INTERNATIONAL STANDARD

ISO 17572-2

Second edition
2015-01-15

Intelligent transport systems (ITS) — Location referencing for geographic databases —

Part 2: Pre-coded location references (pre-coded profile)

Systèmes intelligents de transport (SIT) — Localisation pour bases de données géographiques —

Partie 2: Localisations précodées (profil précodé)

Reference number
ISO 17572-2:2015(E)

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*.

This second edition cancels and replaces the first edition (ISO 17572-2:2008), which has been technically revised.

ISO 17572 consists of the following parts, under the general title *Intelligent transport systems (ITS) — Location referencing for geographic databases*:

— *Part 1: General requirements and conceptual model*

— *Part 2: Pre-coded location references (pre-coded profile)*

— *Part 3: Dynamic location references (dynamic profile)*

Introduction

A location reference (LR) is a unique identification of a geographic object. In a digital world, a real-world geographic object can be represented by a feature in a geographic database. An example of a commonly known location reference is a postal address of a house. Examples of object instances include a particular exit ramp on a particular motorway, a road junction, or a hotel. For efficiency reasons, location references are often coded. This is especially significant if the location reference is used to define the location for information about various objects between different systems. For intelligent transport systems (ITS), many different types of real-world objects will be addressed. Amongst these, location referencing of the road network, or components thereof, is a particular focus.

Communication of a location reference for specific geographic phenomena, corresponding to objects in geographic databases, in a standard, unambiguous manner is a vital part of an integrated ITS system in which different applications and sources of geographic data will be used. Location referencing methods (LRMs, methods of referencing object instances) differ by applications, by the data model used to create the database, or by the enforced object referencing imposed by the specific mapping system used to create and store the database. A standard location referencing method allows for a common and unambiguous identification of object instances representing the same geographic phenomena in different geographic databases produced by different vendors, for varied applications, and operating on multiple hardware/software platforms. If ITS applications using digital map databases are to become widespread, data reference across various applications and systems has to be possible. Information prepared on one system, such as traffic messages, has to be interpretable by all receiving systems. A standard method to refer to specific object instances is essential to achieving such objectives.

Japan, Korea, Australia, Canada, the US, and European ITS bodies are all supporting activities of location referencing. Japan has developed a Link Specification for VICS. Japan has developed the Road Section Identification Data set which uses road sections and reference points after the publication of the first edition of this part of ISO 17572 in 2008. In Europe, the RDS-TMC traffic messaging system has been developed. In addition, methods have been developed and refined in the EVIDENCE and AGORA projects based on intersections identified by geographic coordinates and other intersection descriptors. In the US, standards for location referencing have been developed to accommodate several different location referencing methods.

This International Standard provides specifications for location referencing for ITS systems (although other committees or standardization bodies can subsequently consider extending it to a more generic context). In addition, this edition does not deal with public transport location referencing; this issue will be dealt with in a later edition.

Intelligent transport systems (ITS) — Location referencing for geographic databases —

Part 2: Pre-coded location references (pre-coded profile)

1 Scope

This International Standard specifies location referencing methods (LRMs) that describe locations in the context of geographic databases and will be used to locate transport-related phenomena in an encoder system as well as in the decoder side. This International Standard defines what is meant by such objects and describes the reference in detail, including whether or not components of the reference are mandatory or optional, and their characteristics.

This International Standard specifies two different LRMs:

— pre-coded location references (pre-coded profile);

— dynamic location references (dynamic profile).

This International Standard does not define a physical format for implementing the LRM. However, the requirements for physical formats are defined.

This part of ISO 17572 specifies the pre-coded location referencing method, comprising

— specification of pre-coded location references (pre-coded profile),

— logical format for VICS link location (see Annex A),

— TPEG physical format for ALERT-C-location references (see Annex B, Annex C, and Annex D),

— TPEG physical format for Korean node-link ID references (see Annex E, Annex F, and Annex G), and

— logical format for Road Section Identification Data set (see Annex H).

Logical formats of the implementations at present are described as informative; on the other hand, physical formats of the implementations at present are described as normative in the annexes.

It is consistent with other International Standards developed by ISO/TC 204 such as ISO 14825.

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 17572-1, *Intelligent transport systems (ITS) — Location referencing for geographic databases — Part 1: General requirements and conceptual model*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 17572-1 and the following apply.

3.1
major link
directed link in a road network

4 Abbreviated terms

ALERT	Advice and Problem Location for European Road Traffic
DATEX	Data Exchange (protocol for exchange of traffic and travel information between traffic centres)
GCId	generic component identifier
GDF	Geographic Data File
ID	identifier
ITRF	International Terrestrial Reference Frame
LDB	location database
LI	location information
LR	location referencing (or reference)
LRM	location referencing method
LRS	location referencing system
LRP	location referencing procedure
MOCT	Ministry of Construction and Transportation (Republic of Korea)
RDS	Radio Data System
SOEI	System Operating and Exchanging Information
TMC	Traffic Message Channel
TPEG	Transport Protocol Expert Group
TTI	Traffic and Traveller Information
UTM	Universal Transverse Mercator
VICS	Vehicle Information and Communication System
RSIDs	Road Section Identification Data set

5 Requirements for a location referencing standard

For details, see ISO 17572-1:2014, Clause 4.

For an inventory of location referencing methods, see ISO 17572-1:2014, Annex A.

6 Conceptual data model for location referencing methods

For details, see ISO 17572-1:2014, Clause 5.

For examples of conceptual data model use, see ISO 17572-1:2014, Annex B.

7 Specification of pre-coded location references

7.1 General concept

Pre-coded location referencing is a method which makes use of end-user client devices carrying a location database (LDB) that is exactly the same as the corresponding location database used by a service provider of a particular message being exchanged. All pre-coded location referencing methods shall share the concept of defining a commonly used database of IDs. This concept has been developed in the past for technologies such as RDS-TMC and VICS to allow an (over-the-air) interface to be designed that uses compact code values (IDs) in the corresponding databases to express particular pre-coded locations of various types.

The location referencing method here is divided into three steps performed to implement the location referencing system. The first step is a process of defining the database of location IDs for a given area and the corresponding road network. In this step, different service providers and systems provider agree on a defined database containing all locations to be codable (location database creation). In the second step, this database is provisioned *via* various means into the service providers database as well as into all receiving systems (location database provisioning). The third step is in real-time where a service provider can now make use of that database and reference to locations by using the newly introduced IDs (location database usage). See Figure 1, which illustrates this concept.

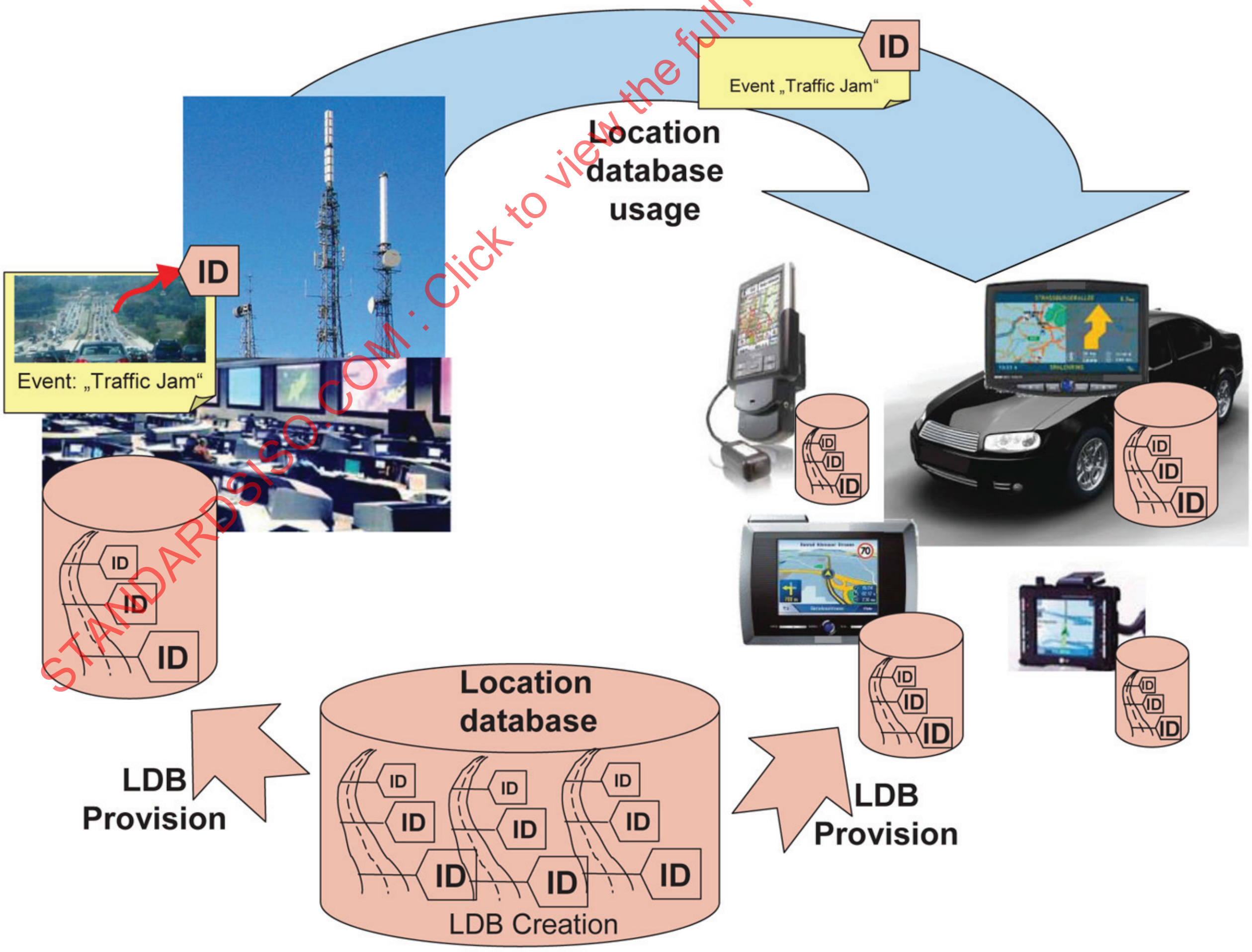

Figure 1 — General concept of pre-coded location referencing system

7.2 Location database creation and updating

The different location referencing systems more or less support standardized ways to create a new release of a location database. All of them share a conceptual model specifying how the different location categories specified in ISO 17572-1 are related to each other. This specification, together with some guiding literature, helps the community to create new releases of the location database.

