OpenLR™

White Paper

Version: 1.5 revision 2

An open standard for encoding, transmitting and decoding location references in digital maps

License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

1. Definitions

a. **"Adaptation"** means a work based upon the Work, or upon the Work and other pre-existing works, such as a translation, adaptation, derivative work, arrangement of music or other alterations of a literary or artistic work, or phonogram or performance and includes cinematographic adaptations or any other form in which the Work may be recast, transformed, or adapted including in any form recognizably derived from the original, except that a work that constitutes a Collection will not be considered an Adaptation for the purpose of this License. For the avoidance of doubt, where the Work is a musical work, performance or phonogram, the synchronization of the Work in timed-relation with a moving image ("synching") will be considered an Adaptation for the purpose of this License.

b. **"Collection"** means a collection of literary or artistic works, such as encyclopedias and anthologies, or performances, phonograms or broadcasts, or other works or subject matter other than works listed in Section 1(f) below, which, by reason of the selection and arrangement of their contents, constitute intellectual creations, in which the Work is included in its entirety in unmodified form along with one or more other contributions, each constituting separate and independent works in themselves, which together are assembled into a collective whole. A work that constitutes a Collection will not be considered an Adaptation (as defined above) for the purposes of this License.

c. **"Distribute"** means to make available to the public the original and copies of the Work through sale or other transfer of ownership.

d. **"Licensor"** means the individual, individuals, entity or entities that offer(s) the Work under the terms of this License.

e. **"Original Author"** means, in the case of a literary or artistic work, the individual, individuals, entity or entities who created the Work or if no individual or entity can be identified, the publisher; and in addition (i) in the case of a performance the actors, singers, musicians, dancers, and other persons who act, sing, deliver, declaim, play in, interpret or otherwise perform literary or artistic works or expressions of folklore; (ii) in the case of a phonogram the producer being the person or legal entity who first fixes the sounds of a performance or other sounds; and, (iii) in the case of broadcasts, the organization that transmits the broadcast.

f. **"Work"** means the literary and/or artistic work offered under the terms of this License including without limitation any production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression including digital form, such as a book, pamphlet and other writing; a lecture, address, sermon or other work of the same nature; a dramatic or dramatico-musical work; a choreographic work or entertainment in dumb show; a musical composition with or without words; a cinematographic work to which are assimilated works expressed by a process analogous to cinematography; a work of drawing, painting, architecture, sculpture, engraving or lithography; a photographic work to which are assimilated works expressed by a process analogous to photography; a work of applied art; an illustration, map, plan, sketch or three-dimensional work relative to geography, topography, architecture or science; a performance; a broadcast; a phonogram; a compilation of data to the extent it is protected as a copyrightable work; or a work performed by a variety or circus performer to the extent it is not otherwise considered a literary or artistic work.

g. **"You"** means an individual or entity exercising rights under this License who has not previously violated the terms of this License with respect to the Work, or who has received express permission from the Licensor to exercise rights under this License despite a previous violation.
h. **"Publicly Perform"** means to perform public recitations of the Work and to communicate to the public those public recitations, by any means or process, including by wire or wireless means or public digital performances; to make available to the public Works in such a way that members of the public may access these Works from a place and at a place individually chosen by them; to perform the Work to the public by any means or process and the communication to the public of the performances of the Work, including by public digital performance; to broadcast and rebroadcast the Work by any means including signs, sounds or images.
i. **"Reproduce"** means to make copies of the Work by any means including without limitation by sound or visual recordings and the right of fixation and reproducing fixations of the Work, including storage of a protected performance or phonogram in digital form or other electronic medium.

2. Fair Dealing Rights. Nothing in this License is intended to reduce, limit, or restrict any uses free from copyright or rights arising from limitations or exceptions that are provided for in connection with the copyright protection under copyright law or other applicable laws.

3. License Grant. Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

a. to Reproduce the Work, to incorporate the Work into one or more Collections, and to Reproduce the Work as incorporated in the Collections; and,
b. to Distribute and Publicly Perform the Work including as incorporated in Collections.
c. For the avoidance of doubt:
    i. **Non-waivable Compulsory License Schemes**. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme cannot be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License;
    ii. **Waivable Compulsory License Schemes**. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme can be waived, the Licensor waives the exclusive right to collect such royalties for any exercise by You of the rights granted under this License; and,
    iii. **Voluntary License Schemes**. The Licensor waives the right to collect royalties, whether individually or, in the event that the Licensor is a member of a collecting society that administers voluntary licensing schemes, via that society, from any exercise by You of the rights granted under this License.

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats, but otherwise you have no rights to make Adaptations. Subject to Section 8(f), all rights not expressly granted by Licensor are hereby reserved.

4. Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

a. You may Distribute or Publicly Perform the Work only under the terms of this License. You must include a copy of, or the Uniform Resource Identifier (URI) for, this License with every copy of the Work You Distribute or Publicly Perform. You may not offer or impose any terms on the Work that restrict the terms of this License or the ability of the recipient of the Work to exercise the rights granted to that recipient under the terms of the License. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties with every copy of the Work You Distribute or Publicly Perform. When You Distribute or Publicly Perform the Work, You may not impose any effective technological measures on the Work that restrict the ability of a recipient of the Work from You to exercise the rights granted to that recipient under the terms of the License. This Section 4(a) applies to the Work as incorporated in a Collection, but this does not require the Collection apart from the Work itself to be made subject to the terms of this License. If You create a Collection, upon notice from any Licensor You must, to the extent practicable, remove from the Collection any credit as required by Section 4(b), as requested.

b. If You Distribute, or Publicly Perform the Work or Collections, You must, unless a request has been made pursuant to Section 4(a), keep intact all copyright notices for the Work and provide, reasonable to the medium or means You are utilizing: (i) the name of the Original Author (or pseudonym, if applicable) if supplied, and/or if the Original Author and/or Licensor designate another party or parties (e.g., a sponsor institute, publishing entity, journal) for attribution ("Attribution Parties") in Licensor's copyright notice, terms of service or by other reasonable means, the name of such party or parties; (ii) the title of the Work if supplied; (iii) to the extent reasonably practicable, the URI, if any, that Licensor specifies to be associated with the Work, unless such URI does not refer to the copyright notice or licensing information for the Work. The credit required by this Section 4(b) may be implemented in any reasonable manner; provided, however, that in the case of a Collection, at a minimum such credit will appear, if a credit for all contributing authors of the Collection appears, then as part of these credits and in a manner at least as prominent as the credits for the other contributing authors. For the avoidance of doubt, You may only use the credit required by this Section for the purpose of attribution in the manner set out above and, by exercising Your rights under this License, You may not implicitly or explicitly assert or imply any connection with, sponsorship or endorsement by the Original Author, Licensor and/or Attribution Parties, as appropriate, of You or Your use of the Work, without the separate, express prior written permission of the Original Author, Licensor and/or Attribution Parties.

c. Except as otherwise agreed in writing by the Licensor or as may be otherwise permitted by applicable law, if You Reproduce, Distribute or Publicly Perform the Work either by itself or as part of any Collections, You must not distort, mutilate, modify or take other derogatory action in relation to the Work which would be prejudicial to the Original Author's honor or reputation.

5. Representations, Warranties and Disclaimer

UNLESS OTHERWISE MUTUALLY AGREED TO BY THE PARTIES IN WRITING, LICENSOR OFFERS THE WORK AS-IS AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE WORK, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTIBILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO SUCH EXCLUSION MAY NOT APPLY TO YOU.

6. Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

7. Termination

a. This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Collections from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.
b. Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.

8. Miscellaneous

a. Each time You Distribute or Publicly Perform the Work or a Collection, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.
b. If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.
c. No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.
d. This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.
e. The rights granted under, and the subject matter referenced, in this License were drafted utilizing the terminology of the Berne Convention for the Protection of Literary and Artistic Works (as amended on September 28, 1979), the Rome Convention of 1961, the WIPO Copyright Treaty of 1996, the WIPO Performances and Phonograms Treaty of 1996 and the Universal Copyright Convention (as revised on July 24, 1971). These rights and subject matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.

