

In order to ensure that the above requirement is met with any combination of optional parameters within the main MSD message, the total length of additional data concepts may not exceed 94 bytes of data encoded in ASN.1 UPER.

5.2 ISO Object identifier

ISO/ITU "Object Identifiers" are explained in informative Annex E.

The full eCall MSD, or any 'optional additional data'-concept, is preceded by its ISO object identifier. When eCall data is stored or used outside of the eCall context this OID shall be prefixed onto all representations of the MSD or any eCall data concept.

In eCall context, when data is being sent to a specific receiver (e.g. PSAP), the OID may be assumed to be known and is not transmitted. Thus the OID is not transferred over the air between the IVS and PSAP.

eCall has been allocated the OID: 1.0.14817.106.2.1

EXPLANATION

1. identifies the data concept as an ISO parent route standard

0. identifies the arc as being identified by a Standards reference number.

14817 In this case ISO 14817 being the parent standard for ITS data registry

106 emergency-service

2 pre-harmonisation-automated-calls

1 cen-15722

Below this OID three nodes are defined:

1.0.14817.106.2.1.1 for ‘Mandatory Data Concepts’

1.0.14817.106.2.1.2 for ‘Optional Data Concepts’

1.0.14817.106.2.1.3 for eCall data elements

5.3 Contents of the ‘Minimum Set of Data’ (MSD)

5.3.1 General

The following sub-clauses provide the definition of the minimum set of data that shall be sent from the vehicle in case of an emergency call.

5.3.2 Basic contents of MSD version 3

Table 1 provides a summary of the semantic contents of the MSD.

The sequence of data presentation shall be as specified in Table 1, represented as described in 5.1.2 and distributed as described in 5.1.4.

For clarity the ‘type’ used in Table 1 is a semantic representation of the type used in the ASN.1 definition. The exact representation is defined in Annex A.

The real position of the element in the data-stream is defined by the ASN.1 ‘unaligned packet encoding rules (UPER), following the definition in Annex A. Elements therefore do not necessarily start or end on a byte boundary.

Table 1 — Contents/format of the MSD data concept

M – Mandatory data field
O – Optional data field

MSD				
msdVersion	INTEGER (1..255)	-	M	MSD format version The format described in this document carries version 3 *See 5.1.4 for detailed information.*
Msd				
msdStructure				
messageIdentifier	INTEGER (1..255)		M	Message identifier, starting with 1 for each new eCall transaction and to be incremented with every application layer MSD retransmission following a request to resend after the incident event
Control			M	
automaticActivation	BOOLEAN			true = Automatic activation false = Manual activation
testCall	BOOLEAN			true = Test call false = Emergency
positionCanBeTrusted	BOOLEAN			true = Position can be trusted false = Low confidence in position "Low confidence in position" shall mean that there is less than 95% confidence that exact position is within a radius of ± 150 m of reported position
vehicleType	ENUM			The category of the vehicle according to UNECE Vehicle classification ECE-TRANS-WP29-78-r4e for type approval according to Directive 2007/46/EC of the European Parliament and of the Council as referenced in eCall Regulations, esp Commission Delegated Regulation (EU) 2017/79. The supported vehicle categories are: (Category M - Power-driven vehicles having at leas four wheels and used for the carriage of people) - Category M1 passenger vehicle - Category M2 buses and coaches - Category M3 buses and coaches (Category N - Power-driven vehicles having at least four wheels and used for the carriage of goods) - Category N1 light commercial vehicles - Category N2 heavy duty vehicles - Category N3 heavy duty vehicles (Category L – Motor vehicles with less than four wheels- but including light quads) - Category L1 P2WV - Category L2 three-wheeled vehicle - Category L3 P2WV

					- Category L4 three wheels asymmetrically arranged - Category L5 vehicle three wheels symmetrically - Category L6 four wheels limited power - Category L7 four wheels limited power 33(Trailers [including semi–trailers]) - Category O - (Agricultural vehicles) - Category T - Category R - Category S (off-road vehicles) - Category G - - Category "Other"
VIN[a]	VIN[1]		M		VIN number according to ISO 3779
vehiclePropulsionStorageType			M		*Contains information about the presence of propulsion storage inside the vehicle sending the MSD.*
gasolineTankPresent	BOOLEAN				true = present; false = not present If no information about the propulsion storage is known, all elements shall be set to FALSE.
dieselTankPresent	BOOLEAN				
compressedNaturalGas	BOOLEAN				
liquidPropaneGas	BOOLEAN				
electricEnergyStorage	BOOLEAN				
hydrogenStorage	BOOLEAN				
otherPropulsionStorage	BOOLEAN				
timeStamp	INTEGER (0..2^{32}-1)	sec	M		Timestamp of the initial data message generation within the current eCall incident event, represented in seconds elapsed since midnight January 1st, 1970 UTC. NOTE 1 The initial message generation immediately follows the eCall generation sequence subsequent to a (confirmed) trigger. NOTE 2 Subsequent transmissions within the given incident use the same timestamp, but the messageIdentifier changes. NOTE 3 Failure value for time stamp set to "0"
vehicleLocation			M		*The last known vehicle position determined at the latest moment possible before message generation.*
positionLatitude	INTEGER (-2^{31}..2^{31}-1)	milliarcsec			Position latitude (WGS84) EXPLANATION (calculation example): *48.3003333 = 48°18'1.20" N = 48*60*60.000" + 18*60.000" + 1.20" = 173881.200" = 173881200 milliarcsec* maximum value:

					90°00'00.000" = 324000000 minimum value: -90°00'00.000" = -324000000 If latitude is invalid or unknown, the representation of value 2147483647 shall be transmitted. If both latitude and longitude have value 0 then the location shall also be interpreted as invalid/unknown. NOTE If the transmitter or receiver determines either latitude or longitude to be invalid/unknown, then it is advised to transmit both longitude and latitude as unknown.
	positionLongitude	INTEGER (-2^{31}..2^{31}-1)	milliarcsec		Position longitude (WGS84) maximum value: 180°00'00.000'' = 648 000 000 minimum value: -180°00'00.000'' = -648 000 000 *See latitude for calculation example and notes.*
vehicleDirection		INTEGER (0..255)	2° (2 degree)	M	The vehicle's last known real direction of travel, expressed in 2°-degrees steps from (magnetic or geographical) north (0– 358, clockwise) determined at the latest moment possible before message generation. EXPLANATION (calculation example): *due North = 0° = 0 * 2° => 0* *due East = 90° = 45 * 2° => 45* *due South = 180° = 90 * 2° => 90* *due West = 270° = 135 * 2° => 135* The direction shall be unaffected by random fluctuations of GNSS signals. If direction of travel is invalid or unknown, the representation of value 255 shall be transmitted
recentVehicleLocationN1				M	*Known location of the vehicle 'some time' before the generation of the data for the MSD message.* The three readings (vehicleLocation, recentVehicleLocationN1 and recentVehicleLocationN2) shall be taken within a timeframe of no more than 15 sec. without the possibility to derive information about the driving speed at the time of triggering.
	latitudeDelta	INTEGER (-512..511)	100 milliarcsec		Latitude Delta (+ for North and – for South; WGS84) with respect to vehicleLocation. 1 Unit = 100 miliarcseconds, which is approximately 3m (on Earth) maximum value: 511 = 0°0'51.100'' (≈1580m) minimum value: -512 = -0°0'51.200'' (≈ -1583m)
	longitudeDelta	INTEGER (-512..511)	100 milliarcsec		Longitude Delta (+ for East and – for West; WGS84) with respect to vehicleLocation. *See latitudeDelta for details*