7.3 Location database provision

After finalization of the creation process, the newly created location database is provisioned into the devices with maintenance service agreements. This is mostly done on a regular based map release update. The location referencing system has to ensure that the encoding and the decoding entities are able to distinguish which release (version) of the database is in use because no conclusion regarding the correctness of the location can be made based on the contents of the IDs alone.

7.4 Location database usage

A service provider, using the current release data set, then creates messages with location references according to specified rules a location reference out of the list of location IDs available and can put additional attributes to it, to define more precisely which part of the road network is referred to. The location reference sent to the receiving system then consists of a list of one or more location IDs and some additional attributes. Presuming that the receiving system has the actual database available, it seeks for the given location IDs and applies the additional attributes according to the location referencing specification. Doing so, the decoder provides the same location definition as requested by the service provider.

8 Implementations at present

8.1 General

Different implementations of pre-coded location referencing have already been specified for some time. Some of them are captured in another International Standard and some of them need some more specification here. This clause provides a list of presently known pre-coded location referencing methods and introduces them shortly. It also refers the reader to the different documents needed to fully apply to the different implementations.

8.2 Vehicle Information and Communication System (VICS)

8.2.1 Location database creation

Vehicle Information and Communication System specifies in Reference [2] a digital map database as the basis for other map provider to adopt the different map IDs into their own digital map. The digital base map consists of nodes and road elements which build up a complete street map on level zero. See Figure 2, which defines the conceptual data model for this map.

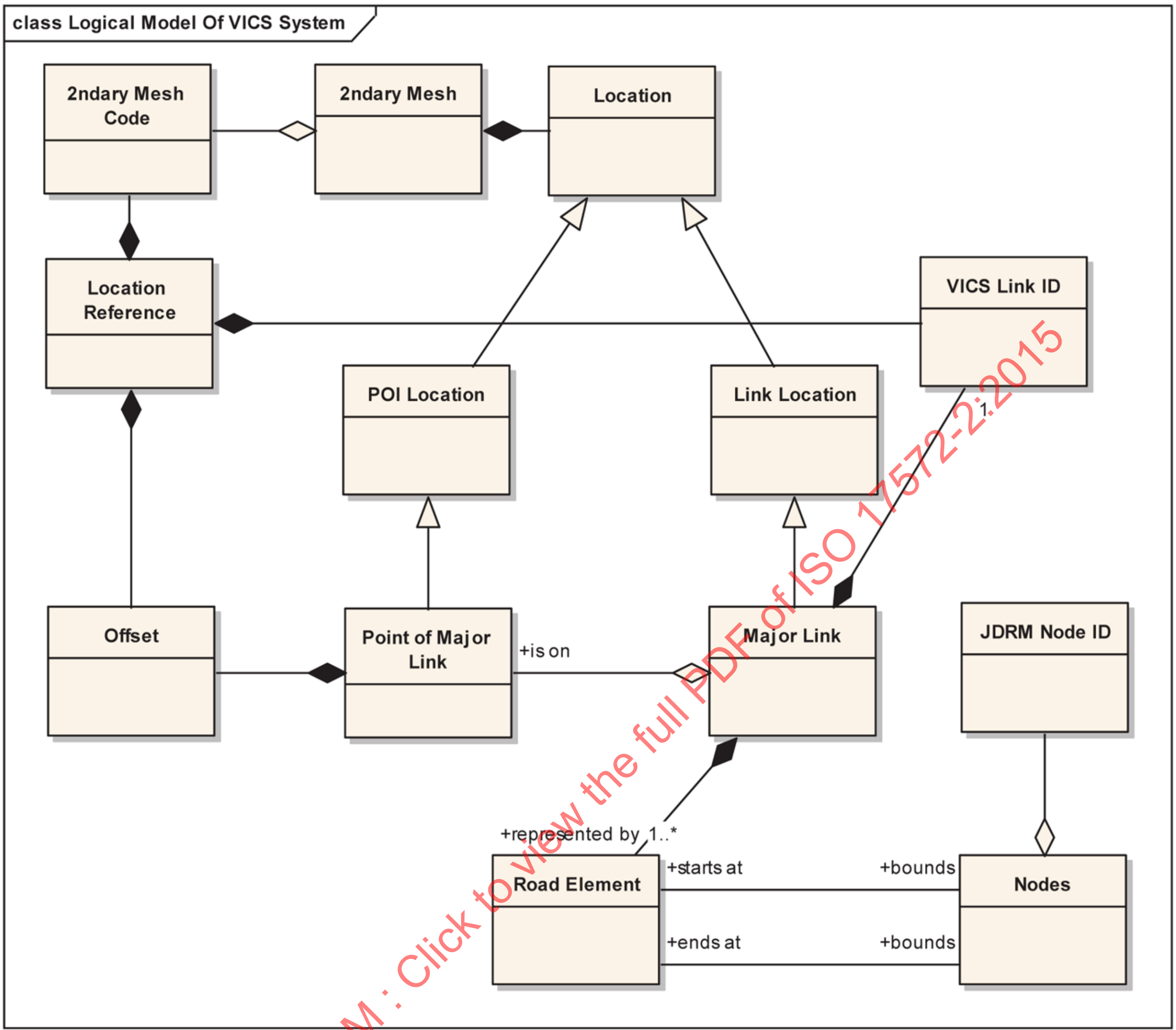

Figure 2 — Logical data model of VICS digital base map

8.2.2 Location database usage

All or any part of the specified digital map database can be referred to by a location reference consisting of VICS-Link-IDs, 2ndary-Mesh-Codes, and offsets. The specification in Reference [1] defines how the digitized location IDs have to be coded to build up a more sophisticated location.

8.3 Traffic Message Channel (TMC)/ALERT-C Specification

8.3.1 General

The location referencing rules defined in ISO 14819-3[11] address the specific requirements of Traffic Message Channel (TMC) systems, which use abbreviated coding formats to provide TTI messages over mobile bearers (e.g. GSM, DAB) or *via* exchange protocols like DATEXII. In particular, the rules address the Radio Data System Traffic Message Channel (RDS-TMC), a means of providing digitally-coded traffic and travel information to travellers using a silent data channel (RDS) on FM radio stations, based on the ALERT-C protocol.[10]

8.3.2 Location database creation

Location types and subtypes are required for language independence of the information given, and to tell the receiving system what data fields to expect.

At the highest level, locations fall into three categories:

a) area locations;

b) linear locations;

c) point locations.

RDS-TMC location tables use a hierarchical structure of pre-defined locations. Locations are identified using a location ID. A system of pointers provides upward references to higher-level locations of which the specified location forms a part. As such, all point locations belong to linear locations and they refer to area locations. Point locations additionally refer to a succeeding and a preceding point location which builds up a connected network of point locations. Further information can be found in a coding handbook that has been written by the TMC forum.[4]

8.3.3 Location database usage

A location ID in such a message refers and serves as a tabular "address" of the pre-stored location details in the location database used by the service. A real world location can have more than one point location within the same location table, which can be expressed by one point location code and an additional attribute extent which counts the steps of succeeding point location to be added to the location. Another additional attribute direction allows to extend from a point location into positive or into negative direction according to the point location direction defined in the location database.

8.4 Korean node link ID system

8.4.1 General

The Ministry of Construction and Transportation (MOCT) of Korea has developed a standard node-link system for ITS in 2004 for effective exchange of real-time traffic information. The node and link ID is made up of 10 digits. Korean standard node-link ID is the standard location ID for TPEG-Loc services in Korea.[3]

8.4.2 Location database creation

In principle, road authorities create and manage standard node-link IDs and digital base map for those standard node/link according to Reference [6], which was published by MOCT. MOCT verifies the IDs and digital base map, then officially distributes them.

8.4.3 Location database usage

Any node or link ID can be served as location ID in location referencing system, but currently, only link ID is used in currently implemented systems.

8.5 Road Section Identification Data set

8.5.1 General

The Road Section Identification Data set was developed to enable exchanging various static/dynamic information on road network.

The location of the information is represented by an appropriate road section with a reference point and a distance from the reference point.

In this part of ISO 17572, a profile of road section identification and reference point identification required in the Road Section Identification Data set are provided. Definition of each link and node corresponding to respective road maps is out of scope.

As permanent ID set is specified in the system, Road Section Identification Data set is independent from avoidable change caused by road map revision.

8.5.2 Location database creation

Road Section Identification Data set creates an authority table for section IDs and reference point IDs.

8.5.3 Location database usage

Road Section Identification Data set is expected to use for exchanging various location information related road between different players. For example, road authorities use this method to provide road information to private sectors.

Annex A
(informative)

Logical format for VICS link location

A.1 Description of the logical structure

A.1.1 General

The subsequent clauses define data elements used for building up the VICS link location reference (database usage). Different descriptions of the data structure help to understand the concept. It consists of a location information (LI) header and location content as shown in Figure A.1, with the latter further subdivided functionally into coordinates, descriptors, and offset information. Figure A.2 describes the structure of the LI main in the form of a UML diagram. A.3 and A.4 do define different views on a logical format.