Version information

Version 1.0

Changes:

- Initial version

Author:	TomTom International B.V
Date:	2009-09-08

Version 1.1

Changes:

- Errata for v1.0 included (sections 4.4.4, 4.6.2, and 9.1)
- Note added for physical format version number (section 4.5.1)
- Section 3.4, Rule-4: U-turns nodes are also valid nodes; invalid nodes might be chosen exceptionally if no other possibility exists
- Encoding steps, Step 1: turn restrictions are optional (section 7.1)
- Encoding steps, Step 9: explanation how to deal with lines longer than 15km (section 7.1)
- Decoding steps, Step 3: find candidate lines directly if no candidate nodes can be found (section 8.1)
- Decoding steps, Step 4: LRP attributes may vary between encoder and decoder map so that values don’t need to match exactly (section 8.1)
- Decoding steps, Step 5: explanation how to deal with a situation where start and end line are equal; retrying other candidate line pairs if no valid shortest-path can be determined (section 8.1)
- Explanation added for handling of prohibited turns in decoding example (section 9.2)

Author:	TomTom International B.V
Date:	2009-12-11

Version 1.2

Changes:

- Description of the XML format for OpenLR™ location references (new section 7)
- Schema definition of the XML format (new section 8)
- Old sections 7 – 10 have moved to sections 9 – 12
- Encoding and decoding example extended with the XML format (section 11)
- Encoding process (section 9) and decoding process (section 10) now handle (abstract) physical formats instead of the concrete binary format.

Author:	TomTom International B.V.
Date:	2010-02-01

Version 1.3

Changes:

- Update explanation of common terms and abbreviations
- Section 2: Introduction of concepts has been updated to the point location support
- Section 3.3: Updated sub-section explaining the offsets
- Section 3.5: Data format Rule-3 has been updated
- Section 3.5: New data format Rule-5
- Section 4: New section for a high-level example of line locations
- Section 5: New section for a high-level example of point locations
- Section 6.4.7: New data type for point locations - orientation
- Section 6.4.8: New data type for point locations - side of road
- Section 6.5.1: Actual version of the binary format updated
- Section 6.5.4: New flag - point location
- Section 6.6.2: New format for offsets (percentage representation)
- Section 7: Update of the binary physical format
- Section 7.1: Simpler overview of the data format for line locations
- Section 7.2: Explanation of the different point location types
- Section 7.3: Characteristics for determining the location type

- Section 7.4: Changelist from version 2 to version 3 (binary format)
- Section 8.6: New data elements for point locations in the XML format
- Section 8.8: New data types supporting point locations in the XML format
- Section 9: Updated schema definition for the XML format
- Section 10: Update of the encoding steps to support point locations
- Section 10.3: New section explaining the offset handling during the encoding phase
- Section 11: Update of the decoding steps to support point locations
- Section 11.3: New section explaining the offset handling during the decoding phase
- Section 12.1: Update of the example for line location encoding and decoding
- Section 12.2: New section showing a small encoding and decoding example for all point location types

Author: TomTom International B.V.
Date: 2010-05-17

Version 1.4

Changes:

- Fix bearing value of the last location reference point in PointAlongLine and PoiWithAccessPoint example
- Document has been split into several main chapters (A – I) following a logical model of chapters providing mandatory information on the OpenLR™ standard, chapters providing recommendations (optional for the OpenLR™ standard) and other chapters for the introduction, examples and appendices.
- Introduction explains the structure of the document and also highlights the chapters covering the OpenLR™ standard.
- Location type section has moved to the concepts chapter.
- Location and map requirements have moved to an own chapter.
- The data types used in the logical data format are defined.
- The logical data format explains the "SideOfRoad" and "Orientation" information.
- The data format rules have been revised. Rule - 4 has been relaxed and has been extended by a recommendation.
- Problems with the relative coordinates in the binary data format have been identified and explained.
- A recommendation has been added to the offsets in the binary data format. When offsets in version 3 are used, then the encoder might add an additional intermediate LR-Point when high accuracy is required.
- Explanations regarding the encoder implementation have been revised. Some possible improvements have been added.
- Explanations regarding the decoder implementation have been revised. The execution of a direct line search has been further explained.

Author: TomTom International BV
Date: 2010-09-14

Version 1.5

Changes:

- The Introduction has been revised and area locations have been added.
- The location types have been extended with the support of area locations.
- The chapter on requirements has been revised and area locations have been added.
- The logical data format has been revised. Location reference points are now built from so called building blocks. The composition of the location reference points is now depending on the location type.
- New building blocks: Radius and Number of Rows/Columns
- The deprecated description of the old offset format has been removed.
- The binary physical format has been extended with the support for area locations.
- The deprecated description of the binary format version 2 has been removed.
- The XML physical format has been extended with the support for area locations.
- The recommendations for the encoder have been revised and split into several sections describing the process for each location type separately.
- The recommendations for the decoder have been revised and split into several sections describing the process for each location type separately.
- The example chapter has been extended with examples for the area location types.

Author:	TomTom International B.V. Deutsches Zentrum fuer Luft- und Raumfahrt e. V., (DLR, German Aerospace Center)
Date:	2012-01-09

Version 1.5 revision 2:

Changes:

- Includes errata for revision 1 (corrects binary format specification for closed line location references in section 8.3.5 and 8.4).

Author:	TomTom International B.V. Deutsches Zentrum fuer Luft- und Raumfahrt e. V., (DLR, German Aerospace Center)
Date:	2012-01-19

Content

Chapter A

Introduction

1. Introduction

Communication of spatial information involves the communication of location. The communication chain of a machine readable location can be described as encoding the location at the sender side, transfer of the code to the receiving system and decoding the code at the receiver side. The process of encoding a location is also called Location Referencing. This White Paper assumes a map on the sender side from which the location is encoded and a map on the receiver side in which the decoded location is found back. An obvious way of Location Referencing is using geographic coordinates. One important disadvantage of using coordinates is that it assumes identical maps at both sides of the communication chain which often is not the case. As a consequence, the decoded location may not be found back in the receiver map, or decoding (e.g. map-matching) may be inaccurate or ambiguous. OpenLR™ is a method for location referencing which does not have this disadvantage. It accommodates requirements of communication of location between systems which have dissimilar maps. OpenLR™ is communication channel independent. It takes bandwidth requirements into account in the sense that OpenLR™ requires minimal bandwidth.

OpenLR™ has been designed for the use case of transferring traffic information from a centre to in-vehicle systems, built-in or used as an add-on (PND, Smart Phone). The information transferred can consist of the current traffic situation at a certain location, a traffic forecast or special alerts. The corresponding locations are roads, a list of connected roads, points of interest, or areas.

The most well-known and most used method to transfer traffic information today is called RDS-TMC. The Location Referencing used in RDS-TMC makes use of pre-coded locations. These pre-coded locations are added to the corresponding locations in the map by the map providers of the sending and the receiving map. The process of encoding is looking up the location code in the map belonging to the relevant location. The process of decoding is finding back the location code in the map and looking up the corresponding location.

From the fact that RDS-TMC makes use of pre-coded location is follows that the amount of locations fit to be transferred is limited. OpenLR™ does not have that restriction. With OpenLR™ every location in a map can be transferred.

The technical goal of OpenLR™ is to specify a location referencing system with the following objectives:

- *dynamic generation of a location reference, including urban and low level roads*
- *compact format for transmission*
- *handling map differences caused by different map vendors or versions*

The industry goal of OpenLR™ is wide-scale adoption by the industry at large. OpenLR™ is therefore proposed as an open standard in an Open Source framework. It shall be usable for anyone dealing with locations and transmitting these between systems having dissimilar maps.

While the standard is developed and maintained by TomTom International B.V., everyone is invited to contribute to its further development.

This document defines the OpenLR™ standard. The Chapter A gives an introduction to the concepts of OpenLR™. Chapter B explains the required location and map properties. Chapter C defines the logical data format for location references and Chapter D and Chapter E define the physical formats for OpenLR™ location references.

The Chapter F provides recommendations for the implementation of an OpenLR™ encoder and Chapter G provides recommendations for the implementation of an OpenLR™ decoder. Examples for the different location types are explained in Chapter H and the appendix is placed in Chapter I.