recentVehicleLocationN2			M	Known location of the vehicle 'some time' before recentVehicleLocationN1. The three readings (vehicleLocation, recentVehicleLocationN1 and recentVehicleLocationN2) shall be taken within a timeframe of no more than 15 sec. without the possibility to derive information about the driving speed at the time of triggering.
latitudeDelta	INTEGER (-512..511)	100 milliarcsec		Latitude Delta (+ for North and – for South) with respect to recentVehicleLocationN1. *See recentVehicleLocationN1. latitudeDelta for details*
longitudeDelta	INTEGER (-512..511)	100 milliarcsec		Longitude Delta (+ for East and – for West) with respect to recentVehicleLocationN2. *See recentVehicleLocationN1. latitudeDelta for details*
numberOfOccupants	INTEGER (0..255)		O	Number of occupants in the vehicle according to available information. If no information about the number of occupants is available, this parameter needs to be omitted or filled with the representation of value 255 NOTE 1 This information is indicative only as it may be not always be reliable in providing exact information about the number of occupants (e.g. because seatbelts may not be fastened by occupants or seatbelts may be fastened for other reasons). NOTE 2 For vehicle categories without enclosing (e.g. motorcycles) 'occupants' will be read as 'riders' or 'vehicle users'.
optionalAdditionalData			O	
oid	RELATIVE-OID			*See 5.1.5*
data	OCTET STRING			*See 5.1.5*

a) The field is named vehicleIdentificationNumber in the ASN.1 definition. The ASN.1 type VIN is defined in Annex A and codes for a correct representation of the World Manufacturer Index (WMI), the Vehicle Type Descriptor (VDS) and the Vehicle Identification Sequence (VIS) that make up a VIN number, taking into account the preconditions of each part.

5.3.3 Previous versions of MSD message

5.3.3.1 General

Previous versions of the MSD message shall be supported by PSAPs to ensure proper handling of legacy systems. Such versions shall be described with a table described and defined in the same way as Table 1.

5.3.3.2 MSD message version 1

MSD message version 1 has been withdrawn due to an erroneous ASN.1 data representation which might cause disruption in the handling of eCalls by PSAP systems. As a result this version shall not be supported by PSAPs and shall not be used by an IVS as of publication of EN 15722:2015.

5.3.3.3 MSD message version 2 (2015)

MSD message version 2 was introduced with the publication of EN 15722:2015. The decision to make recentVehicleLocationN1 and recentVehicleLocationN2 mandatory incurred a change that enforced a new version (3) in the 2020 release. That necessity made it possible to incorporate some other changes more efficiently as well.

Version 2, described in Table 2, shall no longer be implemented in eCall IVS devices, but it shall be implemented in receiving systems.

Table 2 — Contents/format of the MSD data concept version 2

M – Mandatory data field

O – Optional data field

MSD				
msdVersion	INTEGER (1..255)	-	M	MSD format version The format described in this table carries version 2 *See 5.1.4 for detailed information.*
Msd				
msdStructure				
messageIdentifier	INTEGER (1..255)		M	Message identifier, starting with 1 for each new eCall transaction and to be incremented with every application layer MSD retransmission following a new 'Send MSD' request after the incident event
Control			M	
automaticActivation	BOOLEAN			true = Automatic activation false = Manual activation
testCall	BOOLEAN			true = Test call false = Emergency
positionCanBeTrustd	BOOLEAN			true = Position can be trusted false = Low confidence in position "Low confidence in position" shall mean that there is less than 95% confidence that exact position is within a radius of ± 150 m of reported position
vehicleType	ENUM			The category of the vehicle according to UNECE Vehicle classification ECE-TRANS-WP29-78-r4e for type approval according to Directive 2007/46/EC of the European Parliament and of the Council as referenced in eCall Regulations, esp Commission Delegated Regulation (EU) 2017/79. The supported vehicle categories are: (Category M - Power-driven vehicles having at leas four wheels and used for the carriage of passengers) - Category M1 passenger vehicle - Category M2 buses and coaches - Category M3 buses and coaches (Category N - Power-driven vehicles having at least four wheels and used for the carriage of goods)

					- Category N1 light commercial vehicles - Category N2 heavy duty vehicles - Category N3 heavy duty vehicles (Category L – Motor vehicles with less than four wheels- but including light quads) - Category L1 P2WV - Category L2 three-wheeled vehicle - Category L3 P2WV - Category L4 three wheels asymmetrically arranged - Category L5 vehicle three wheels symmetrically - Category L6 four wheels limited power - Category L7 four wheels limited power
VIN[a]		VIN[1]		M	VIN number according to ISO 3779
vehiclePropulsionStorageType				M	*Contains information about the presence of propulsion storage inside the vehicle sending the MSD.*
	gasolineTankPresent	BOOLEAN			true = present; false = not present If no information about the propulsion storage is known, all elements shall be set to FALSE.
	dieselTankPresent	BOOLEAN			
	compressedNaturalGas	BOOLEAN			
	liquidPropaneGas	BOOLEAN			
	electricEnergyStorage	BOOLEAN			
	hydrogenStorage	BOOLEAN			
	otherPropulsionStorage	BOOLEAN			
timeStamp		INTEGER (0..2^{32}-1)	sec	M	Timestamp of the initial data message generation within the current eCall incident event, represented in seconds elapsed since midnight January 1st, 1970 UTC. NOTE 1 The initial message generation immediately follows the eCall generation sequence subsequent to a (confirmed) trigger. NOTE 2 Subsequent transmissions within the given incident use the same timestamp, but the messageIdentifier changes. NOTE 3 Failure value for time stamp set to "0"
vehicleLocation				M	*The last known vehicle position determined at the latest moment possible before message generation.*
	positionLatitude	INTEGER (-2^{31}..2^{31}-1)	milliarcsec		Position latitude (WGS84) EXPLANATION (calculation example): *48.3003333 = 48°18'1.20" N = 48*60*60.000" + 18*60.000" + 1.20" = 173881.200" = 173881200 milliarcsec* maximum value: 90°00'00.000" = 324000000