All or any part of the LI can be omitted optionally if it is possible to refer to a location between databases without all or any part of LI content by defining unambiguous rules for a physical format and by establishing a management system.

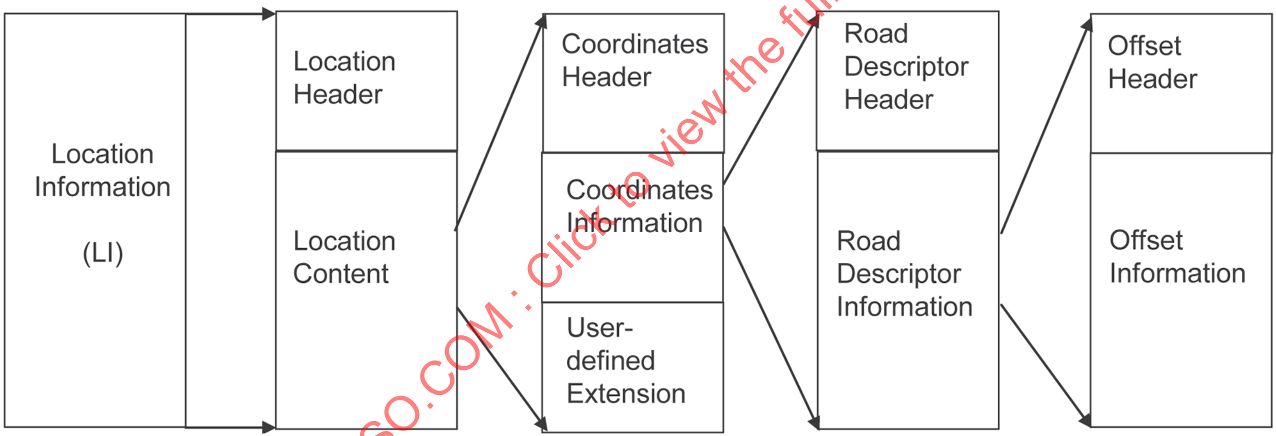

Figure A.1 — Outline diagram of the logical structure

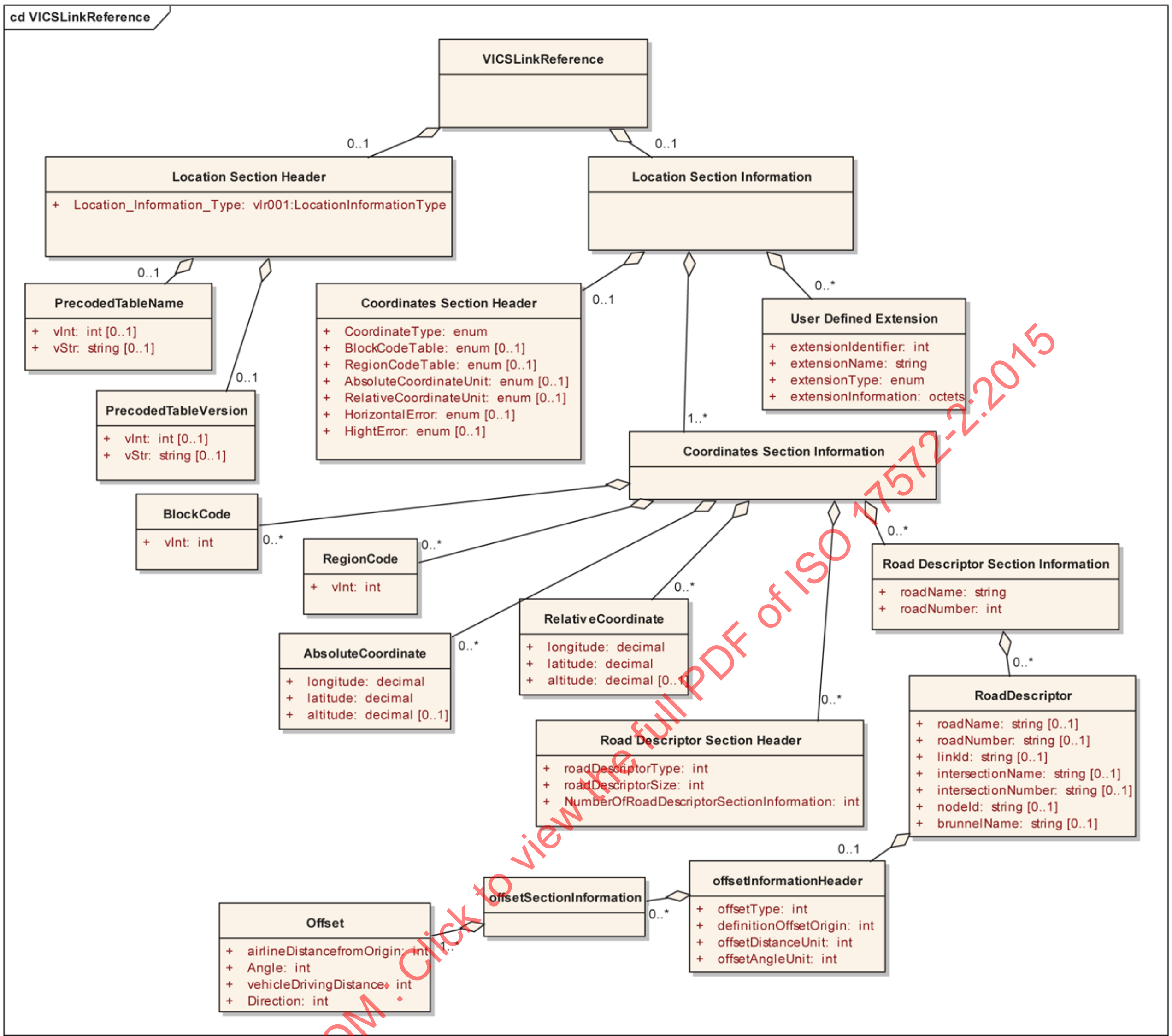

Figure A.2 — Outline diagram of the logical structure in UML

A.1.2 Data values

Table A.1 lists all specific values of enumerations used in the location reference format in this annex.

Table A.1 — Enumerations used in the location reference format in this annex

Data value name	Definition
basemap1	A parameter specifying that the location is digitized on a map of 1/2 500~1/10 000 scale
basemap2	A parameter specifying that the location is digitized on a map of 1/25 000~1/50 000 scale
basemap3	A parameter specifying that the location is digitized on a map of more than 1/100 000 scale
ddmmss	A parameter specifying a coordinate is expressed using decimal integer value of degree, minute, and second
degree	A parameter specifying that a unit of coordinates is degree
error1	A parameter specifying that a height error is less than 1 m

Table A.1 *(continued)*

Data value name	Definition
error2	A parameter specifying that a height error is less than 10 m
extensiontype1|2|...|n	The type of user-defined extension of which *n* different enumerated values are specified in the User-Defined Extension Header data frame
absolute	A parameter specifying that a coordinate system is absolute
relative	A parameter specifying that a coordinate system is relative
grid	A parameter specifying that the coordinate has a grid code
Relative_X	The horizontal X value of relative coordinates of a point
Relative_Y	The horizontal Y value of relative coordinates of a point
Relative_Z	The value of height in relative coordinates of a point
pgrid1	A parameter specifying that a coordinate has a private grid
rct1	A parameter specifying that region code table 1 is used
rct2	A parameter specifying that region code table 2 is used
rectc	A parameter specifying that a coordinate system is rectangular
second	A parameter specifying that a unit of absolute coordinates is second
secondu1	A parameter specifying that a unit of relative coordinates is normalized
secondu2	A parameter specifying that a unit of relative coordinates is some value
utmp	A parameter specifying that the blocking is the UTM primary mesh dividing
utms	A parameter specifying that the blocking is the UTM secondary mesh dividing
x	The value of horizontal axis on relative coordinates
y	The value of vertical axis on relative coordinates
z	The value of height in relative coordinates

A.1.3 Data elements

In Table A.2, the maximum value (labelled with MAX) in the column "Valid value rule", is specified first when defining unambiguous rules for a physical format by establishing the system implementation.

A.1.4 Data frames

See Table A.3.

Table A.2 — Data elements used in the location reference format in this annex

Data element name	Definition	Data type	Valid value rule
Absolute_Coordinate_Unit	A unit for an absolute coordinate	BIT STRING	ENUMERATED { degree, ddmmss, second}
Airline_Distance_from_Origin	The shortest distance from an origin to a point	Distance_type	CHOICE {INTEGER, REAL}
Altitude	The geographic altitude of a node	Altitude_type	CHOICE {INTEGER, REAL}
Angle	An integer angle from a starting point to a feature, in units defined by Offset_Angle_Unit	Angle_type	CHOICE {INTEGER, REAL}
Block_Code	A code given to an area such as one of UTM's meshes or some rectangles	INTEGER	SIZE (0..MAX)
Block_Code_Table	A table of block codes	BIT STRING	ENUMERATED { utms, utmp, rectc, pgrid1}
Brunnel_Name	Text name of a Brunnel	UTF8String	SIZE (0..255)
Coordinate_Type	A type of the coordinate such as the absolute, the relative, and the composite	BIT STRING	ENUMERATED { absolute, relative, grid}
Definition_Offset_Origin	A parameter specifying whether an origin of offset is starting node or end node	BIT STRING	ENUMERATED { Start point, End point}
Direction	A parameter specifying the direction from a starting point to a feature, in units defined by Offset_Distance_Unit either as integer value or as real value	Direction_type	CHOICE {INTEGER, REAL}