The following chapters define the OpenLR™ standard:

- Chapter B
  Requirements
- Chapter C
  Logical data format
- Chapter D
  Physical data format - Binary
- Chapter E
  Physical data format - XML

Common terms and abbreviations

The following tables explain common terms and abbreviations used in this document and in the context of location referencing

Term	Description
Area	A two-dimensional part of the surface of the earth which is bounded by a closed curve. An area may cover parts of the road network, then covering several roads or parts of roads.
Bit	A bit is a binary digit, taking a logical value of either "1" or "0" (also referred to as "true" or "false" respectively).
Byte	A byte is a basic unit of measurement of information storage in computer science. There is no standard but a byte most often consists of eight bits.
Candidate line	A line object in the network of a decoder which might be referenced by the Location Code (as generated by the encoder).
Candidate node	A node object in the network of a decoder which might be referenced by the Location code (as generated by the encoder).
Decoder	A software component which decodes a Location Code and finds the corresponding location back in a map.
Encoder	A software component which generates a Location code for a location in a map.
Intermediate	An internal Location Reference Point (LRP); neither the start LRP nor the end LRP (destination).
Intermediate location reference point	see “Intermediate”
Line	A one-dimensional representation of a road or part of road in a road network. A line starts and ends at a node. It is directed, this means two-way traffic flow is represented by two (directed) lines, one per direction.
Location	A specification of the position on the earth surface of an object in a digital map.
Location reference	Location Code, created according to a specific set of rules, used to reference a location.
Location reference path	A route in a road network in a digital map which is referenced by the location reference. This path might be longer than the location itself.
Location Reference Point (LRP)	A description of an object in a digital map which consists of coordinates and additional information about a line in the map.
Map	A Map is the geospatial representation of an area on the earth surface.
Node	A zero-dimensional object in the road network. A node acts as start and end for lines.
Offset	The specification of a position along a path to indicate the start or the end of a location.
Point	A zero-dimensional element that specifies a geometric location. One coordinate pair specifies the location.
Route	A route is a collection of line objects in a digital map connecting a departure location and a destination location, defined according to certain criteria which might include time distance or cost.
TMC	Traffic Message Channel; public or commercial traffic information broadcast channel using RDS (Radio-Data-System).
XML	Extensible Markup Language

Table 1: Explanation of common terms

Abbreviation	Description
AF	Attribute Flag – a flag which indicates that the binary representation of the location reference includes attribute information
ArF	Area Flags – flags which indicate that the location reference describes an area
BEAR	Bearing – angle between the direction to a point in the network and a reference direction (here: the true North)
COORD	Coordinates – a pair of two values (longitude and latitude) representing a position in a two-dimensional network
DNP	Distance to Next Point – the length in meter to the next location reference point (measured along the location reference path between these two LRP)
FOW	Form Of Way – Certain aspects of the physical form that a line takes. It is based on a number of certain physical and traffic properties.
FRC	Functional Road Class – A classification based on the importance of the role that the line performs in the connectivity of the total road network.
lat	Latitude – geographic coordinate used for north-south measurement
LFRCNP	Lowest Functional Road Class to Next Point
lon	Longitude – geographic coordinate used for east-west measurement
LRP	Location Reference Point – a point of the location which holds relevant information for a map-independent location reference
NCOLS	Number of columns in a grid
NOFF	Negative Offset – distance in meter along the location reference path between the real end of the location and the end of the location reference path
NOffF	Negative Offset Flag – a flag which indicates that a negative offset is included in the location reference
NROWS	Number of rows in a grid
ORI	Orientation – determines where a point is located with respect to the driving direction of a line
POFF	Positive Offset – distance in meter along the location reference path between the start of the location reference path and the real start of the location
POffF	Positive Offset Flag – a flag which indicates that a negative offset is included in the location reference
POI	Point of interest
RAD	Radius
RFU	Reserved for future use – a bit in a binary stream which does not have a use yet
SOR	Side of road – determines where a point is located with respect to a line
TMC	Traffic Message Channel – is a technology for delivering traffic and travel information to drivers using pre-defined points and segments in a network
VER	Version – Version information

Table 2: Explanation of common abbreviations

2. Concepts

OpenLR™ describes a method and a format for encoding, transmitting and decoding (map-independent) references of locations. Locations are objects in a digital map, like points, paths and areas. The method makes it possible to encode a location in a map, send it to a system having another (possibly different) map and find the location back in this receiving map. Provided that both, encoder map and decoder map are meeting "navigable map" standards (in terms of accuracy and content), the encoder does not need to know about the decoder map and the decoder also does not have to care about the map used for encoding the location. The format to transmit such location reference is compact so that it can be used in systems having bandwidth restrictions.

The OpenLR™ standard can handle line locations (e.g. paths), point locations (e.g. POIs) and area locations (e.g. regions) in a digital map. OpenLR™ can be used for transmitting locations between different maps. One use case is the distribution of traffic information from a traffic information centre to several customers, like car drivers.

OpenLR™ can handle locations which are bound to the road network but also locations which can be everywhere on earth (not bound to the road network). A line location is an example for a location which is bound to the road network and a simple geo-coordinate is an example for a location which is not bound to the road network.

The main idea of OpenLR™ is to describe a location with an ordered set of location reference points (LRPs). This set of LRPs uniquely identifies the location in a digital map and is called the location reference. A physical representation of the LRPs can be used to transmit location information from one digital map to another. Every single LRP specifies a position or a line in a digital map.

Locations which are not bound to the road network are expressed through geo-coordinates or a set of geo-coordinates. Additional parameter may indicate the expansion of an area location.

The main idea for locations which are bound to the road network is covering the location with a concatenation of (several) shortest-paths. The concatenation of such shortest-paths shall cover the location completely. Each shortest-path is specified by information about its start line and its end line. This information is combined in the location reference points (LRPs). The LRPs are ordered from the start of the location to the end of the location and the shortest-path between two subsequent LRPs covers a part of the location. The concatenation of all these shortest-paths covers the location completely and this path is called the location reference path. The location reference path may be longer than the original location and offsets trim this path down to the size of the location path. Offsets are also used to define a location on a line more precisely (e.g. point locations along a line) than using the start and end node of that line.

The role of the OpenLR™ encoder is to determine the number and the positions of the location reference points to describe the location uniquely. The role of the OpenLR™ decoder is to resolve the received location reference points to a location in its own map.

The physical format for transmitting location references is also described in the OpenLR™ standard. A binary format is compact and can be used for the transmission of location reference data where the data size is important to be small. An XML format can be used if the need of having a compact format is less important and the existing environment for data transmission may already be based on XML. Additional formats compliant to other protocols like Datex-II or TPEG may be specified in separate documents.

The data used for a location reference point relies on map data common for navigable digital road maps. This common data includes a geographic position and if bound to the road network also line attributes like "functional road class" and "form of way".

The location properties and the map requirements are outlined in the following sections. The OpenLR™ standard is developed and maintained by TomTom International B.V. but it is intended as an open standard that can easily be used by other parties. Any contributions to the further development are welcome and can be discussed and submitted via the OpenLR™ website http://www.openlr.org.

3. Location types

OpenLR™ supports several types of locations. The following table gives an overview of the supported types and also provides a link to the section where the specific type is explained in detail.

Name	Category	Details
Line	Line location	see section 3.1
Geo-Coordinate	Point location	see section 3.2.1
PointAlongLine	Point location	see section 3.2.2
PoiWithAccessPoint	Point location	see section 3.2.3
Circle	Area location	see section 3.3.1
Rectangle	Area location	see section 3.3.2
Grid	Area location	see section 3.3.3
Polygon	Area location	see section 3.3.4
ClosedLine	Area location	see section 3.3.5

Table 3: Overview of the location types

3.1 Line locations

A line location is a one-dimensional part of a road or a road network. Line locations consist of at least one line. If more than one line is used to describe the location then these lines need to form a connected path in the road network. Offsets can be used to identify locations which do not start or end exactly at a network node. Examples of line locations are jams, (temporary) speed limits, (calculated) routes. The following figures (see Figure 1 and Figure 2) show different types of line locations where the location is marked in red and the position of the offsets are shown as green dots.