				minimum value: -90°00'00.000" = -324000000 If latitude is invalid or unknown, the representation of value 2147483647 shall be transmitted. If both latitude and longitude have value 0 then the location shall also be interpreted as invalid/unknown. NOTE If the transmitter or receiver determines either latitude or longitude to be invalid/unknown, then it is advised to transmit both longitude and latitude as unknown.
positionLongitude	INTEGER (-2^{31}..2^{31}-1)	milliarcsec		Position longitude (WGS84) maximum value: 180°00'00.000'' = 648 000 000 minimum value: -180°00'00.000'' = -648 000 000 *See latitude for calculation example and notes.*
vehicleDirection	INTEGER (0..255)	2° (2 degree)	M	The vehicle's last known real direction of travel (expressed in 2°-degrees steps from (magnetic or geographical) north (0– 358, clockwise) determined at the latest moment possible before message generation. EXPLANATION (calculation example): *due North = 0° = 0 * 2° => 0* *due East = 90° = 45 * 2° => 45* *due South = 180° = 90 * 2° => 90* *due West = 270° = 135 * 2° => 135* The direction shall be unaffected by random fluctuations of GNSS signals. If direction of travel is invalid or unknown, the representation of value 255 shall be transmitted
recentVehicleLocationN1			O	*Known location of the vehicle some time before the generation of the data for the MSD message.* The recent location shall be chosen such that they could normally assist the receiving party to confirm the current location of the vehicle in different driving environments such as city or motorway.
latitudeDelta	INTEGER (-512..511)	100 milliarcsec		Latitude Delta (+ for North and – for South; WGS84) with respect to vehicleLocation. 1 Unit = 100 miliarcseconds, which is approximately 3m (on Earth) maximum value: 511 = 0°0'51.100'' (≈1580m) minimum value: -512 = -0°0'51.200'' (≈ -1583m)
longitudeDelta	INTEGER (-512..511)	100 milliarcsec		Longitude Delta (+ for East and – for West; WGS84) with respect to vehicleLocation. *See latitudeDelta for details*

recentVehicleLocationN2			O	Known location of the vehicle some time before recentVehicleLocationN1. The recent location shall be chosen such that they could normally assist the receiving party to confirm the current location of the vehicle in different driving environments such as city or motorway.
latitudeDelta	INTEGER (-512..511)	100 milliarcsec		Latitude Delta (+ for North and – for South) with respect to recentVehicleLocationN1. *See recentVehicleLocationN1. latitudeDelta for details*
longitudeDelta	INTEGER (-512..511)	100 milliarcsec		Longitude Delta (+ for East and – for West) with respect to recentVehicleLocationN2. *See recentVehicleLocationN1. latitudeDelta for details*
numberOfPassengers	INTEGER (0..255)		O	Number of occupants in the vehicle according to available information. If no information about the number of occupants is available, this parameter needs to be omitted or filled with the representation of value 255 NOTE 1 This information is indicative only as it may be not always be reliable in providing exact information about the number of passengers (e.g. because seatbelts may not be fastened by passengers or seatbelts may be fastened for other reasons). NOTE 2 For vehicle categories without enclosing (e.g. motorcycles) 'occupants' will be read as 'riders' or 'vehicle users'.
optionalAdditionalData			O	
oid	RELATIVE-OID			*See 5.1.5*
data	OCTET STRING			*See 5.1.5*

a) The field is named vehicleIdentificationNumber in the ASN.1 definition. The ASN.1 type VIN is defined in Annex A and codes for a correct representation of the World Manufacturer Index (WMI), the Vehicle Type Descriptor (VDS) and the Vehicle Identification Sequence (VIS) that make up a VIN number, taking into account the preconditions of each part.

NOTE See EN 15722:2015 for further information regarding MSD version 2, including its ASN.1 definition and examples.

Annex A
(normative)