Table A.2 *(continued)*

Data element name	Definition	Data type	Valid value rule
Extension_Identifier	Identifier of a user-defined data element	Extension_Id_type	CHOICE {INTEGER, UTF8String}
Extension_Name	Name of a user-defined data element	UTF8String	SIZE (0..255)
Extension_Type	Type of a user-defined data element	BIT STRING	ENUMERATED { extensiontype1, extensiontype2, ... extensiontypen}
Height_Error	The height (vertical) error specifying an altitude accuracy of coordinates	BIT STRING	ENUMERATED { error1, error2}
Horizontal_Error	The horizontal error specifying a horizontal accuracy of coordinates	BIT STRING	ENUMERATED { basemap1, basemap2, basemap3}
Intersection_Name	Text name of an intersection	UTF8String	SIZE (0..255)
Intersection_Number	Integer number of an intersection	INTEGER	SIZE (0..MAX)
Latitude	ITRF geographic latitude of a node	Latitude_type	CHOICE {INTEGER, REAL}
Link_ID	Link identifier	INTEGER	SIZE (0..MAX)
Location_Information_Type	A parameter specifying whether the location information used for the location referencing is information of a point, lines, or area	BIT STRING	ENUMERATED{ point, line, face}
Longitude	ITRF geographic longitude of a node	Longitude_type	CHOICE {INTEGER, REAL}
Node_ID	Node identifier	INTEGER	SIZE (0..MAX)

Table A.2 *(continued)*

Data element name	Definition	Data type	Valid value rule
Number_of_Absolute_Coordinates	The number of absolute coordinates in a Coordinate_Information data frame	INTEGER	SIZE (0..MAX)
Number_of_Coordinates_Information	The number of Coordinate_Information data frames	INTEGER	SIZE (0..MAX)
Number_of_Blocks	The number of blocks in a Coordinate_Information data frame	INTEGER	SIZE (0..MAX)
Number_of_Offsets	The number of offsets in an Offset_Information data frame	INTEGER	SIZE (0..MAX)
Number_of_Offset_Information	The number of Offset_Information data frames	INTEGER	SIZE (0..MAX)
Number_of_Regions	The number of regions in a Coordinate_Information data frame	INTEGER	SIZE (0..MAX)
Number_of_Relative_Coordinates	The number of relative coordinates in a Coordinate_Information data frame	INTEGER	SIZE (0..MAX)
Number_of_Road_Descriptors	The number of Road_Descriptors in a Road_Descriptor_Information data frame	INTEGER	SIZE (0..MAX)
Number_of_Road_Descriptor_Information	The number of Road_Descriptor_Information data frames	INTEGER	SIZE (0..MAX)
Offset_Angle_Unit	The unit of an angle defining the direction from a starting point to the feature, such as a degree or a radian	BIT STRING	ENUMERATED{ degree, radian}
Offset_Distance_Unit	The unit defining a distance from a starting point to the feature, such as a metre or 10 m	BIT STRING	ENUMERATED{ 1m, 10m, 100m}
Offset_Type	A type of the offset, on-road, or airline	BIT STRING	ENUMERATED{ on-route, airline}
Precoded_Table_Name	The text identifier of a pre-coded table containing location information	UTF8String	SIZE (0..255)
Precoded_Table_Version	Version of the pre-coded table	Precoded_Table_Version_type	CHOICE {INTEGER, UTF8String}
Region_Code	A code of a region such as a country, city, or state	INTEGER	SIZE (0..MAX)

Table A.2 *(continued)*

Data element name	Definition	Data type	Valid value rule
Region_Code_Table	A table of Region_Codes	BIT STRING	ENUMERATED { rct1, rct2}
Relative_Coordinate_Unit	A unit for a relative coordinate	BIT STRING	ENUMERATED { secondu1, secondu2}
Relative_X	The value of horizontal axis on relative coordinates	INTEGER	SIZE (0..MAX)
Relative_Y	The value of vertical axis on relative coordinates	INTEGER	SIZE (0..MAX)
Relative_Z	The value of height in relative coordinates	INTEGER	SIZE (0..MAX)
Road_Descriptor_Size	A number of characters of a road descriptor	INTEGER	SIZE (0..MAX)
Road_Descriptor_Type	Classification of a road descriptor such as express way, national road and others	INTEGER	SIZE (0..MAX)
Road_Name	Text name of a road	UTF8String	SIZE (0..255)
Road_Number	Integer number of a road	INTEGER	SIZE (0..MAX)
Vehicle_Driving_Distance	The distance a vehicle drives to an offset point	Distance_type	CHOICE {INTEGER, REAL}

Table A.3 — Data frame name used in the location reference format in this annex

Data frame name	Definition	Data elements/Nested frames
Location_Information	A data frame containing a Location_Section_Header and Location_Section_Information	Location_Section_Header
		Location_Section_Information
Location_Section_Header	A data frame containing header information for location information	Precoded_Table_Name
		Precoded_Table_Version
		Location_Information_Type
Location_Section_Information	A data frame containing coordinate, road descriptor, and offset information and headers	Coordinates_Section_Header
		Coordinates_Section_Information[a]
		User_Defined_Extension
Coordinates_Section_Header	A data frame containing header information for coordinate information	Coordinate_Type
		Block_Code_Table
		Region_Code_Table
		Absolute_Coordinate_Unit
		Relative_Coordinate_Unit
		Horizontal_Error
		Height_Error
		Number_of_Coordinates_Section_Information
Coordinates_Section_Information[a]	A data frame containing block, region, absolute, and relative coordinate information	Number_of_Blocks
		Number_of_Regions
		Number_of_Absolute_Coordinates
		Number_of_Relative_Coordinates
		Road_Descriptor_Section_Header
		Road_Descriptor_Section_Information[a]
Blocks[a]	A data frame containing Block_Codes	Block_Code
Regions[a]	A data frame containing Region_Codes	Region_Code

[a] Indicates that a data frame is repeatable; data frame “X” occurs “Number_of_X” times.

Table A.3 *(continued)*

Data frame name	Definition	Data elements/Nested frames
Absolute_Coordinates[a]	A data frame of ITRF geographic coordinates	Latitude
		Longitude
		Altitude
Relative_Coordinates[a]	A data frame containing relative x, y, and z coordinate information	Relative_X
		Relative_Y
		Relative_Z
Road_Descriptor_Section_Header	A data frame containing road descriptors header information	Road_Descriptor_Type
		Road_Descriptor_Size
		Number_of_Road_Descriptor Section_Information
Road_Descriptor_Section_Information[a]	A data frame containing road descriptor section information	Number_of_Road_Descriptors
		Road_Descriptor[a]
Road_Descriptor[a]	A data frame containing road descriptors	Road_Name
		Road_Number
		Link_ID
		Intersection_Name
		Intersection_Number
		Node_ID
		Brunnel_Name
		Offset_Section_Header
		Offset_Section_Information[a]
Offset_Section_Header	A data frame containing header information for offsets	Offset_Type
		Definition_Offset_Origin
		Offset_Distance_Unit
		Offset_Angle_Unit
		Number_of_Offset_Section_Information

[a] Indicates that a data frame is repeatable; data frame “X” occurs “Number_of_X” times.

Table A.3 *(continued)*

Data frame name	Definition	Data elements/Nested frames
Offset_Section_Information[a]	A data frame containing offsets	Number_of_Offsets
		Offsets[a]
Offsets[a]	A data frame containing offset information on a distance from a base point to the feature	Airline_Distance_from_Origin
		Angle
		Vehicle_Driving_Distance
		Direction
User_Defined_Extension	A data frame containing user defined extension information	User_Defined_Extension_Header
		User_Defined_Extension_Information
User_Defined_Extension_Header	A data frame containing header information for user defined information	Extension_Identifier
		Extension_Name
		Extension_Type
User_Defined_Extension_Information	A data frame containing data elements and/or nested data frames are user-defined and not specified	<user defined>

[a] Indicates that a data frame is repeatable; data frame "X" occurs "Number_of_X" times.

A.2 Detailed diagram of logical structure

- **Lrp1-linf** : *Location information of Location referencing procedure 1*
  - **lsheader** : *Location Section Header*
    - **ptname** : *Precoded Table Name*
    - **ptversion** : *Precoded Table Version*
    - **linft** : *Location Information Type*
  - **lsinf** : *Location Section Information*
    - **cshi** : *Coordinates Section Header*
      - **ct** : *Coordinate Type*
      - **bct** : *Block Code Table*
      - **rct** : *Region Code Table*
      - **acu** : *Absolute Coordinate Unit*
      - **rcu** : *Relative Coordinate Unit*
      - **horerr** : *Horizontal Error*
      - **hterr** : *Height Error*
      - **numcsi** : *Number of Coordinates Section Information*
    - **csi** : *Coordinates Section Information (numcsi occurrences)*
      - **numblocks** : *Number of Blocks*
      - **bcodes** : *Block Code (numblocks occurrences)*
      - **numregion** : *Number of Regions*
      - **rcodes** : *Region Code (numregions occurrences)*
      - **numac** : *Number of Absolute Coordinates*
      - **ac** : *Absolute Coordinate (numac occurrences)*
        - **lat** : *Latitude*
        - **lon** : *Longitude*
        - **alt** : *Altitude*
      - **numrc** : *Number of Relative Coordinates*
      - **rc** : *Relative Coordinate (numrc occurrences)*
        - **x** : *Relative X*
        - **y** : *Relative Y*
        - **z** : *Relative Z*
      - **rdsh** : *Road Descriptor Section Header*
        - **rdt** : *Road Descriptor Type*
        - **rdsize** : *Road Descriptor Size*
        - **numrdsinf** : *Number of Road Descriptor Section Information*
      - **rdsinf** : *Road Descriptor Section Information (numrdsinf occurrences)*
        - **numrd** : *Number of Road Descriptor*
        - **rds** : *Road Descriptor (numrd occurrences)*
          - **rname** : *Road Name*
          - **rnum** : *Road Number*
          - **linkid** : *Link ID*
          - **intersectname** : *Intersection Name*
          - **intersectid** : *Intersection Number*
          - **nodeid** : *Node ID*
          - **brunnelname** : *brunnel Name*
          - **offsecthdr** : *offset Information Header*
            - **offsett** : *Offset Type*
            - **defoffsetorg** : *Definition Offset Origin*
            - **offsetdu** : *Offset Distance Unit*
            - **offsetau** : *Offset Angle Unit*
            - **numoffinf** : *Number of Offset Section Information*
          - **offsectinf** : *Offset Section Information (numoffinf occurrences)*
            - **noffset** : *Number of Offsets*
            - **offsets** : *Offsets (noffset occurrences)*
              - **airlinedfo** : *Airline Distance fromOrigin*
              - **angle** : *Angle*
              - **vehicledrivingd** : *Vehicle Driving Distance*
              - **direction** : *Direction*
    - **UDExt** : *User-defined Extension*
      - **UDExtHeader** : *User-defined ExtensionHeader*
        - **extID** : *Extension Identifier*
        - **extName** : *Extension Name*
        - **extType** : *Extension Type*
      - **UDExtinf** : *User-defined ExtensionInformation*
        - **< user-defined data elements and/or nested data frames >**