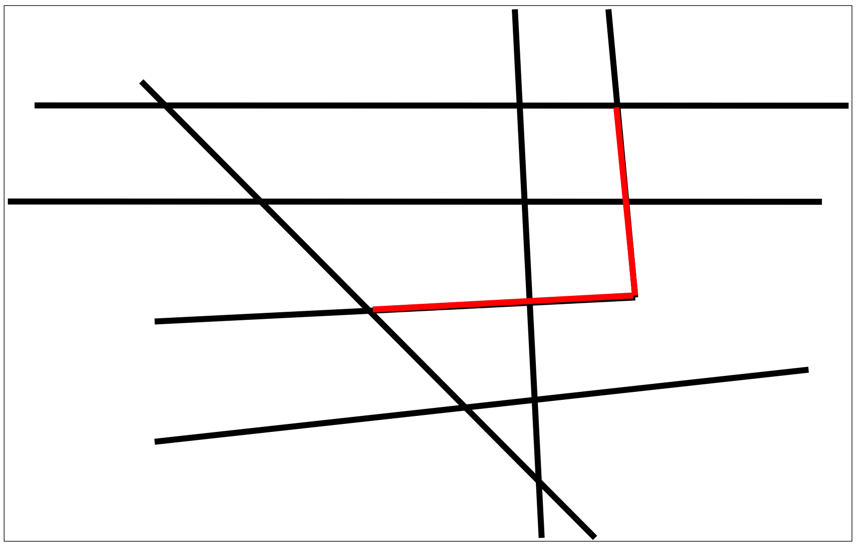

Figure 1: Line location without offsets

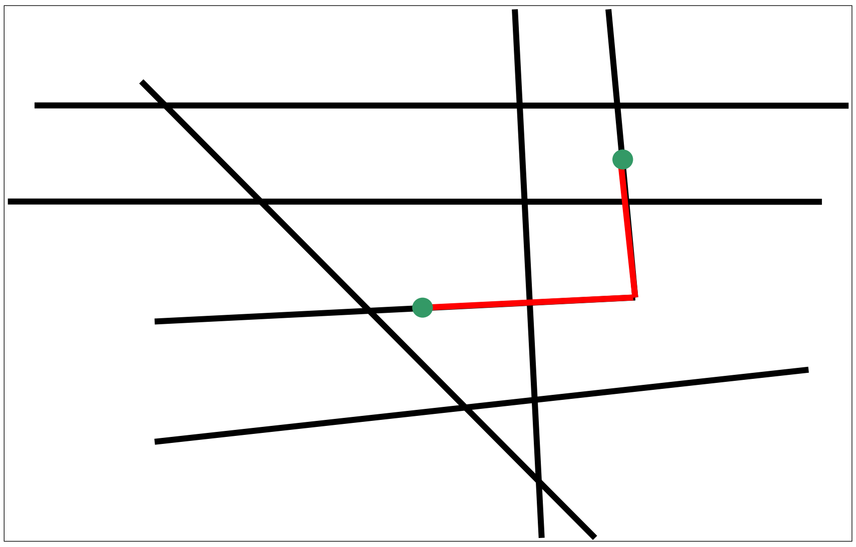

Figure 2: Line location with offsets

3.2 Point locations

A point location is a zero-dimensional element in a map that specifies a geometric location. At least one coordinate pair specifies the point location. The following sections outline different point location types when seen in combination with a (road) network and their real-world examples. The types differ in how the coordinate pair is related to the (road) network.

3.2.1. Geo-Coordinate

A geo-coordinate pair is a position in a map defined by its longitude and latitude coordinate values. This type of point location may or may not be bound to the network and can be everywhere on the surface. Figure 3 shows an example of such a point location (red point).

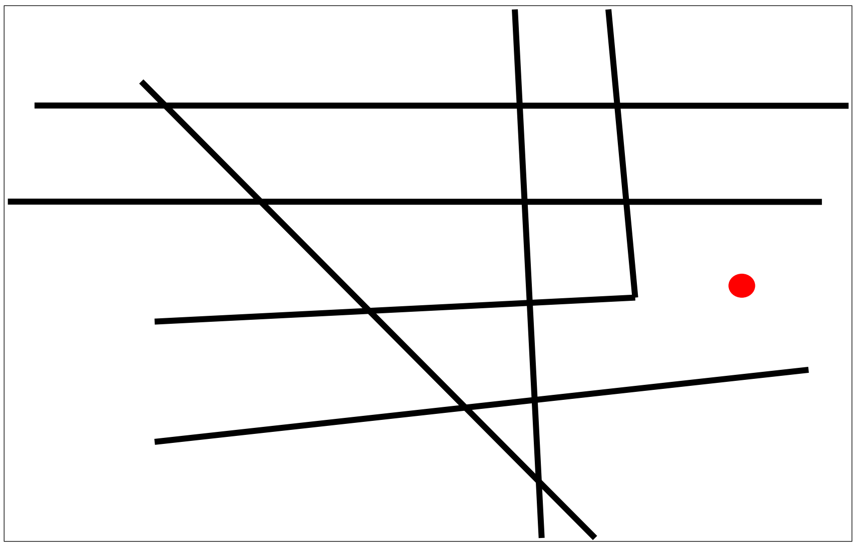

Figure 3: Example - Geo-coordinate

Real world examples for a geo-coordinate as a point location are all coordinate pairs on the surface. This is the general type of a point location. All other types can also be expressed by only using the geo-coordinate pair.

3.2.2. Point along line

The next point location type is a location along a line. Such a line is bounded by two nodes. This point location is dependent on the road network and Figure 4 shows such an example (red point). The point can be on the right side of the line, on the left side of the line, on both sides of the line, or directly on the line. Additionally the point may have an orientation to indicate in which direction of the line the information referenced at that point is useful.

Real world examples of this point location type are all points of interests like petrol stations, shopping malls and restaurants and also house numbers and address points. But it can also be used to reference to speed cams or induction loops.

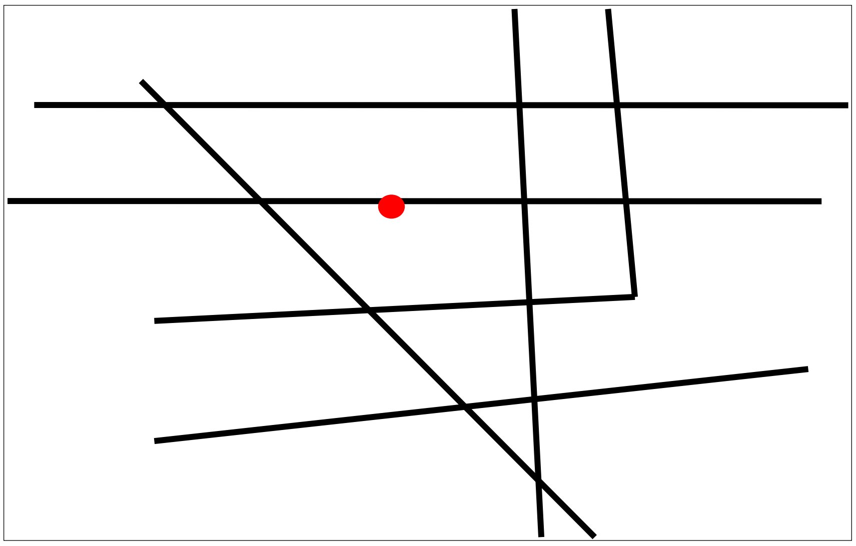

Figure 4: Example - Point along line

3.2.3. POI (point of interest) with access point (on a line)

Another point location type is a simple point location (geo-coordinate) in combination with a so called “access point” which is part of a line. The access point can be on the right side of the line, on the left side of the line, on both sides of the line, or directly on the line. Additionally the point may have an orientation to indicate in which direction of the line the information referenced at that point is useful.

Figure 5 shows an example of this point location type. The red point indicates the point location to be referenced. In combination with this point location there is an access point (green point). The access point is the point within the network in order to access the red point location. An application can use the green point to navigate the user to the desired point location.

Examples for such point locations can be address points but also all point of interests (POI) not being closely related to the road network. One might also think of access to petrol stations or parking garages.