ASN.1 definition of MSD

A.1 ASN.1 definition of MSD

This module definition is appropriate for transmission of MSD via Pan-European eCall (via any means of communication, see before) and may be used for transmission of MSD via EN 16102 (Operating requirements for third party support).

```
MSDASN1Module

-- Definition of the eCall related MSD message in ASN.1
-- Any MSD message will encoded using this scheme, following the
-- UPER encoding rules.
--
-- The MSD message is defined in CEN standard EN 15722.
-- Comments in this definition are taken from that standard. In
-- case of inconsistency in the comment, the text of EN 15722
-- prevails.

DEFINITIONS
AUTOMATIC TAGS ::=
BEGIN

-- Version of this ASN.1 MSD specification
-- (inclusion of this element allows software developers to
-- automatically read out the current version number from the ASN.1
-- compilation for automatic inclusion into the msdVersion parameter
-- of the MSD message, i.e. can reduce the chance of using an ASN.1
-- description of one version but saying it is another)

CurrentVersion::= INTEGER (3)

-- ECallMessage is the top level information element
-- The ECallMessage structure supports only one message type (msd)
-- Extendibility at this level is not allowed, thus ensuring that the
-- msdVersion (message format version) can be extracted directly.
-- Elements:
--   msdVersion:   MSD format version
--                 The format described in this document carries version 3
--   msd:          Minimum Set Of Data uplink from vehicle
--
-- The OCTET STRING (CONTAINING ...) construct is used to ensure that any
-- implementation can extract the msdVersion value from any version,
-- without decoding errors.

ECallMessage ::= SEQUENCE {
   msdVersion  INTEGER(0 .. 255),
   msd         OCTET STRING (CONTAINING MSDMessage)
}


-- The main uplink msd message from the vehicle
-- Elements:
--   msdStructure: The main MSD structure
--   optionalAdditionalData: Additional data
```

```
-- Extendable in future versions at this level e.g. to add extra data
--

MSDMessage ::= SEQUENCE {
   msdStructure           MSDStructure,
   optionalAdditionalData AdditionalData OPTIONAL,
  ...
}

-- The main MSD structure, excluding additional data
-- Elements:
--   messageIdentifier: Message identifier, starting with 1 for each
--                      new eCall session and to be incremented with
--                      every application layer MSD retransmission
--   control:           see ControlType
--   vehicleIdentificationNumber:  see VIN
--   vehiclePropulsionStorageType: see VehiclePropulsionStorageType
--   timestamp: Timestamp of incident event
--              As seconds elapsed since midnight January 1st, 1970 UTC.
--              Failure value for time stamp set to "0"
--   vehicleLocation:  see VehicleLocation
--   vehicleDirection: Direction of travel
--                     in 2°-degrees steps from either magnetic or
--                     geographical north (0- 358, clockwise)
--                     If direction of travel is invalid or unknown,
--                     the value 255 shall be used
--                     Only values from 0 to 179 and 255 are valid.
--   recentVehicleLocationN1: location delta with respect to
--                            vehicleLocation
--                            see VehicleLocationDelta
--   recentVehicleLocationN2: location delta with respect to
--                            recentVehicleLocationN1
--                            see VehicleLocationDelta
--   numberOfOccupants: Number of occupants in the vehicle according
--                      to available information.
--                      If no information about the number of
--                      occupants is available, this parameter needs
--                      to be omitted or set to 255.
--

MSDStructure  ::= SEQUENCE {
   messageIdentifier             INTEGER(0 .. 255),
   control                       ControlType,
   vehicleIdentificationNumber   VIN,
   vehiclePropulsionStorageType  VehiclePropulsionStorageType,
   timestamp                     INTEGER(0 .. 4294967295),
   vehicleLocation               VehicleLocation,
   vehicleDirection              INTEGER(0 .. 179 | 255),
   recentVehicleLocationN1       VehicleLocationDelta,
   recentVehicleLocationN2       VehicleLocationDelta,
   numberOfOccupants            INTEGER(0 .. 255) OPTIONAL,
  ...
}

-- The ControlType is a collection of the following elements:
-- Elements:
--   automaticActivation:  true = Automatic activation,
--                         false = Manual activation
--   testCall:             true = Test call, false = Emergency
--   positionCanBeTrusted: true = Position can be trusted,
--                         false = low confidence in position
--                         NOTE "Low confidence in position"
```

```
--                          shall mean that there is less than 95%
--                          confidence that exact position is
--                          within the limits of a radius of ±150m
--                          of reported position
--   vehicleType:           see VehicleType

ControlType ::= SEQUENCE {
  automaticActivation  BOOLEAN,
  testCall             BOOLEAN,
  positionCanBeTrusted BOOLEAN,
  vehicleType          VehicleType
}

-- Definition of the vehicle type reporting the incident.
-- NOTE: Vehicle definitions category M, N according directive 2007/46/EC;
--       category L according directive 2002/24/EC, other categories
--       according to UNECE UNECE TRANS/WP.29/78/Rev.4 (2016) World Forum for
--       Harmonization of Vehicle Regulations; Consolidated Resolution on the
--       Construction of Vehicles (R.E.3)
-- NOTE: The normal encoding of the MSD is ASN.1 UPER in which case the
--       numerical values specified below are ignored
--
-- Extendable in future versions for new vehicle types. Implementations
-- should anticipate on receiving an unknown value.

VehicleType ::= ENUMERATED{
   passengerVehicleCategoryM1 (1),
   busesAndCoachesCategoryM2 (2),
   busesAndCoachesCategoryM3 (3),
   lightCommercialVehiclesN1 (4),
   heavyDutyVehiclesCategoryN2 (5),
   heavyDutyVehiclesCategoryN3 (6),
   motorcyclesCategoryL1e (7),
   motorcyclesCategoryL2e (8),
   motorcyclesCategoryL3e (9),
   motorcyclesCategoryL4e (10),
   motorcyclesCategoryL5e (11),
   motorcyclesCategoryL6e (12),
   motorcyclesCategoryL7e (13),
   trailersCategoryO (14),
   agriVehiclesCategoryR (15),
   agriVehiclesCategoryS (16),
   agriVehiclesCategoryT (17),
   offRoadVehiclesCategoryG (18),
   specialPurposeMotorCaravanCategorySA (19),
   specialPurposeArmouredVehicleCategorySB (20),
   specialPurposeAmbulanceCategorySC (21),
   specialPurposeHearseCategorySD (22),
   otherVehicleCategory  (23),
   ...
}

-- VIN (vehicle identification number) according ISO 3779
--   isowmi: World Manufacturer Index (WMI)
--   isovds: Vehicle Type Descriptor (VDS)
--   Vehicle Identifier Section (VIS) consisting of
--     isovisModelyear: Modelyear from Vehicle Identifier Section (VIS)
--     isovisSeqPlant:  Plant code + sequential number
--                      from Vehicle Identifier Section (VIS)

VIN ::= SEQUENCE {
```

```
  isowmi          PrintableString (SIZE(3))
(FROM("A".."H"|"J".."N"|"P"|"R"..."Z"|"0".."9")),
  isovds          PrintableString (SIZE(6))
(FROM("A".."H"|"J".."N"|"P"|"R"..."Z"|"0".."9")),
  isovisModelyear PrintableString (SIZE(1))
(FROM("A".."H"|"J".."N"|"P"|"R"..."Z"|"0".."9")),
  isovisSeqPlant  PrintableString (SIZE(7))
(FROM("A".."H"|"J".."N"|"P"|"R"..."Z"|"0".."9"))
}

-- VehiclePropulsionStorageType is a collection of elements
-- that contain information about the presence of propulsion
-- storage inside the vehicle sending the MSD.
--
-- For each storage type the following coding applies:
--   false  = indicates a type of storage not present
--   true = indicates type of storage which is present
-- The following storage types are supported:
--   Gasoline tank
--   Diesel tank
--   Compressed natural gas (CNG)
--   Liquid propane gas (LPG)
--   Electric energy storage (vehicle has electric
--      propulsion and associated batteries)
--   Hydrogen storage
--   other storage
-- If the type of energy storage is unknown, then all elements
-- shall be set to false.
-- Extendible in future versions for new fuel storage types

VehiclePropulsionStorageType ::= SEQUENCE {
  gasolineTankPresent   BOOLEAN DEFAULT FALSE,
  dieselTankPresent     BOOLEAN DEFAULT FALSE,
  compressedNaturalGas  BOOLEAN DEFAULT FALSE,
  liquidPropaneGas      BOOLEAN DEFAULT FALSE,
  electricEnergyStorage BOOLEAN DEFAULT FALSE,
  hydrogenStorage       BOOLEAN DEFAULT FALSE,
  otherStorage          BOOLEAN DEFAULT FALSE,
 ...
}

-- VehicleLocation:
-- The current location of the vehicle
-- Elements:
--  Position latitude (WGS84) in milliarcsec
--  Position longitude (WGS84) in milliarcsec
--   32 bits (4 octets) allocated to make signed value handling easier
--
--  If latitude and/or longitude is invalid or unknown,
--  the representation of value 2147483647 shall be transmitted.

--
VehicleLocation  ::= SEQUENCE {
--   Actual valid value range (-324000000 to 324000000)
  positionLatitude  INTEGER(-2147483648..2147483647),
--   Actual valid value range (-648000000 to 648000000)
  positionLongitude INTEGER(-2147483648..2147483647)
}

-- VehicleLocationDelta:
-- Description of (the delta of) a recent vehicle location
-- before the incident
```

```
--  Latitude Delta (+ for North and - for South)
--  Longitude Delta (+ for East and - for West)
--  both with respect to a previous location.
--   1 Unit = 100 miliarcseconds, which is approximately 3m

VehicleLocationDelta ::= SEQUENCE {
  latitudeDelta  INTEGER (-512..511),
   longitudeDelta INTEGER (-512..511)
}

-- AdditionalData:
-- Further additional bytes of data encoded as in a
-- seperate ASN.1 definition
--
-- Elements:
--   oid: Object identifier which uniquely identifies the format
--        and meaning of the data which follows.
--   data:Transparent optional additional data,
--        according to the format referenced by the oid
--        The user must ensure that the size of this element
--        is restricted to ensure that the total ECallMessage is
--        small enough for the relevant transmission medium.

AdditionalData ::= SEQUENCE {
  oid  RELATIVE-OID,
  data OCTET STRING
}

-- Several of the elements above are "extendable"
-- according to the ASN.1 standard to facilitate future extensions
-- whilst also maintaining backwards compatibility. Implementations
-- must provision for such extensions


END
```