A.3 Structure in ASN.1

```
Lrp1-linf DEFINITIONS::=            --Lrp1-linf: Location information of Location
referencing procedure 1
BEGIN
EXPORTS
   Lrp1-linf;
IMPORTS
Precoded-Table, Blockcode-Table;

   Lrp1-linf::= SEQUENCE {
      lsheader      Lsheader       OPTIONAL,                -Location_Section_Header
      lsinf         Lsinf                                   -Location_Section_Information
      }

   Lisheader::= SEQUENCE {
      ptname        UTF8String     OPTIONAL,                -Precoded_Table_Name
      ptversion     Ptversion      OPTIONAL,                -Precoded_Table_Version
      linft         Linft          OPTIONAL                 -Location_Information_Type
      }

   Ptversion::= CHOICE {
      vint          INTEGER,
      vstr          UTF8String
      }

   Linft::= ENUMERATED {
      point,
      line,
      area
      }

   Lsinf::= SEQUENCE {
      cshi          Csih           OPTIONAL,                -Coordinates_Section_Header
      csi           Csi            OPTIONAL,                -Coordinates_Section_Informa-
tion
      udext         UDExt          OPTIONAL                 -User-Defined Extension
      }

   Csih::= SEQUENCE {
      ct            Ct             OPTIONAL,                -Coordinate_Type
      bct           Bct            OPTIONAL,                -Block_Code_Table
      rct           Rct            OPTIONAL,                -Region_Code_Table
      acu           Acu            OPTIONAL,                -Absolute_Coordinate_Unit
      rcu           Rcu            OPTIONAL,                -Relative_Coordinate_Unit
      horerr        Horerr         OPTIONAL,                -Horizontal_Error
      hterr         Hterr          OPTIONAL,                -Height_Error
      numcsi        INTEGER        OPTIONAL                 -Number_of_Coordinates_
                                                            -Information
      }

   Ct::= ENUMERATED {
      absolute,
      relative,
      grid
      }

   Bct::= ENUMERATED {
      utms,
      utmp,
      rectc,
      pgrid1
      }

   Rct::= ENUMERATED {
      rct1,
      rct2,
      }
```

```
Acu::= ENUMERATED {
   degree,
   ddmmss,
   second
   }

Rcu::= ENUMERATED {
   secondu1,
   secondu2
   }

Horerr::= ENUMERATED {
   basemap1,
   basemap2,
   basemap3
   }

Hterr::= ENUMERATED {
   error1,
   error2
   }

Csi::= SEQUENCE {
   numblock      INTEGER         OPTIONAL,                -Number_of_Blocks
   bcodes        SEQUENCE (SIZE (0..MAX)) OF INTEGER,     -Block_Codes
   numregion     INTEGER         OPTIONAL,                -Number_of_Regions
   rcodes        SEQUENCE (SIZE (0..MAX)) OF INTEGER,     -Region_Codes
   numac         INTEGER         OPTIONAL,                -Number_of_Absolute_
                                                           Coordinates
   ac            SEQUENCE (SIZE (0..MAX)) OF Ac,          -Absolute_Coordinates
   numrc         INTEGER         OPTIONAL,                -Number_of_Relative_
                                                           Coordinates
   rc            SEQUENCE (SIZE (0..MAX)) OF Rc,          -Relative_Coordinates
   rdsh          Rdsh            OPTIONAL,                -Road_Descriptor_Section_
                                                           Header
   rdsinf        Rdsinf          OPTIONAL                 -Road_Descriptor_Section_
                                                           Information
   }

Ac::= SEQUENCE {
   lat           Latitude,
   lon           Longitude,
   alt           Altitude        OPTIONAL
   }

Latitude::= CHOICE {
   lat_int       INTEGER         (-90000000..90000000),   -microdegrees
   lat_real      REAL            (-90..90)                -degrees
   }

Longitude::= CHOICE {
   lon_int       INTEGER         (-180000000..180000000), -microdegrees
   lon_real      REAL            (-180..180)              -degrees
   }

Altitude::= CHOICE {
   int           INTEGER         (-1000..10000),          -meters
   real          REAL            (-1..10)                 -kilometers
   }

Rc::= SEQUENCE {
   x             INTEGER,                                 -Relative_X
   y             INTEGER,                                 -Relative_Y
   z             INTEGER         OPTIONAL                 -Relative_Z
   }

Rdsh::= SEQUENCE {
   rdt           INTEGER         OPTIONAL,                -Road_Descriptor_Type
   rdsize        INTEGER         OPTIONAL,                -Road_Descriptor_Size
   numrdsinf     INTEGER         OPTIONAL                 -Number_of_Road_Descriptor_
                                                           Section_Information
```



```
   }

Rdsinf::= SEQUENCE {
   numrd          INTEGER         OPTIONAL,                 -Number_of_Road_Descriptors
   rds            SEQUENCE (SIZE (0..255)) OF Rd            -Road_Descriptors
   }

Rd::= SEQUENCE {
   rname          UTF8String      OPTIONAL,                 -Road_Name
   rnum           INTEGER         OPTIONAL,                 -Road_Number
   linkid         INTEGER         OPTIONAL,                 -Link_ID
   intersectname  UTF8String      OPTIONAL,                 -Intersection_Name
   intersectid    INTEGER         OPTIONAL,                 -Intersection_Number
   nodeid         INTEGER         OPTIONAL,                 -Node_ID
   brunnelname    UTF8String      OPTIONAL,                 -Brunnel_Name,
   offsecthdr     Offsecthdr      OPTIONAL,                 -Offset_Section_Header
   offsectinf     SEQUENCE (SIZE (0..255)) OF Offsectinf    -Offset_Section_Information
   }

Offsecthdr::= SEQUENCE {
   offsett        Offsett         OPTIONAL,                 -Offset_Type
   defoffsetorg   Defoffsetorg    OPTIONAL,                 -Definition_Offset_Origin
   offsetdu       Offsetdu        OPTIONAL,                 -Offset_Distance_Unit
   offsetau       Offsetau        OPTIONAL,                 -Offset_Angle_Unit
   numoffinf      INTEGER         OPTIONAL                  -Number_of_Offset_Section_
                                                             Information
   }

Offsett::= ENUMERATED {
   on-route,
   airline
   }

Defoffsetorg::= ENUMERATED {
   Start point,
   End point
   }

Offsetdu::= ENUMERATED {
   1m,
   10m,
   100m
   }

Offsetau::= ENUMERATED {
   degree,
   radian
   }

Offsectinf::= SEQUENCE {
   noffset        INTEGER         OPTIONAL,                 -Number_of_Offsets
   offsets        SEQUENCE (SIZE (0..255)) OF Offsets
   }

Offsets::= SEQUENCE {
   airlinedfo     INTEGER         OPTIONAL,                 -Airline Distance_from_Origin
   angle          Angle_type      OPTIONAL,                 -Angle
   vehicledrivingd INTEGER        OPTIONAL,                 -Vehicle_Driving_Distance
   direction      Direction_type  OPTIONAL                  -Direction
   }

Angle_type::= CHOICE {
   ang_int        INTEGER,
   ang_real       REAL
   }

Direction_type::= CHOICE {
   direc_int      INTEGER,
   direc_real     REAL
   }
```

```
   UDExt::= SEQUENCE {
      uDEExtHeader  UDExtHeader     OPTIONAL,                 -User-Defined Extension Header
      uDExtinf      UDExtinf                                  -User-Defined Extension
                                                               Information
      }

   UDExtHeader::= SEQUENCE {
      extID         ExtID           OPTIONAL,                 -Extension_Identifier
      extName       UTF8String      OPTIONAL,                 -Extension_Name
      extType       UTF8String      OPTIONAL                  -Extension_Type
      }

   ExtID::= CHOICE {
      extIDint      INTEGER,
      extIDstr      UTF8String
      }

   UDExtinf::= SEQUENCE {  -user-defined set of data elements and/or nested data frames
      byte          BYTE
      }
END
```

A.4 Structure as XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:Lrp1linf="Lrp1linf" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    elementFormDefault="qualified">
    <xsd:simpleType name="UTF8String">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="255"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="Linft">
        <xsd:restriction base="UTF8String">
            <xsd:enumeration value="point"/>
            <xsd:enumeration value="line"/>
            <xsd:enumeration value="area"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="Ct">
        <xsd:restriction base="UTF8String">
            <xsd:enumeration value="absolute"/>
            <xsd:enumeration value="relative"/>
            <xsd:enumeration value="grid"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="Bct">
        <xsd:restriction base="UTF8String">
            <xsd:enumeration value="utms"/>
            <xsd:enumeration value="utmp"/>
            <xsd:enumeration value="rectc"/>
            <xsd:enumeration value="pgrid1"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="Rct">
        <xsd:restriction base="UTF8String">
            <xsd:enumeration value="rct1"/>
            <xsd:enumeration value="rct2"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="Acu">
        <xsd:restriction base="UTF8String">
            <xsd:enumeration value="degree"/>
            <xsd:enumeration value="ddmmss"/>
            <xsd:enumeration value="second"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="Rcu">
        <xsd:restriction base="UTF8String">
            <xsd:enumeration value="secondu1"/>
            <xsd:enumeration value="secondu2"/>
```