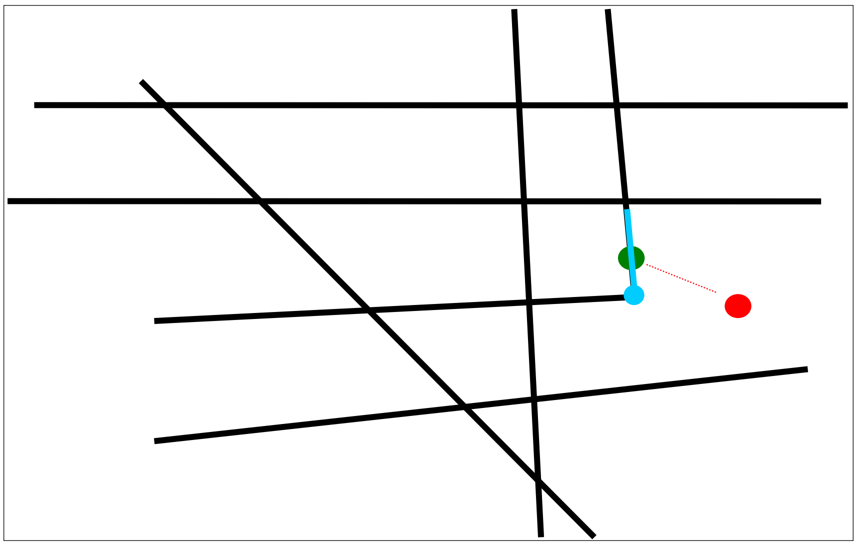

Figure 5: Example - POI with access point

3.3 Area locations

An area location is a two-dimensional part of the surface of the earth which is bounded by a closed curve. An area location may cover parts of the road network but does not necessarily need to. Examples for area locations not covering the network are areas describing woodland, a sea or an agrarian country. In OpenLR™ area locations are defined by their boundary.

3.3.1. Circle

A circle location is given by the position of the center and the radius. The center position is a geo-coordinate pair of longitude and latitude coordinate values that can be everywhere on the surface. The radius is integer-valued and given in meters. Figure 6 shows an example of such an area location.

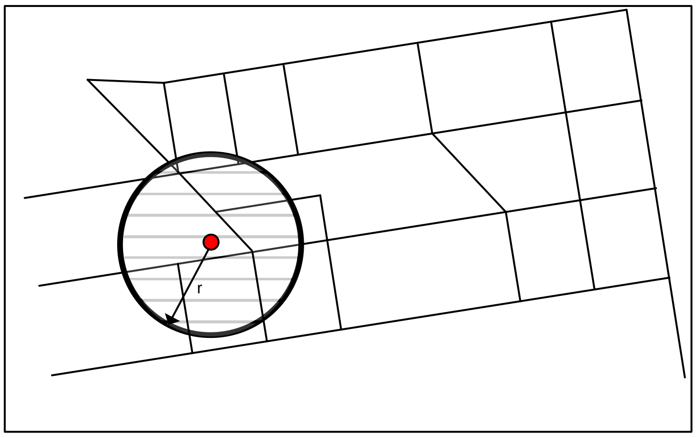

Figure 6: Example – Circle

A real-world example for a circle location is a Wi-Fi hotspot with its signal range.

3.3.2. Rectangle

A rectangle location references to a rectangular shape. It is given by two geo-coordinate pairs which can be everywhere on the surface. The geo-coordinate pairs define the lower left (A) and the upper right (B) corner of the rectangular shape (see Figure 7).

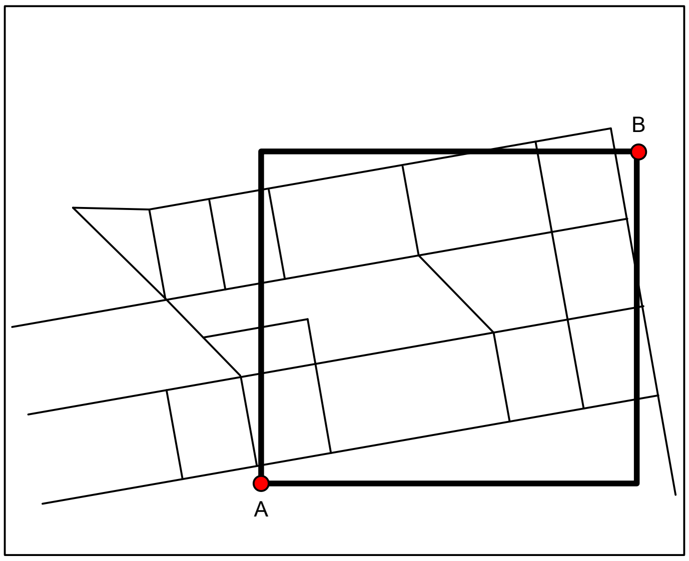

Figure 7: Example - Rectangle location

Real-world examples are weather information or any area where the shape is not exactly specified and the location references shall be light-weighted.

3.3.3. Grid

A grid location is a special instance of a rectangle location. It is given by a base rectangular shape as described in section 3.3.2. This base rectangle is the lower left cell of the grid and can be multiplied to the North (by defining the number of rows) and to the East (by defining the number of columns). Figure 8 shows an example of such an area location.

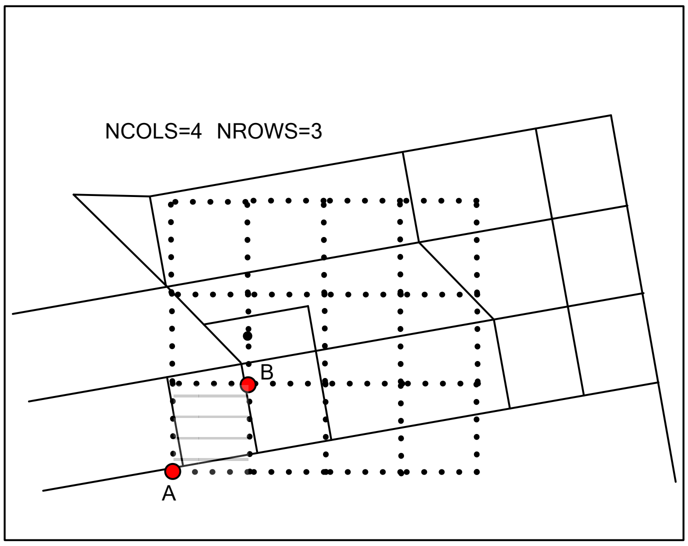

Figure 8: Example - Grid location

Real-world examples are weather reports about e.g. average rainfall for every cell of the grid.

3.3.4. Polygon

A polygon location is a non-intersecting shape defined by a sequence of geo-coordinate pairs. The coordinate pairs can be everywhere on the surface. They define the corners of the underlying geometrical polygon. The boundary of this polygon is constituted by straight lines between every pair of consecutive corners in the sequence, plus the straight line between the last and the first corner.

Figure 9 shows an example of such an area location, defined by the interior of the polygon.

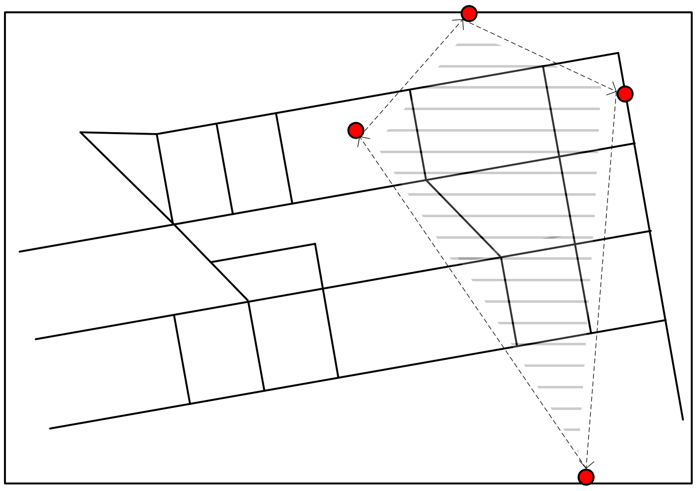

Figure 9: Example - Polygon

Real-world examples for such area locations include low emission zones, areas affected by weather or environmental conditions (bad weather, smog), flood areas, areas that are congested (due to any cause, e.g. traffic overload, public event, or disaster), administrative areas, pedestrian areas, large crowds of people, areas that are blocked for traffic, and areas subject to city toll.

3.3.5. ClosedLine

A closed line location references the area defined by a closed path (i.e. a circuit) in the road network. The boundary always consists of road segments. Figure 10 shows an example of such an area location.