A.2 Syntax check of ASN.1 definition of MSD

```
ASN.1 Studio Version 9.0
Copyright (C) 2018 OSS Nokalva, Inc.  All rights reserved.
This product is licensed for use by
  "Cheiron IT bv, Leiden, Holland - Standards Editor", License "66410E",
  only for project "CEN TC278 WG15 related standards work"

ASN.1 syntax check result: C0043I: 0 error messages, 0 warning messages and 0 informatory
messages issued.

MSDv3: Compilation summary: The project MSDv3 includes 3 PDUs and 0 ASN.1 values.
```

A.3 Examples of ASN.1 encoded MSD

The example below is shown in ASN.1 value encoding (plain text):

```
value ECallMessage ::=
{
  msdVersion 3,
  msd
    CONTAINING
    {
      msdStructure
      {
        messageIdentifier 1,
        control
        {
          automaticActivation TRUE,
          testCall FALSE,
          positionCanBeTrusted TRUE,
          vehicleType passengerVehicleCategoryM1
        },
        vehicleIdentificationNumber
        {
          isowmi "ECA",
          isovds "LLEXAM",
          isovisModelyear "P",
          isovisSeqPlant "LE02020"
        },
        vehiclePropulsionStorageType
        {
          gasolineTankPresent TRUE,
          electricEnergyStorage TRUE
        },
        timestamp 1579992331,
        vehicleLocation
        {
          positionLatitude 187996428,
          positionLongitude 18859320
        },
        vehicleDirection 45,
        recentVehicleLocationN1
        {
          latitudeDelta 0,
          longitudeDelta 10
        },
        recentVehicleLocationN2
        {
          latitudeDelta 0,
          longitudeDelta 30
        },
        numberOfOccupants 2
      }
    }
}
```

The same example encoded in UPER (hexadecimal representation, 38 bytes):

```
0324101A 01C614A2 873C52AB A8700100 10089AF1 66285C59 A4C86408 FE29C16C 01054010 F010
```

Annex B
(informative)

ASN.1 Data representation PER and BER explained

B.1 What is ASN.1

NOTE This introduction was inspired by the information found in http://www.w3.org/Protocols/HTTP-NG/asn1.html. Definitions, examples and encoding traces were created using OSS Nokalva ASN.1 Studio.

ASN.1 is a notation for describing data structures; it is very much like a type declaration in C or C++. One of the key components of ASN.1 are Procotol Data Units (PDUs). Let's take a look at the ASN.1 definition for the MSD. It contains the PDU 'ECallMessage', which is defined like so:

```
ECallMessage ::= SEQUENCE {
      msdVersion INTEGER(0 .. 255),
      msd   MSDMessage
}
```

This defines EcallMessage as an object with two members:

— msdVersion, which content is an integer value between 0 and 255, and

— msd, which is a complex object of type MSDMessage, defined later on

Hence ASN.1 offers means of defining the type (i.e. INTEGER), the range (i.e. 0..255) and more complex constructions. Further down the definition one can find additional functionality that ASN.1 had to offer:

— optional element: some elements can be declared optional, and thus be left out of a data encoding (see MSDStructure.recentVehicleLocationN1);

— enumeration: an element can contain enumerated data, in such a way that the encoding is very efficient without losing content (see VehicleType);

— strict range checking: not only ranges like 0 .. 255 are possible, but combined ranges as well (see VIN)

The ASN.1 definition is used to encode and decode data content. Several encoding rules exist, among which:

— Basic Encoding Rules (BER): the original rules for taking an ASN.1 data type, and turning it into a sequence of bits and bytes;

— Packed Encoding Rules (PER): instead of using a generic style of encoding that encodes all types in a uniform way, the PER specialise the encoding based on the data type to generate much more compact representations;

— XML Encoding Rules (XER): an encoding using XML

B.2 Encoding data using ASN.1

B.2.1 General

The main purpose of any ASN.1 definition is to encode and decode data. This paragraph describes some of the basic procedures.

B.2.2 Basic Encoding Rules (BER)

The Basic Encoding Rules (BER) were the original rules for taking an ASN.1 data type, and turning it into a sequence of bits and bytes. BER uses a form of encoding commonly known as Tag-Length-Value. Each item is encoded as a tag, indicating what type it is, a length indicating the size of the object, and a value, which contains the actual contents of the object.

Tags are reasonably simple - in there simplest form they consist of a single byte; the highest two bits indicate the tag class:: whether the tag is an official ISO tag, an application wide tag, a private tag, or a tag that only has meaning for this particular structure. The next bit is a flag to indicate whether the tagged object is a simple type, such as a number or a string, or a compound type made up from a bunch of other types. The remaining 5 bits are used to represent the tag number. If the tag is too big to fit in 5 bits, then these bits are set to all ones, and the tag number is encoded in the following bytes as a sequence of seven bit bytes. The high bit of these bytes is used as a flag to indicate whether there's more tag available.

Lengths are also quite simple. There are two ways of encoding lengths - the definite form, and the indefinite form. For the definite form, if the length is less than 128, you just use a single byte, with the high bit set to zero. Otherwise the high bit is set to one, and the low seven bits set to the length of length. The length is then encoded in that many bytes. The indefinite form is encoded by sending a length field with a length of length of 0 - i.e. [1000|0000]. The object is ended by sending two zero bytes.

B.2.3 Distinguished Encoding Rules (DER)

DER is a restricted variant of BER for producing unequivocal transfer syntax for data structures described by ASN.1. DER is the same thing as BER with all but one sender's options removed. It provides for exactly one way to encode an ASN.1 value. DER is intended for situations when a unique encoding is needed, such as in cryptography, and ensures that a data structure that needs to be digitally signed produces a unique serialized representation.