```
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name=”Horerr”>
    <xsd:restriction base=”UTF8String”>
        <xsd:enumeration value=”basemap1”/>
        <xsd:enumeration value=”basemap2”/>
        <xsd:enumeration value=”basemap3”/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name=”Hterr”>
    <xsd:restriction base=”UTF8String”>
        <xsd:enumeration value=”error1”/>
        <xsd:enumeration value=”error2”/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name=”ExtType”>
    <xsd:restriction base=”UTF8String”/>
</xsd:simpleType>
<xsd:simpleType name=”Offsett”>
    <xsd:restriction base=”UTF8String”>
        <xsd:enumeration value=”on-route”/>
        <xsd:enumeration value=”airline”/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name=”Defoffsetorg”>
    <xsd:restriction base=”UTF8String”>
        <xsd:enumeration value=”Start point”/>
        <xsd:enumeration value=”End point”/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name=”Offsetdu”>
    <xsd:restriction base=”UTF8String”>
        <xsd:enumeration value=”1m”/>
        <xsd:enumeration value=”10m”/>
        <xsd:enumeration value=”100m”/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name=”Offsetau”>
    <xsd:restriction base=”UTF8String”>
        <xsd:enumeration value=”degree”/>
        <xsd:enumeration value=”radian”/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name=”Lsheader”>
    <xsd:sequence>
        <xsd:element name=”ptname” type=”UTF8String” minOccurs=”0”/>
        <xsd:element name=”ptversion” type=”Ptversion” minOccurs=”0”/>
        <xsd:element name=”linft” type=”Linft” minOccurs=”0”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”Lsinf”>
    <xsd:sequence>
        <xsd:element name=”cshi” type=”Csih” minOccurs=”0”/>
        <xsd:element name=”csi” type=”Csi” minOccurs=”0”/>
        <xsd:element name=”udext” type=”UDExt” minOccurs=”0”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”Ptversion”>
    xsd:choice>
        <xsd:element name=”vint” type=”xsd:integer”/>
        <xsd:element name=”vstr” type=”UTF8String”/>
    </xsd:choice>
</xsd:complexType>
<xsd:complexType name=”Csih”>
    <xsd:sequence>
        <xsd:element name=”ct” type=”Ct” minOccurs=”0”/>
        <xsd:element name=”bct” type=”Bct” minOccurs=”0”/>
        <xsd:element name=”rct” type=”Rct” minOccurs=”0”/>
        <xsd:element name=”acu” type=”Acu” minOccurs=”0”/>
        <xsd:element name=”rcu” type=”Rcu” minOccurs=”0”/>
        <xsd:element name=”horerr” type=”Horerr” minOccurs=”0”/>
```

```
            <xsd:element name="hterr" type="Hterr" minOccurs="0"/>
            <xsd:element name="numcsi" type="xsd:integer" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Csi">
        <xsd:sequence>
            <xsd:element name="numblock" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="bcodes" type="xsd:integer" minOccurs="0"
                                        maxOccurs="unbounded"/>
            <xsd:element name="numregion" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="rcodes" type="xsd:integer" minOccurs="0"
                                        maxOccurs="unbounded"/>
            <xsd:element name="numac" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="ac" type="Ac" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="numrc" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="rc" type="Rc" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="rdsh" type="Rdsh" minOccurs="0"/>
            <xsd:element name="rdsinf" type="Rdsinf" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Ac">
        <xsd:sequence>
            <xsd:element name="lat" type="Latitude"/>
            <xsd:element name="lon" type="Longitude"/>
            <xsd:element name="alt" type="Altitude" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Rc">
        <xsd:sequence>
            <xsd:element name="x" type="xsd:integer">
                <xsd:annotation>
                    <xsd:documentation>relative</xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="y" type="xsd:integer">
                <xsd:annotation>
                    <xsd:documentation>relative</xsd:documentation>
                </xsd:annotation>
            </xsd:element>
            <xsd:element name="z" type="xsd:integer" minOccurs="0">
                <xsd:annotation>
                    <xsd:documentation>relative</xsd:documentation>
                </xsd:annotation>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Rdsh">
        <xsd:sequence>
            <xsd:element name="rdt" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="rdsize" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="numrdsinf" type="xsd:integer" minOccurs="0"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Rdsinf">
        <xsd:sequence>
            <xsd:element name="numrd" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="rds" type="Rd" minOccurs="0" maxOccurs="255"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="Rd">
        <xsd:sequence>
            <xsd:element name="rname" type="UTF8String" minOccurs="0"/>
            <xsd:element name="rnum" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="linkid" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="intersectname" type="UTF8String" minOccurs="0"/>
            <xsd:element name="intersectid" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="nodeid" type="xsd:integer" minOccurs="0"/>
            <xsd:element name="brunnelname" type="UTF8String" minOccurs="0"/>
            <xsd:element name="offsecthdr" type="Offsecthdr" minOccurs="0"/>
            <xsd:element name="offsectinf" type="Offsectinf" minOccurs="0"
                                           maxOccurs="unbounded"/>
```

```
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”Offsecthdr”>
    <xsd:sequence>
        <xsd:element name=”offsett” type=”Offsett” minOccurs=”0”/>
        <xsd:element name=”defoffsetorg” type=”Defoffsetorg” minOccurs=”0”/>
        <xsd:element name=”offsetdu” type=”Offsetdu” minOccurs=”0”/>
        <xsd:element name=”offsetau” type=”Offsetau” minOccurs=”0”/>
        <xsd:element name=”numoffinf” type=”xsd:integer” minOccurs=”0”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”Offsectinf”>
    <xsd:sequence>
        <xsd:element name=”noffset” type=”xsd:integer” minOccurs=”0”/>
        <xsd:element name=”offsets” type=”Offsets” minOccurs=”0” maxOccurs=”255”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”Offsets”>
    <xsd:sequence>
        <xsd:element name=”airlinedfo” type=”xsd:integer” minOccurs=”0”/>
        <xsd:element name=”angle” type=”Angletype” minOccurs=”0”/>
        <xsd:element name=”vehicledrivingd” type=”xsd:integer” minOccurs=”0”/>
        <xsd:element name=”direction” type=”Directiontype” minOccurs=”0”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”Angletype”>
    <xsd:choice>
        <xsd:element name=”angint” type=”xsd:integer”/>
        <xsd:element name=”angreal” type=”xsd:float”/>
    </xsd:choice>
</xsd:complexType>
<xsd:complexType name=”Directiontype”>
    <xsd:choice>
        <xsd:element name=”direcint” type=”xsd:integer”/>
        <xsd:element name=”direcreal” type=”xsd:float”/>
    </xsd:choice>
</xsd:complexType>
<xsd:complexType name=”UDExt”>
    <xsd:sequence>
        <xsd:element name=”UDExtHeader” type=”UDExtHeader” minOccurs=”0”/>
        <xsd:element name=”UDExtinf” type=”UDExtinf”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”UDExtHeader”>
    <xsd:sequence>
        <xsd:element name=”extID” type=”ExtID” minOccurs=”0”/>
        <xsd:element name=”extName” type=”UTF8String” minOccurs=”0”/>
        <xsd:element name=”extType” type=”UTF8String” minOccurs=”0”>
            <xsd:annotation>
                <xsd:documentation>Extension_Type</xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”UDExtinf”>
    <xsd:sequence>
        <xsd:element name=”byte” type=”xsd:byte”/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name=”ExtID”>
    <xsd:choice>
        <xsd:element name=”extIDint” type=”xsd:integer”/>
        <xsd:element name=”extIDstr” type=”UTF8String”/>
    </xsd:choice>
</xsd:complexType>
<xsd:complexType name=”Longitude”>
    <xsd:choice>
        <xsd:element name=”lonint”>
            <xsd:annotation>
                <xsd:documentation>microdegrees</xsd:documentation>
            </xsd:annotation>
```

```
            <xsd:simpleType>
                <xsd:restriction base="xsd:integer">
                    <xsd:minInclusive value="-180000000"/>
                    <xsd:maxInclusive value="180000000"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="lonreal">
            <xsd:annotation>
                <xsd:documentation>degrees</xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:float">
                    <xsd:maxInclusive value="180"/>
                    <xsd:minInclusive value="-180"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:choice>
</xsd:complexType>
<xsd:complexType name="Latitude">
    <xsd:choice>
        <xsd:element name="latint">
            <xsd:annotation>
                <xsd:documentation>microdegrees</xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:integer">
                    <xsd:maxInclusive value="90000000"/>
                    <xsd:minInclusive value="-90000000"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="latreal">
            <xsd:annotation>
                <xsd:documentation>degrees</xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:float">
                    <xsd:maxInclusive value="90"/>
                    <xsd:minInclusive value="-90"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:choice>
</xsd:complexType>
<xsd:complexType name="Altitude">
    <xsd:choice>
        <xsd:element name="altint">
            <xsd:annotation>
                <xsd:documentation>meters</xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:integer">
                    <xsd:maxInclusive value="10000"/>
                    <xsd:minInclusive value="-1000"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element name="altreal">
            <xsd:annotation>
                <xsd:documentation>kilometers</xsd:documentation>
            </xsd:annotation>
            <xsd:simpleType>
                <xsd:restriction base="xsd:float">
                    <xsd:maxInclusive value="10"/>
                    <xsd:minInclusive value="-1"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
    </xsd:choice>
```

```
    </xsd:complexType>
    <xsd:element name="Lrp1linf">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="lsheader" type="Lsheader" minOccurs="0"/>
                <xsd:element name="lsinf" type="Lsinf"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

Annex B
(informative)

ALERT-C location reference, TPEG2 logical structure

B.1 Description of logical structure

B.1.1 General

Although it is intended to replace RDS-TMC by new services like TPEG Automotive Profile (TAP), in the meantime, easy adoption and migration into existing systems is needed. In addition to that, reuse of potential functionality from existing solutions is intended to be made possible. Whereas, for example, TPEG Applications can replace the former ALERT-C (TLR) method with its inheriting event lists, in terms of location referencing, it seems also to be a great potential to allow continuation of using ALERT-C-Location tables already in place, in addition to the new dynamic LRM. This gives the opportunity to code, very efficiently, not only already-existing location codes but also to allow any location to be coded. The duality of requirements has been adhered to in such a way that this data format specification defines data elements for both methodologies including necessary selection means.