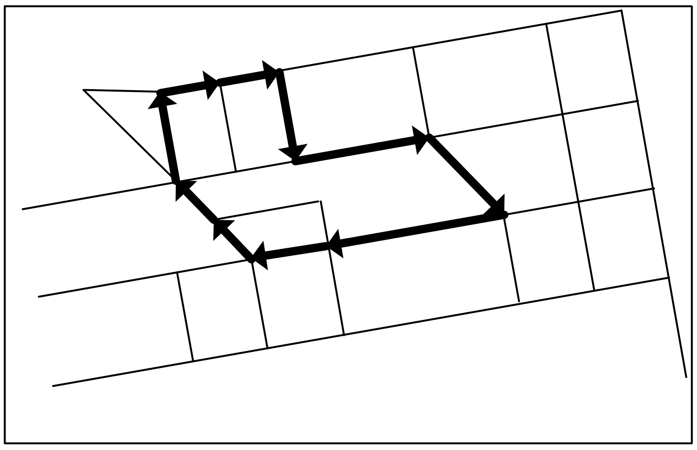

Figure 10: Example - Closed line location

Real-world examples include low emission zones, congestion areas, areas that are blocked for traffic, or even the course of a marathon.

Chapter B
Requirements

4. Requirements

The OpenLR™ standard focuses on creating map agnostic references for line, point and area locations. The digital map used for encoding and decoding and the location itself must fulfill some requirements in order to guarantee acceptable results for the OpenLR™ method. Furthermore the different location types may also define requirements in order to transmit meaningful locations.

4.1 Map requirements

A digital map consists of nodes and lines. Lines represent real world roads or parts of a road. Nodes represent crossings of roads or parts of roads. Nodes are the connecting points between the lines.

The encoder and decoder map might differ but nevertheless the OpenLR™ standard provides a method to reference to the same location represented in both maps.

For being able to generate a map-independent location reference and also being able to resolve locations properly a map should contain information about the following data:

- Coordinates in WGS84
  - Every node in the network should have coordinates in the WGS84 format.
  - The preferable accuracy is decamicrodegrees for each value.
- Length in meter
  - Every line should have a length value in meters indicating its real dimension along the geometry.
- Geometry
  - Every line should know about its real geometry in the real world.
  - Lines shall not be abstracted by the airline.
- Functional road class (FRC)
  - Every line in the network should have a functional road class value indicating its importance in the network.
- Form of way (FOW)
  - Every line in the network should have a form of way value indicating its physical properties.

If the coordinates are not in the WGS84 format they shall be transformed into WGS84. The same applies if length values are not available in meters.

Different digital maps may have different classifications for the functional road class and the form of way values. It may also happen there exist more or less values. Therefore OpenLR™ defines its own ranges for FRC and FOW. There needs to be a proper mapping between the FRC values in the digital map and the FRC values as defined in the logical data format (see section 5.2.2). The same applies for the FOW values (see section 5.2.3). If no such functional road class values or form of way values are provided by the digital map data then these may be derived from other information available in the digital map.

Note:
If a digital map provides less information than required the encoding and decoding processes do generally still succeed but the error rate may increase considerably. At least two out of the following three information types should be available in the map data: geometry (for the calculation of bearing values [angle to the true North]), functional road class, form of way. The order geometry, FRC and FOW also indicates the importance of the information for a location reference point to be unique.

4.2 Location properties

The location types may also define additional requirements in order to deal with a meaningful location. Locations which do not fulfill the location type requirements cannot be encoded with OpenLR™. The OpenLR™ encoder should check each location if the requirements are met and should report on any error.

4.2.1. Line location

Line locations must fulfill the following requirements:

- A line location must be connected.
  - Two subsequent lines in the location must also be connected and adjacent in the underlying network
  - If a driving direction is available then the location shall be traversable from its start to its end
- A line location is represented by an ordered list of line elements.
  - The list of line elements needs to be ordered from the start of a location to the end of a location
- Offsets are always indicated by positive values and the total length of the offsets shall not exceed the length of the complete location.

4.2.2. Point locations

4.2.2.1 Geo-Coordinate

Geo-Coordinate locations do not have further requirements.

4.2.2.2 Point along line

Point along line locations must fulfill the following requirements:

- The reference line is a single line in the road network.
- The offset value defining the position of the location on the line shall not exceed the length of the line.

4.2.2.3 POI with access point

POI with access point locations must fulfill the following requirements:

- The reference line is a single line in the road network.
- The offset value defining the position of the location on the line shall not exceed the length of the line.
- The distance between the start node of the reference line and the position of the POI shall not exceed the maximum distance between two location reference points as defined in Rule-1 in section 6.

4.2.3. Area locations

Area locations should focus on simple (geometric) shapes. Mathematical operations like union or intersection of shapes are not supported and need to be implemented by the application using OpenLR™ area locations.

4.2.3.1 Circle

Circle locations must fulfill the following requirements:

- The radius value must be positive.

4.2.3.2 Rectangle

Rectangle locations must fulfill the following requirements:

- The lower left coordinate and the upper right coordinate must differ in both components (longitude and latitude value).
- The lower left coordinate is southwestern of the upper right coordinate.

4.2.3.3 Grid

Grid locations must fulfill the following requirements:

- The lower left coordinate and the upper right coordinate must differ in both components (longitude and latitude value).
- The lower left coordinate is southwestern of the upper right coordinate.
- The numbers of rows and columns of a grid location must be greater than 1.
- The number of cells forming a grid is exactly the product of the number of rows and the number of columns (number of cells = number of rows * number of columns).

4.2.3.4 Polygon

Polygon locations must fulfill the following requirements:

- The boundary of an area must not cross itself.
- The sequence of geo coordinates of a polygon location must be ordered.

4.2.3.5 ClosedLine

ClosedLine locations must fulfill the following requirements:

- The lines must be connected.
    - Two subsequent lines in the location must also be connected and adjacent in the underlying network
    - If a driving direction is available then the location shall be traversable from its start to its end
- The lines must form a circuit.
    - The last line is a predecessor of the start line and both lines are connected.

Chapter C

Logical data format

5. OpenLR™ - Logical Data format specification

A location reference is a description of a designated part of a digital map or a sequence of geographical positions. OpenLR™ uses for this description the model of “location reference points” (LRPs). Each LRP is a combination of building blocks. The composition of an LRP depends on the location type. The building blocks are defined in the following sections. They provide a logical view onto the data required to describe a location uniquely. This logical interpretation can then result in different physical formats.

A location reference point determines a position in a digital map or a line in a digital map if further line or path attributes are provided. Besides the position or line information additional data may be used to further specify the character of a location.

5.1 Data types

The logical data format makes use of the following data types:

Data type	Description
float	Single precision floating-point numbering format
integer	Natural number including 0 together with the negatives of the non-zero natural numbers

Table 4: Data types (logical data format)

5.2 Building blocks

This section describes the building blocks which are used to compose a location reference point.

5.2.1. Coordinate pair

Coordinate pair stands for a pair of WGS84 longitude (lon) and latitude (lat) values. This coordinate pair specifies a geometric point in a digital map. The lon and lat values are stored in decamicrodegree resolution (10^{-5} , five decimals).

Abbreviation: COORD Type: (float, float)

5.2.2. Functional Road Class

The functional road class (FRC) of a line is a road classification based on the importance of the road represented by the line. The possible values of the FRC attribute are shown in Table 5, covering the range of navigable roads from highest to lowest importance. If there are fewer or more FRC values defined in the encoder map, or decoder map respectively, than these 8 values then a proper mapping needs to be done or less important classes need to be ignored.

FRC
FRC 0 – Main road
FRC 1 – First class road
FRC 2 – Second class road
FRC 3 – Third class road
FRC 4 – Fourth class road
FRC 5 – Fifth class road
FRC 6 – Sixth class road
FRC 7 – Other class road

Table 5: Logical format: Functional road class

Abbreviation: FRC Type: integer

5.2.3. Form of way

The form of way (FOW) describes the physical road type of a line. The possible values of the FOW attribute are shown in Table 6.