DER can be considered a canonical form of BER. For example, in BER a Boolean value of true can be encoded as any of 255 non-zero byte values, while in DER there is only one way to encode a Boolean value of true.

The most significant DER encoding constraints are:

— Length encoding shall use the definite form. Additionally, the shortest possible length encoding shall be used.

— Bitstring, octetstring, and restricted character strings shall use the primitive encoding.

— Elements of a Set are encoded in sorted order, based on their tag value.

B.2.4 Packed Encoding Rules (PER/UPER)

The packed encoding rules use a different style of encoding. Instead of using a generic style of encoding that encodes all types in a uniform way, the PER specialise the encoding based on the data type to generate much more compact representations.

Packed encoding rules have particular value for messages of known data structure (such as the MSD)

PER only generates tags when they are needed to prevent ambiguity this only occurs when ASN.1's version of union (CHOICE) is used. PER also only generates lengths when the size of an object can vary. Even then, PER tries to represent the lengths in the most compact form possible.

The presence of optional elements in a sequence is indicated by a list of single bit flags placed at the start of a sequence - if the bit is set, then the option is present.

PER encodings are not always aligned on byte boundaries- if the 'aligned' variant of the rules is used, then strings *will* be aligned. The unaligned variant (UPER) treats the encoding as a string of bits, allowing things like booleans and small integers to be squished together in one byte.

B.2.5 XML Encoding Rules (XER)

Data can obviously also be encoded into an XML representation. Apart from XER, the 'basic XML encoding', one can choose to use the Canonical or the Extended XML encoding rules (CXER or EXER).

B.3 Examples

B.3.1 General

This paragraph shows in depth examples using ASN.1 encoding rules. For clarity a basic example (source: https://en.wikipedia.org/wiki/ASN.1) is used, examples of MSD encodings are found elsewhere.

B.3.2 ASN.1 example definition

The examples in this appendix will use the following definition:

```
FooProtocol

DEFINITIONS ::= BEGIN

       FooQuestion ::= SEQUENCE {
              trackingNumber      INTEGER,
              question     PrintableString
       }

       FooAnswer ::= SEQUENCE {
              questionNumber      INTEGER,
              answer       BOOLEAN
    }

END
```

This definition gives us two PDU's: FooQuestion and FooAnswer. We'll create an instance of both:

```
myQuestion FooQuestion ::= {
    trackingNumber  5,
    question "Anybody there?"
}

theirAnswer FooAnswer ::= {      questionNumber      5,     answer TRUE
}
```

In the remainder we will find out how these are encoded using the different encoding rulesets.

B.3.3 Encoding using BER or DER

myQuestion encoded using DER or BER (definite length) gives the following hexadecimal representation:

```
30 13 02 01 05 13 0E 41 6E 79 62 6F 64 79 20 74 68 65 72 65 3F
30 SEQUENCE
   13 of length .13 (19) bytes, containing
     02 INTEGER (type .02 (2))
        01 of length .01 (1) byte, containing
           05 value .05 (5)
  13 PrintableString (type .13 (19))
        0E of length .0E (14) bytes, containing
           41 character .41 ( 65, "A")
           6E character .6E (110, "n")
           79 character .79 (121, "y")
           62 character .62 ( 98, "b")
    …
           65 character .65 (101, "3")
           3F character .3F ( 63, "!")
```

Using BER with indefinite length, the coding is slightly different:

```
30 80 02 01 05 13 0E 41 6E 79 62 6F 64 79 20 74 68 65 72 65 3F 00 00
```

Where .80 represents the indefinite length, and the encoded message is terminated by '.00.00'.

B.3.4 Encoding using PER

If myQuestion is encoded using (aligned) PER the result differs 'drastically':

```
01 05 0E 41 6E 79 62 6F 64 79 20 74 68 65 72 65 3F
```

The difference is caused by not including information about the data structure in the encoded content. As a result the encoding is 4 bytes less than when using BER encoding. To decode, one needs the encoding definition, to be able to decode like this:

```
--TYPE INFORMATION: SEQUENCE
    --TYPE INFORMATION: INTEGER
01    First byte: length = .01, 1 byte
05    Value bytes (1 byte): .05 (5)
    --TYPE INFORMATION: PrintableString
0E    First byte: length = .0E, 14 bytes
…     Value bytes (14 bytes) =.41.6E.79.62.6F.64.79.20.74.68.65.72.65.3F
```

The information in the bold lines is taken from the encoding schema.

Using unaligned PER (UPER) the encoded data gets changed completely and the result is 1 byte less than the aligned version:

```
01 05 0E 83 BB CE 2D F9 3C A0 E9 A3 2F 2C AF C0
```

The difference is caused by the fact that a elements of a PrintableString are made up from 7 bit characters and these are now no longer aligned on an 8-bit boundary by prepadding the bits with a 0. This table shows the different encoding of the beginning of the string "Are ":

	.41	.6E	.79	.62	result	
PER encoding	01000001	01101110	01111001	01100010	01000001 01101110 01111001 01100010	
UPER encoding	1000001	1101110	1111001	1100010	1000001 *1101110* 1111001 1100010	83 *BB* CE

The number of bytes spared by using UPER is depending on how strict the schema is made and what data types are used.

B.3.5 Encoding using XER and EXER

When the XML Encoding Rules are used a rather simple XML representation can be created:

```
<?xml version="1.0" encoding="UTF-8"?>
<FooQuestion>
<trackingNumber>5</trackingNumber>
<question>Anybody there?</question>
</FooQuestion>
```

Annex C
(informative)

Formal XML format description (XSD) for the MSD

In respect of EN 16102 (EN 16102 Intelligent transport systems — eSafety — eCall: Operating requirements for third party support) the following elaboration of an xsd file is an alternative formal description for the ASN.1.

'Minimum Set of Data' defined in Annex A in order to provide messages in XML format.

It is equivalent to the ASN.1 description in Annex A, and a simple conversion between both formats is possible.

The XML format is less compact than the ASN.1 PER unaligned format, but may allow easier subsequent server-based processing.

As specified in 5.1.2 of this document, this XML data presentation should not be used for transmission of an eCall via a bandwidth-limited air interface standard as defined in EN 16072 [6] because in this case the ASN.1PER unaligned format should be used.

This XML format of the Minimum Set of Data is referred to in EN 16072 [6].

NOTE 1 This XML format might be used, for example, for subsequent processing of an MSD within the PSAP systems, or for forwarding an MSD from a PSAP to another relevant organization.