B.1.2 Pre-coded ALERT-C location referencing

This annex defines the data structure of ALERT-C location referencing to be used in pre-coded LRC container. The method of ALERT-C location referencing is defined in the ALERT-C International Standard, ISO 14819-3.[11] ALERT-C defines a hierarchical structure of areas, lines, and points. The points are connected to predecessors and successors and belong to lines and areas. All elements in the structure are defined by one unique number which is used to refer to that particular part of the structure. In addition to the normal ALERT-C location referencing method, a precise location referencing method is also foreseen here.

The proposed method defined in the precise location referencing specification[5] defines attributes to code an arbitrary position in between two points being successors in the structure defined in Part 3 of the ALERT-C Standard.[11] These attributes are made optional because not every location will want to make use of it. In B.2 “TLR message components”, the sub-elements defined for ALERT-C location are specified.

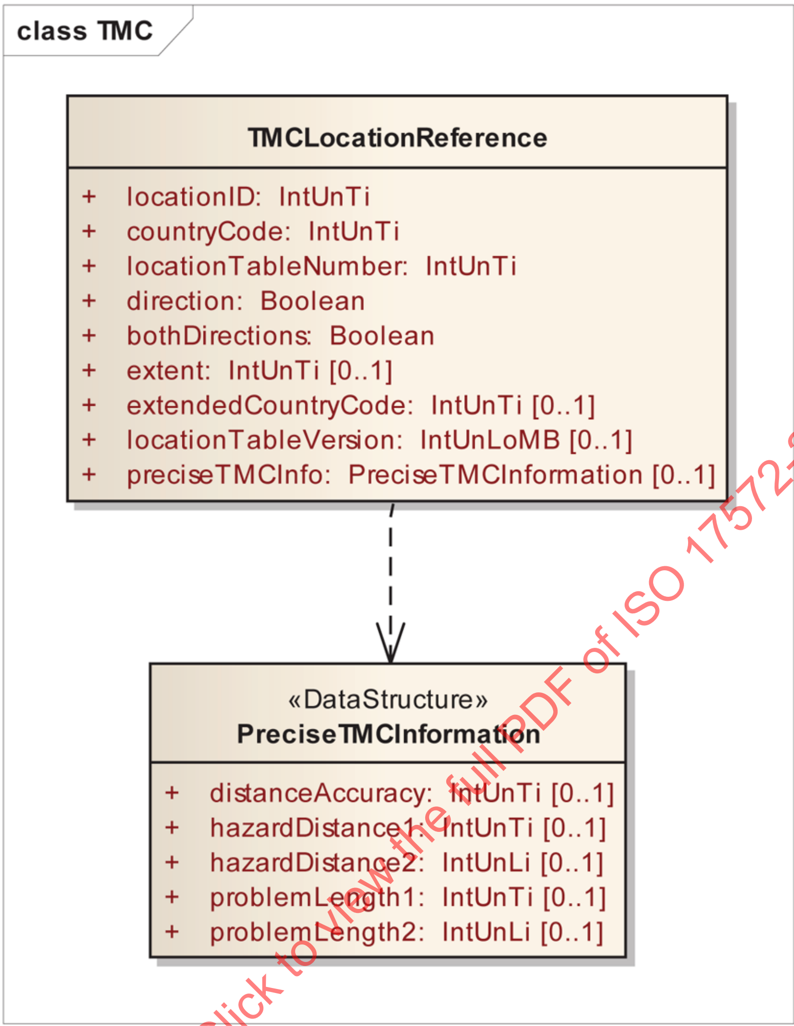

Figure B.1 — TPEG2 UML structure[15] of TMCLocationReference (ALERT-C)

B.2 TLR message components

B.2.1 TMCLocationReference

The TMC location stores a pre-coded location description including the possibility to code precisely a part of a road section in-between TMC locations.

Name	Type	Multiplicity	Description
locationID	IntUnTi	1	The actual location number (primary location)
countryCode	IntUnTi	1	Defines the different countries being responsible for the database creation. The table of country codes is specified in ISO 14819-3.
locationTableNumber	IntUnTi	1	The value is specified per country and can differ over time.
direction	Boolean	1	If the direction flag is true, it means positive direction, i.e. following to the successor in the location list.

Name	Type	Multiplicity	Description
bothDirections	Boolean	1	If set to true, the location shall be treated as being applicable for both directions. NOTE Be aware that ISO 18234 defines this information in two pieces (1: implicitly per event code and 2: as inversion bit in the control code).
extent	IntUnTi	0..1	Number of hops in the linked list of location codes. The direction is defined by the direction attribute. Absent value is defined 0 and means inside the current location.
extendedCountryCode	IntUnTi	0..1	In non-European countries, the full range of location tables per country code can be used provided that the Extended Country Code (ECC) is in use and transmitted (see IEC 62106:2000).
locationTableVersion	IntUnLoMB	0..1	If 1 byte, 4 bits for major and 3 bits for minor. If 2 bytes, 7 bits for major and 7 bits for minor. 3 bytes, not possible because TMC restricts to 0–99 for both.
preciseTMCInfo	PreciseTMCInformation	0..1	n.a.

B.3 TLR datatypes

B.3.1 PreciseTMCInformation

Defines the collection of optional precise TMC location coding.

Name	Type	Multiplicity	Description
distanceAccuracy	IntUnTi	0..1	0 = 100 m, 1 = 500 m, 2 = 1 km, 3 = more than 1 km 100m, if attribute is absent
hazardDistance1	IntUnTi	0..1	Upstream distance from primary location (against the direction of traffic), also known as D1. Range up to 25 km calculated in steps of 100 m.
hazardDistance2	IntUnLi	0..1	Same as hazardDistance1, but range up to 6 553 km.
problemLength1	IntUnTi	0..1	Distance: start of the problem to the end of the problem, which is not necessarily the primary location. Length up to 25km.
problemLength2	IntUnLi	0..1	Same as problemLength1, but length up to 6 553 km.

Annex C
(normative)

ALERT-C location reference, TPEG2 binary representation

C.1 General

This annex defines the TPEG2 binary representation[16] of TLR container.

C.2 TMC message components

C.2.1 List of generic component IDs

The GCId of root element is defined by the Location Referencing Container specification.[18]

Name	ID
TMCLocationReference	defined by LRC

C.2.2 TMCLocationReference

<TMCLocationReference(x)>:=	
<IntUnTi>(x),	: ID of this component
<IntUnLoMB>(lengthComp),	: Number of bytes in component
<IntUnLoMB>(lengthAttr),	: Number of bytes in attributes
<IntUnTi>(locationID),	: The actual location number (primary location)
<IntUnTi>(countryCode),	: Defines the different countries being responsible for the database creation. The table of country codes is specified in ISO 14819-3.
<IntUnTi>(locationTableNumber),	: The value is specified per country and can differ over time.
BitArray(selector),	
if (bit 0 of selector is set)	
<Boolean>(direction),	: If the direction flag is true, it means positive direction, i.e. following to the successor in the location list.
if (bit 1 of selector is set)	
<Boolean>(bothDirections),	: If set to true, the location shall be treated as being applicable for both directions. NOTE: Be aware that ISO 18234 defines this information in two pieces (1: implicitly per event code and 2: as inversion bit in the control code).
if (bit 2 of selector is set)	
<IntUnTi>(extent),	: Number of hops in the linked list of location codes. The direction is defined by the direction attribute. Absent value is defined 0 and means inside the current location.
if (bit 3 of selector is set)	

<IntUnTi>(extendedCountryCode),	: In non-European countries, the full range of location tables per country code can be used provided that the Extended Country Code (ECC) is in use and transmitted (see IEC 62106:2000).
if (bit 4 of selector is set)	
<IntUnLoMB>(locationTableVersion),	: If 1 byte, 4 bits for major and 3 bits for minor. If 2 bytes, 7 bits for major and 7 bits for minor. 3 bytes not possible because TMC restricts to 0–99 for both.
if (bit 5 of selector is set)	
<PreciseTMCInformation>(preciseTMCInfo);	

C.3 TLR datatypes

<PreciseTMCInformation>:=	
BitArray(selector),	
if (bit 0 of selector is set)	
<IntUnTi>(distanceAccuracy),	: 0 = 100 m, 1 = 500 m, 2 = 1 km, 3 = more than 1 km 100 m if attribute is absent
if (bit 1 of selector is set)	
<IntUnTi>(hazardDistance1),	: Upstream distance from primary location (against the direction of traffic), also known as D1. Range up to 25 km calculated in steps of 100 m.
if (bit 2 of selector is set)	
<IntUnLi>(hazardDistance2),	: Same as hazardDistance1, but range up to 6 553 km.
if (bit 3 of selector is set)	
<IntUnTi>(problemLength1),	: Distance: the start of the problem to the end of the problem, which is not necessarily the primary location. Length up to 25 km.
if (bit 4 of selector is set)	
<IntUnLi>(problemLength2);	: Same as problemLength1, but length up to 6 553 km.