FOW	Description
UNDEFINED	The physical road type is unknown.
MOTORWAY	A Motorway is defined as a road permitted for motorized vehicles only in combination with a prescribed minimum speed. It has two or more physically separated carriageways and no single level-crossings.
MULTIPLE_CARRIAGEWAY	A multiple carriageway is defined as a road with physically separated carriageways regardless of the number of lanes. If a road is also a motorway, it should be coded as such and not as a multiple carriageway.
SINGLE_CARRIAGEWAY	All roads without separate carriageways are considered as roads with a single carriageway.
ROUNDABOUT	A Roundabout is a road which forms a ring on which traffic traveling in only one direction is allowed.
TRAFFICSQUARE	A Traffic Square is an open area (partly) enclosed by roads which is used for non-traffic purposes and which is not a Roundabout.
SLIPROAD	A Slip Road is a road especially designed to enter or leave a line.
OTHER	The physical road type is known but does not fit into one of the other categories.

Table 6: Logical Format: Form of way

<u>Abbreviation</u>: FOW   <u>Type</u>: integer

5.2.4. Bearing

The bearing (BEAR) describes the angle between the true North and a line which is defined by the coordinate of the LR-point and a coordinate which is BEARDIST along the line defined by the LR-point attributes. If the line length is less than BEARDIST then the opposite point of the line is used (regardless of BEARDIST). The bearing is measured in degrees and always positive (measuring clockwise from North). The parameter BEARDIST is defined in Table 7.

<u>Abbreviation</u>: BEAR   <u>Type</u>: integer

Abbreviation	Description	Value	Unit
BEARDIST	distance between two coordinates which form a line for the calculation of the bearing value	20	meters

Table 7: Logical format: Parameter BEARDIST

Figure 11 shows how the second point for the bearing calculation is determined. The figure shows a line from A to B which is longer than BEARDIST. The shaded part of this line is exactly BEARDIST meters long so that the point marked with B’ is BEARDIST meters away from A when we walk along the line from A to B. The straight line from A to B’ is now considered for the calculation of the bearing value. Note, this is different to the angle we would calculate if we always took the opposite node of the line (in this case, this would be B).

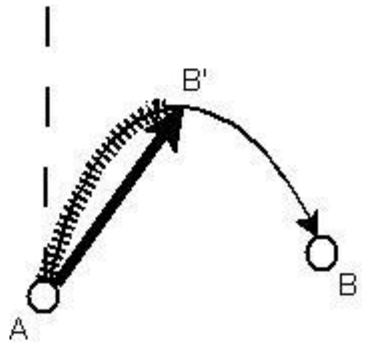

Figure 11: Logical format: Bearing point

Figure 12 shows two examples of the bearing value calculation. There are two lines, one from A to B and one from A to C. For both lines the arcs indicate the angles to the North.

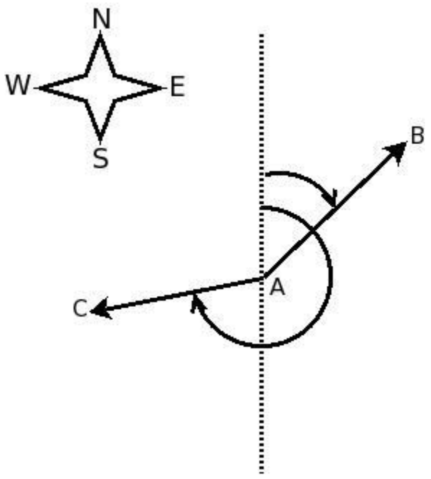

Figure 12: Logical format: Bearing

5.2.5. Distance to next LR-point

The distance to next point (DNP) field describes the distance to the next LRP in the topological connection of the LRPs. The distance is measured in meters and is calculated along the location reference path between two subsequent LR-points. The last LRP has the distance value 0.

Abbreviation: DNP   Type: integer

Figure 13 shows an example of the distance calculation and assignment. The three LRPs are in a sequence from A over B to C. Therefore the distance between A and B along the location reference path is assigned to A. The LRP B holds the distance between B and C and the LR-point C has a distance value of 0.

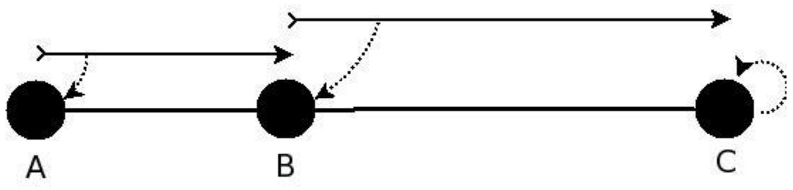

Figure 13: Logical format: Distance to next point

5.2.6. Lowest FRC to next LR-point

The lowest FRC to the next point (LFRCNP) is the lowest FRC value which appears in the location reference path between two consecutive LR-points. This information could be used to limit the number of road classes which need to be scanned during the decoding. The highest FRC value is 0 and the lowest possible FRC is valued with 7.

Abbreviation:	LFRCNP	Type:	integer

5.2.7. Radius

The radius describes the distance from a center point to a circle's boundary. It is given as an integer value in a resolution of an accuracy of 1 meter.

Abbreviation:	RAD	Type:	integer

5.2.8. Number of columns / rows

The number of columns (NCOLS) defines the total number of base cells of a grid in eastern direction. The number of rows (NROWS) defines the total number of base cells of a grid in northern direction. Both are given as integer values.

Abbreviation:	NCOLS	Type:	integer
Abbreviation:	NROWS	Type:	integer

5.2.9. Offsets

Offsets are needed to locate start and end of a line location and the position of a point location precisely. Offset values shorten the location reference path down to the desired location and are measured in meters.

Line locations in OpenLR™ rely on the condition that the path defined by the location reference points (location reference path) covers the location completely. The location reference path may even be longer than the location path. Point locations identified by a network line might also use offsets to determine the concrete position of the desired point on that line.

OpenLR™ defines two offset values. A positive offset (POFF) is used to locate the precise start of a location. The POFF defines the distance between the start of the location reference path and the start of the location. The negative offset (NOFF) is used to locate the precise end of the location and it defines the distance between the end of the location and the end of the location reference path.

5.2.9.1 Positive offset

The positive offset (POFF) is the difference of the start point of the location and the start point of the desired location along the location reference path. The value is measured in meters. Figure 14 shows an example for the calculation of the positive and negative offset. The lines indicate the location reference path and the hatching indicates the desired location.

Abbreviation:	POFF	Type:	integer

5.2.9.2 Negative offset

The negative offset (NOFF) is the difference of the end point of the desired location and the end point of the location reference path. The value is measured in meters. Figure 14 shows an example for the calculation of the positive and negative offset. The lines are indicating the location reference path and the hatching indicates the desired location.

Abbreviation:	NOFF	Type:	integer

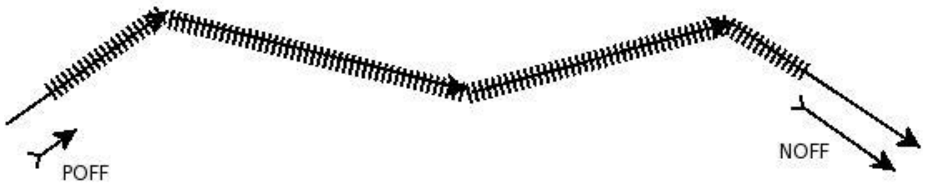

Figure 14: Logical format: Positive and negative offset

5.3 Additional data

Additional data specifies the character of a location. It is not required to find back a location in the decoding process.

5.3.1. Side of road

The side of road information (SOR) describes the relationship between the point of interest and a referenced line. The point can be on the right side of the line, on the left side of the line, on both sides of the line or directly on the line.

SOR	Description
0	Point is directly on (or above) the road, or determination of right/left side is not applicable (default)
1	Point is on right side of the road
2	Point is on the left side of the road
3	Point is on both sides of the road

Abbreviation: SOR   Type: integer

5.3.2. Orientation

The orientation information (ORI) describes the relationship between the point of interest and the direction of a referenced line. The point may be directed in the same direction as the line, against that direction, both directions, or the direction of the point might be unknown.