NOTE 2 This XSD was generated using the OSS ASN.1 to XSD translator. The translator was fed the normative MSD definition found in Annex A of this document. The XSD was generated using the 'exer' option (creating a stricter XSD) with 'wildcard' extensibility.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
        Generated by: OSS ASN.1 to XSD Translator Version 4.1
        Generated for: Cheiron IT bv, Leiden, Holland - Standards Editor, License 16073.
        Abstract syntax: 15722_MSD_v3
        Options and file names specified:
        -warningmessages -informatorymessages -exer -useExtensibility
        15722_MSD_v3.asn
  -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:oss="http://www.oss.com/XSD">

    <xsd:element name="CurrentVersion" type="CurrentVersion"/>
    <xsd:element name="ECallMessage" type="ECallMessage"/>
    <xsd:element name="MSDMessage" type="MSDMessage"/>

    <xsd:complexType name="MSDMessage">
        <xsd:sequence>
            <xsd:element name="msdStructure" type="MSDStructure"/>
            <xsd:element name="optionalAdditionalData" type="AdditionalData" minOccurs="0"/>
            <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="AdditionalData">
        <xsd:sequence>
            <xsd:element name="oid" type="RELATIVE_OID"/>
            <xsd:element name="data" type="OCTET_STRING"/>
        </xsd:sequence>
    </xsd:complexType>
```

```
<xsd:simpleType name="OCTET_STRING">
    <xsd:restriction base="xsd:string">
        <xsd:whiteSpace value="collapse"/>
        <xsd:pattern value="( *([0-9]|[A-F]|[a-f]) *)*"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="RELATIVE_OID">
    <xsd:restriction base="xsd:string">
        <xsd:pattern value="( *([a-z]|[0-9]|\{|\}|\.)* *)"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="MSDStructure">
    <xsd:sequence>
        <xsd:element name="messageIdentifier">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="0"/>
                    <xsd:maxInclusive value="255"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="control" type="ControlType"/>
        <xsd:element name="vehicleIdentificationNumber" type="VIN"/>
        <xsd:element name="vehiclePropulsionStorageType" type="VehiclePropulsionStorageType"/>
        <xsd:element name="timestamp">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="0"/>
                    <xsd:maxInclusive value="4294967295"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="vehicleLocation" type="VehicleLocation"/>
        <xsd:element name="vehicleDirection">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="0"/>
                    <xsd:maxInclusive value="255"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="recentVehicleLocationN1" type="VehicleLocationDelta"/>
        <xsd:element name="recentVehicleLocationN2" type="VehicleLocationDelta"/>
        <xsd:element name="numberOfOccupants" minOccurs="0">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="0"/>
                    <xsd:maxInclusive value="255"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="INTEGER">
    <xsd:restriction base="xsd:integer"/>
</xsd:simpleType>

<xsd:complexType name="VehicleLocationDelta">
    <xsd:sequence>
        <xsd:element name="latitudeDelta">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="-512"/>
                    <xsd:maxInclusive value="511"/>
```

```
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="longitudeDelta">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="-512"/>
                    <xsd:maxInclusive value="511"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="VehicleLocation">
    <xsd:sequence>
        <xsd:element name="positionLatitude">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="-2147483648"/>
                    <xsd:maxInclusive value="2147483647"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="positionLongitude">
            <xsd:simpleType>
                <xsd:restriction base="INTEGER">
                    <xsd:minInclusive value="-2147483648"/>
                    <xsd:maxInclusive value="2147483647"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="VehiclePropulsionStorageType">
    <xsd:sequence>
        <xsd:element name="gasolineTankPresent" type="BOOLEAN" minOccurs="0"/>
        <xsd:element name="dieselTankPresent" type="BOOLEAN" minOccurs="0"/>
        <xsd:element name="compressedNaturalGas" type="BOOLEAN" minOccurs="0"/>
        <xsd:element name="liquidPropaneGas" type="BOOLEAN" minOccurs="0"/>
        <xsd:element name="electricEnergyStorage" type="BOOLEAN" minOccurs="0"/>
        <xsd:element name="hydrogenStorage" type="BOOLEAN" minOccurs="0"/>
        <xsd:element name="otherStorage" type="BOOLEAN" minOccurs="0"/>
        <xsd:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="BOOLEAN">
    <xsd:choice>
        <xsd:element name="true" type="NULL"/>
        <xsd:element name="false" type="NULL"/>
    </xsd:choice>
</xsd:complexType>

<xsd:simpleType name="NULL">
    <xsd:restriction base="xsd:string">
        <xsd:length value="0"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="VIN">
    <xsd:sequence>
        <xsd:element name="isowmi">
            <xsd:simpleType>
                <xsd:restriction base="PrintableString">
                    <xsd:pattern value="[0-9A-HJ-NPR-Z]{3,3}"/>
                </xsd:restriction>
```

```
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="isovds">
            <xsd:simpleType>
                <xsd:restriction base="PrintableString">
                    <xsd:pattern value="[0-9A-HJ-NPR-Z]{6,6}"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="isovisModelyear">
            <xsd:simpleType>
                <xsd:restriction base="PrintableString">
                    <xsd:pattern value="[0-9A-HJ-NPR-Z]{1,1}"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="isovisSeqPlant">
            <xsd:simpleType>
                <xsd:restriction base="PrintableString">
                    <xsd:pattern value="[0-9A-HJ-NPR-Z]{7,7}"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="PrintableString">
    <xsd:restriction base="xsd:string">
        <xsd:pattern value="[ &apos;\(\)\+-:=\?A-Za-z]*"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="ControlType">
    <xsd:sequence>
        <xsd:element name="automaticActivation" type="BOOLEAN"/>
        <xsd:element name="testCall" type="BOOLEAN"/>
        <xsd:element name="positionCanBeTrusted" type="BOOLEAN"/>
        <xsd:element name="vehicleType" type="VehicleType"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="VehicleType">
    <xsd:choice>
        <xsd:element name="passengerVehicleCategoryM1" type="NULL"/>
        <xsd:element name="busesAndCoachesCategoryM2" type="NULL"/>
        <xsd:element name="busesAndCoachesCategoryM3" type="NULL"/>
        <xsd:element name="lightCommercialVehiclesN1" type="NULL"/>
        <xsd:element name="heavyDutyVehiclesCategoryN2" type="NULL"/>
        <xsd:element name="heavyDutyVehiclesCategoryN3" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL1e" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL2e" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL3e" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL4e" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL5e" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL6e" type="NULL"/>
        <xsd:element name="motorcyclesCategoryL7e" type="NULL"/>
        <xsd:element name="trailersCategoryO" type="NULL"/>
        <xsd:element name="agriVehiclesCategoryR" type="NULL"/>
        <xsd:element name="agriVehiclesCategoryS" type="NULL"/>
        <xsd:element name="agriVehiclesCategoryT" type="NULL"/>
        <xsd:element name="offRoadVehiclesCategoryG" type="NULL"/>
        <xsd:element name="specialPurposeMotorCaravanCategorySA" type="NULL"/>
        <xsd:element name="specialPurposeArmouredVehicleCategorySB" type="NULL"/>
        <xsd:element name="specialPurposeAmbulanceCategorySC" type="NULL"/>
        <xsd:element name="specialPurposeHearseCategorySD" type="NULL"/>
        <xsd:element name="otherVehicleCategory" type="NULL"/>
    </xsd:choice>
</xsd:complexType>
```