Annex D
(normative)

ALERT-C location reference, TPEG2 XML representation

D.1 General

This annex defines the TPEG2 XML representation[17] of TLR container.

D.2 TLR message components

```
<xs:element name=”TMCLocationReference” type=”TMCLocationReference”/>
<xs:complexType name=”TMCLocationReference”>
 <xs:complexContent>
 <xs:extension base=”tsf:ApplicationRootMessageML”>
 <xs:sequence>
 <xs:element name=”locationID” type=”tdt:IntUnTi”/>
 <xs:element name=”countryCode” type=”tdt:IntUnTi”/>
 <xs:element name=”locationTableNumber” type=”tdt:IntUnTi”/>
 <xs:element name=”direction” type=”tdt:Boolean”/>
 <xs:element name=”bothDirections” type=”tdt:Boolean”/>
 <xs:element name=”extent” type=”tdt:IntUnTi” minOccurs=”0”/>
 <xs:element name=”extendedCountryCode” type=”tdt:IntUnTi” minOccurs=”0”/>
 <xs:element name=”locationTableVersion” type=”tdt:IntUnLoMB” minOccurs=”0”/>
 <xs:element name=”preciseTMCInfo” type=”PreciseTMCInformation” minOccurs=”0”/>
 </xs:sequence>
 </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

D.3 TLR datatypes

```
<xs:complexType name=”PreciseTMCInformation”>
 <xs:sequence>
 <xs:element name=”distanceAccuracy” type=”tdt:IntUnTi” minOccurs=”0”/>
 <xs:element name=”hazardDistance1” type=”tdt:IntUnTi” minOccurs=”0”/>
 <xs:element name=”hazardDistance2” type=”tdt:IntUnLi” minOccurs=”0”/>
 <xs:element name=”problemLength1” type=”tdt:IntUnTi” minOccurs=”0”/>
 <xs:element name=”problemLength2” type=”tdt:IntUnLi” minOccurs=”0”/>
 </xs:sequence>
</xs:complexType>
```

D.4 Full TLR schema definition

```
<?xml version=”1.0” encoding=”UTF-8”?>
<!- This XML schema is generated with tpegUMLconverter V2.3 ->
<xs:schema xmlns=”http://www.tisa.org/TPEG/TLR_2_0”
  targetNamespace=”http://www.tisa.org/TPEG/TLR_2_0”
  xmlns:xs=”http://www.w3.org/2001/XMLSchema”
  xmlns:tsf=”http://www.tisa.org/TPEG/SFW_1_1”
  xmlns:tdt=”http://www.tisa.org/TPEG/TPEGDataTypes_2_0”
  elementFormDefault=”qualified”
  attributeFormDefault=”qualified”>
  <xs:import namespace=”http://www.tisa.org/TPEG/SFW_1_1” schemaLocation=”SFW_1_1.xsd”/>
  <xs:import namespace=”http://www.tisa.org/TPEG/TPEGData-
Types_2_0” schemaLocation=”TDT_2_0.xsd”/>
  <xs:element name=”TMCLocationReference” type=”TMCLocationReference”/>
  <xs:complexType name=”TMCLocationReference”>
    <xs:complexContent>
      <xs:extension base=”tsf:ApplicationRootMessageML”>
        <xs:sequence>
```

```
          <xs:element name="locationID" type="tdt:IntUnTi"/>
          <xs:element name="countryCode" type="tdt:IntUnTi"/>
          <xs:element name="locationTableNumber" type="tdt:IntUnTi"/>
          <xs:element name="direction" type="tdt:Boolean"/>
          <xs:element name="bothDirections" type="tdt:Boolean"/>
          <xs:element name="extent" type="tdt:IntUnTi" minOccurs="0"/>
          <xs:element name="extendedCountryCode" type="tdt:IntUnTi" minOccurs="0"/>
          <xs:element name="locationTableVersion" type="tdt:IntUnLoMB" minOccurs="0"/>
          <xs:element name="preciseTMCInfo" type="PreciseTMCInformation" minOccurs="0"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="PreciseTMCInformation">
    <xs:sequence>
      <xs:element name="distanceAccuracy" type="tdt:IntUnTi" minOccurs="0"/>
      <xs:element name="hazardDistance1" type="tdt:IntUnTi" minOccurs="0"/>
      <xs:element name="hazardDistance2" type="tdt:IntUnLi" minOccurs="0"/>
      <xs:element name="problemLength1" type="tdt:IntUnTi" minOccurs="0"/>
      <xs:element name="problemLength2" type="tdt:IntUnLi" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Annex E
(informative)

Korean node-link location reference, TPEG2 logical structure

E.1 Description of the logical structure

This annex defines the data structure of KoreanNodeLinkLocationReference (see Figure E.1) to be used in pre-coded LRC container. The method of Korean node-link location referencing is described in 8.4.

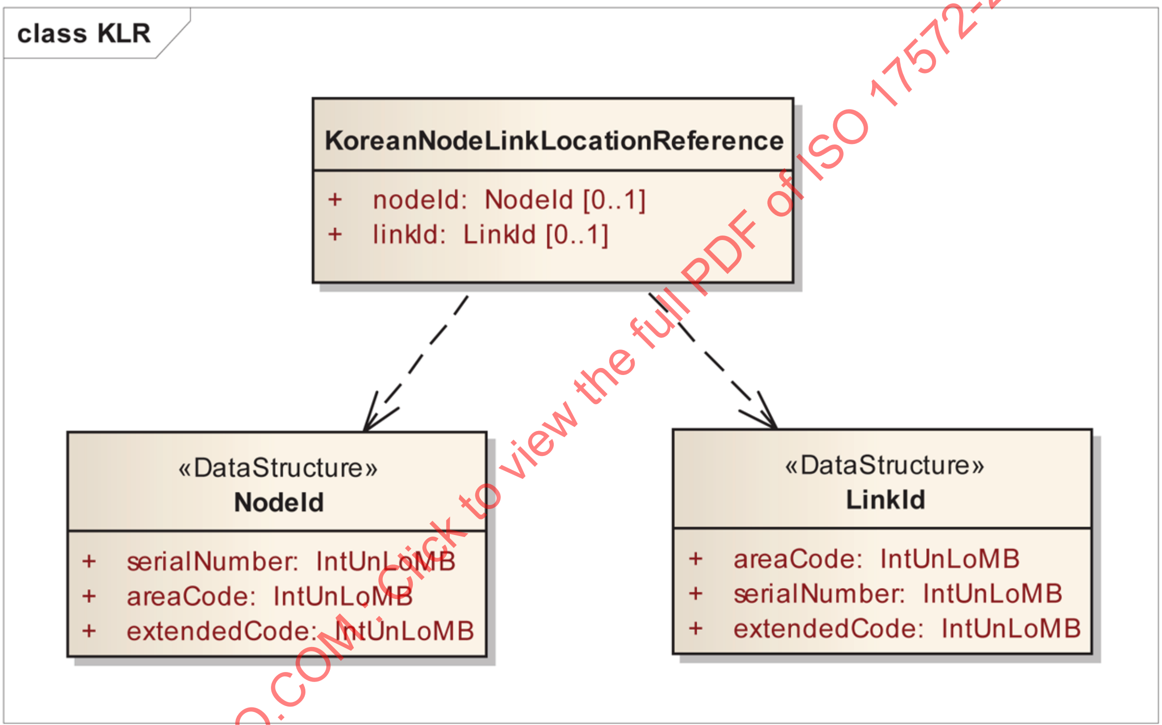

Figure E.1 — TPEG2 UML structure[15] of KoreanNodeLinkLocationReference

E.2 KLR message components

E.2.1 KoreanNodeLinkLocationReference

Name	Type	Multiplicity	Description
nodeId	NodeId	0..1	n.a.
linkId	LinkId	0..1	n.a.

E.3 KLR datatypes

E.3.1 LinkId

Name	Type	Multiplicity	Description
areaCode	IntUnLoMB	1	n.a.
serialNumber	IntUnLoMB	1	n.a.
extendedCode	IntUnLoMB	1	n.a.

E.3.2 NodeId

Name	Type	Multiplicity	Description
serialNumber	IntUnLoMB	1	n.a.
areaCode	IntUnLoMB	1	n.a.
extendedCode	IntUnLoMB	1	n.a.

Annex F
(normative)

Korean node-link location reference, TPEG2 binary representation

F.1 General

This annex defines the TPEG2 binary representation[16] of Korean node-link location reference container.

F.2 KLR message components

F.2.1 List of generic component IDs

The GCId of the root element is defined by Location Referencing Container specification.[18]

Name	ID
KoreanNodeLinkLocationReference	x

F.2.2 KoreanNodeLinkLocationReference

<KoreanNodeLinkLocationReference(x)>:=	
<IntUnTi>(x),	: ID of this component
<IntUnLoMB>(lengthComp),	: Number of bytes in component
<IntUnLoMB>(lengthAttr),	: Number of bytes in attributes
BitArray(selector),	
if (bit 0 of selector is set)	
<NodeId>(nodeId),	
if (bit 1 of selector is set)	
<LinkId>(linkId);	

F.3 KLR datatypes

F.3.1 LinkId

<LinkId>:=	
<IntUnLoMB>(areaCode),	
<IntUnLoMB>(serialNumber),	
<IntUnLoMB>(extendedCode);	

F.3.2 NodeId

<NodeId>:=	
<IntUnLoMB>(serialNumber),	
<IntUnLoMB>(areaCode),	
<IntUnLoMB>(extendedCode);	