```
    <xsd:complexType name="ECallMessage">
        <xsd:sequence>
            <xsd:element name="msdVersion">
                <xsd:simpleType>
                    <xsd:restriction base="INTEGER">
                        <xsd:minInclusive value="0"/>
                        <xsd:maxInclusive value="255"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element name="msd">
                <xsd:complexType mixed="true">
                    <xsd:sequence>
                        <xsd:element name="MSDMessage" type="MSDMessage" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:simpleType name="CurrentVersion">
        <xsd:restriction base="INTEGER">
            <xsd:minInclusive value="3"/>
            <xsd:maxInclusive value="3"/>
        </xsd:restriction>
    </xsd:simpleType>

</xsd:schema>
```

Annex D
(informative)

Explanation of rationale for MSD data concept elements

This document is a part of global and consistent set of standards which will enable an end-to-end and an EU wide homogenous interoperability between a vehicle and a PSAP. A key requirement is a consistent set of data. The rationale for the inclusion of the specific data concepts in the MSD is as laid out in table D.1:

Table D.1 — Purpose of data concept elements

M – Mandatory data field
O – Optional data field

MSD		
msdVersion	M	Discriminate between MSD formats.
msd		
msdStructure		
messageIdentifier	M	The purpose of this data concept is so that if any of the information in the MSD has been updated, it can be discriminated from the original MSD.
control	M	Advise the emergency services of - whether the eCall has been manually or automatically generated - whether the eCall is a real emergency or known to be a test call. - whether the position information can be trusted - the type of vehicle This is considered useful information for the PSAP/emergency services, for example: - in case of an incident that generates eCalls from multiple vehicles, the automatic eCalls may be considered to be a higher priority. - if the position cannot be trusted the emergency services need to take extra steps to help confirm the location. - the type of vehicle gives extra information, relevant to the emergency services
vehicleIdentificationNumber	M	Enables the possibility to advise the emergency services of the make, model and colour of the affected vehicle. This is important for the emergency services to plan their actions to locate the correct vehicle and to distinguish between two separate calls from equipped vehicles.
vehiclePropulsion… …StorageType	M	Advise the emergency services of the type of vehicle energy storage(s) present. This is important particularly relating to fire risk and electrical power source issues.
timeStamp	M	Identify the accurate time of the incident. This is important in issues to do with how long the injured persons have been affected. It is important also for emergency services performance management.
vehicleLocation	M	Advise the emergency services of the exact location of the vehicle so that emergency services can locate it as quickly as possible.

vehicleDirection	M	Advise the emergency services of the direction of travel. This may be important e.g. where the incident is on a dual carriageway.
recentVehicleLocationN1	M	Further ensure that the direction of travel and accuracy of location are accurate. This is particularly in the case of complex junctions, multi-level roads and road over road bridges. This data is designed not to enable speed measurement.
recentVehicleLocationN2	M	
numberOfOccupants	O	Advise the emergency services of the likely minimum number of occupants of the affected vehicle. This may affect the form of their response.
optionalAdditionalData	O	Voluntary optional data fields are not required, but may be able to provide the emergency services with useful relevant additional information. The additional data field may include an address where other relevant related data or functions are available.

Annex E
(informative)

Object Identifiers (OID)

E.1 Formal definition of OID

The formal definition of OIDs can be found in ISO/IEC 8824 [ITU-T recommendation X.208] – Abstract Syntax Notation 1 (ASN.1).

The definition of OID is in chapter 3131; the idea of hierarchy is further explained in appendixes C and D.

The encodings - how you can transfer an OID as bits in a message - are defined in ISO/IEC 8825-1, ISO/IEC 8825-2; and/or ITU-T X.209, X.690, X.691.

E.2 What is an object identifier?

Object identifiers are, basically, strings of numbers. They are allocated in a hierarchical manner, so that, for instance, the authority for “1.” determines what 1.1, 1.2, 1.3, 1.nn means; the authority for “2.” determines what 2.1, 2.2, 2.3, 2.nn means. The authority that determines what "1.2.3" means is the only one that can say what "1.2.3.4" means, etc.

Object identifiers are used in a variety of protocols to provide a unique identification of a data concept/data element.

E.3 Object Identifiers and ISO standards

An object identifier is an ASN.1 data type that is used as a means of defining unique identifiers for objects. Values of the object identifier data type can then be used to name the objects to which they relate.

The object identifier data type consists of a sequence of one or more non-negative integers. They are often referred to as ‘arcs’, ‘arcs’ that define a hierarchy, or tree, of object identifier values.

The first arc in the sequence identifies the registration authority responsible for allocating the values of the second and subsequent arcs. For example:

iso(1) indicates that an initial arc value of 1 identifies ISO as the registration authority.

Subsequent arcs in the sequence are determined by ISO, or are allocated by registration authorities subordinate to ISO.

NOTE Under the iso arc, a second arc has been allocated to identify organizations recognized by ISO, e.g. CEN, IEEE etc. The two-integer sequence is not required in the case of an ISO Standard.

E.4 OID for eCall data concepts

The data concepts for eCall can be referenced and obtained from the data registry supporting eCall.

NOTE Current eCall/eSafety data registry source at date of submission of this document: www.eSafetyData.com). eCall Standards are developed by CEN TC278 WG15.

Bibliography

[1] EN ISO 24978, *Intelligent transport systems - ITS Safety and emergency messages using any available wireless media - Data registry procedures (ISO 24978)*

[2] ISO/IEC/ITU IS/Rec 8824-1/ X.680/8824-1:2002, Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation Published 2002. Cor 1 2006

[3] ISO/IEC/ITU IS/Rec 8824-2 /X.681:2002, Information technology — Abstract Syntax Notation One (ASN.1): Information object specification Published 2002. Amd1 2004

[4] ISO/IEC/ITU IS/Rec 8825-1/X.690:2002, Information technology —ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER). Published 2002. Amd 1 2004 JTC1/SC6

[5] ISO/IEC/ITU IS/Rec 8825-2/X.691:2002, Information technology — ASN.1 encoding rules: Specification of Packed Encoding Rules (PER) Published 2002.Amd 1 2004. Cor 1 2006. Amd 2 2007. Cor 2 2006

[6] EN 16072, *Intelligent transport systems - ESafety - Pan-European eCall operating requirements*

[7] ISO 3779, *Road vehicles - Vehicle identification number (VIN) - Content and structure*

[8] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)