ORI	Description
0	Point has no sense of orientation, or determination of orientation is not applicable (default)
1	Point has orientation from first LRP towards second LRP
2	Point has orientation from second LRP towards first LRP
3	Point has orientation in both directions

Table 8: Orientation of a point location

Abbreviation: ORI   Type: integer

5.4 Location Reference Point

The basis of a location reference is a sequence of location reference points (LRPs). Such an LRP is composed of the building blocks defined in section 5.2. A single LRP may be bound to the road network. In such a case all values of the LRP refer to a node or line within the road network. The coordinates refer to a node of a line or a point on a line and the additional attributes refer to attributes of a line. The relationship between the attributes and an LRP are shown in section 5.4.2.

If an LRP is not bound to the road network then the LRP consists of at least one coordinate pair (see 5.2.1).

5.4.1. Topological connection of LR-points

The location reference points must be stored in a topological order so that a point A will directly be followed by a point B if B comes after A in the location reference path and there is no other LR-point in between. This topological order defines a “next point”-relationship of successive LR-points. The last point in this order has no next point in this relationship.

Figure 15 shows an example of this relationship. The LR-points are indicated by A, B and C and the black lines and arrows indicate the order of the points from A to C in the location reference path. In this example the LR-point A has B as next point, B has C as next point and C has no next point. The arrows may indicate a path in the road network or a direct line connection between two coordinates.

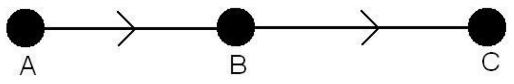

Figure 15: Topological connection of LRPs

5.4.2. Relationship between attributes and LR-points

If an LRP is bound to the road network all attributes are linked to this LR-point. For all LR-points (except the last LR-point) the attributes describe an outgoing line of the node at the LR-point coordinate which belongs to the location. The attributes of the last LR-point direct to an incoming line of the node at the LR-point coordinate which belongs to the location.

Figure 16 shows an example of the relationship between an LR-point and the attributes. The lines indicate the location reference path and the nodes A, B and C are the LR-points. Note that there is also a line whose start and end node is not an LR-point (the third line in the sequence). This line does not need to be referenced because it is covered by the shortest path between the LR-points B and C.

The LR-points A and B direct to an outgoing line and the last LR-point C directs to an incoming line.

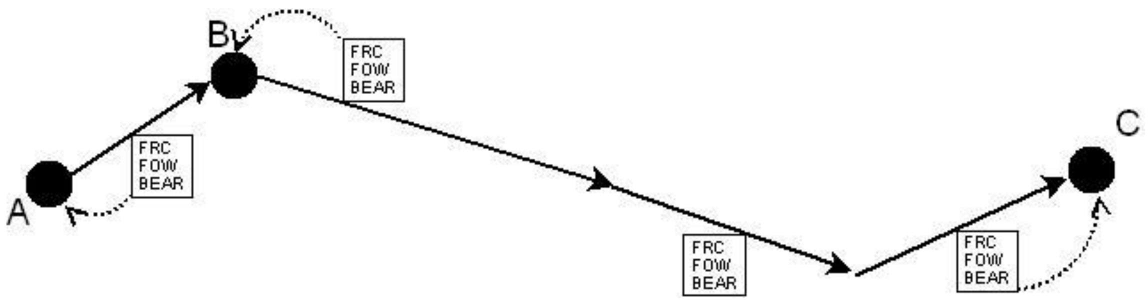

Figure 16: Relationship between attributes and LR-points

5.4.3. LRP composition per location type

The composition of an LRP depends on the location type. The layout of an LRP may also vary for a single location type. The following specifications just collect the building blocks (on a logical level) but do not define the physical layout.

5.4.3.1 Line

The location type Line uses three different types of location reference points. The “First LRP” describes the start line of the line location. The “Last LRP” refers to the end line of the location. Both types are mandatory for a line location.

An "Intermediate LRP" is added to the list of location reference points if the encoder decides to mark additional lines along the location path. These additional LRPs act as a guide for the decoder to reconstruct the location.

Besides the reference information to a specific line in the network the First LRP and any Intermediate LRP contain information to reduce the calculation effort on decoder side and to verify the correctness of the calculated path.

For further precision of the location the location reference may contain information about positive and negative offsets.

Building blocks used:

- First LRP
    - COORD
    - BEAR
    - FRC
    - FOW
    - LFRCNP
    - DNP
- Intermediate LRP
    - COORD
    - BEAR
    - FRC
    - FOW
    - LFRCNP
    - DNP
- Last LRP
    - COORD
    - BEAR
    - FRC
    - FOW

Additional building blocks (optional):

- POFF
- NOFF

5.4.3.2 Geo-Coordinate

The Geo-Coordinate location type makes use of a single LRP type which contains a single coordinate pair.

Building blocks used:

- COORD

5.4.3.3 Point along line

The Point along line location type describes the reference line using a "First LRP" and a "Last LRP" type. The "First LRP" describes the start line of the reference path. The "Last LRP" refers to the end of the reference path. Both types are mandatory for a point along line location.

Besides the reference information to a specific line in the network the First LRP contains information to reduce the calculation effort on decoder side and to verify the correctness of the calculated path. The specification of a positive offset defines the desired point along the location path that shall be encoded. If the offset is missing the point is implicitly defined by the first LRP.

Optionally the side of the road and or the orientation with respect to the direction from start to end can be added to the location reference.

Building blocks used:

- First LRP
    - COORD

  - BEAR
  - FRC
  - FOW
  - LFRCNP
  - DNP
- Last LRP
  - COORD
  - BEAR
  - FRC
  - FOW

Additional building blocks (optional):
- POFF
- SOR
- ORI

5.4.3.4 POI with access point

The POI with access point location type describes the reference line using a "First LRP" and a "Last LRP" type. The "First LRP" describes the start line of the line location. The "Last LRP" refers to the end line of the location. The POI is referenced by a coordinate pair. Both LRP types and the coordinate pair are mandatory for this location type.

Besides the reference information to a specific line in the network the First LRP contains information to reduce the calculation effort on decoder side and to verify the correctness of the calculated path. The specification of a positive offset defines the desired access point along the location path that shall be encoded. If the offset is missing the access point is implicitly defined by the first LRP.

Optionally the side of the road and or the orientation with respect to the direction from start to end can be added to the location reference.

Building blocks used:
- First LRP
  - COORD
  - BEAR
  - FRC
  - FOW
  - LFRCNP
  - DNP
- Last LRP
  - COORD
  - BEAR
  - FRC
  - FOW
- COORD

Additional building blocks (optional):
- POFF
- SOR
- ORI

5.4.3.5 Circle

The circle location type makes use of a coordinate pair and a radius value.

Building blocks used:
- COORD
- RAD

5.4.3.6 Rectangle

The rectangle location type uses two coordinate pairs to describe the lower left and upper right coordinate of the rectangle.

Building blocks used:

- COORD

5.4.3.7 Grid

The grid location type uses two coordinate pairs to describe the lower left and upper right coordinate of the basis rectangle. The number of rows and columns specify how many cells form this grid in total (number of cells = number of rows * number of columns).

Building blocks used:

- COORD
- NCOLS
- NROWS

5.4.3.8 Polygon

The polygon location type makes use of coordinate pairs. The minimum number of coordinate pairs is three and there exists no maximum number.

Building blocks used:

- COORD

5.4.3.9 Closed line

The closed line location defines three different types of location reference points. The “First LRP” describes the start line of the closed line location. The “Last ClosedLine LRP” refers to the end line of the location whereby the end node of the end line equals the start node of the start line. Both types are mandatory for a closed line location.

An “Intermediate LRP” is added to the list of location reference points if the encoder decides to mark additional lines along the location path. These additional LRPs act as a guide for the decoder to reconstruct the location.

Besides the reference information to a specific line in the network the First LRP and any Intermediate LRP contain information to reduce the calculation effort on decoder side and to verify the correctness of the calculated path.

Building blocks used:

- First LRP
  - COORD
  - BEAR
  - FRC
  - FOW
  - LFRCNP
  - DNP
- Intermediate LRP
  - COORD
  - BEAR
  - FRC
  - FOW
  - LFRCNP
  - DNP
- Last ClosedLine LRP
  - BEAR
  - FRC

- FOW

6. Data format rules

The OpenLR™ rules describe additional regulations for OpenLR™ location references, supplementary to the general "shortest-path" rule for sub-dividing the location reference path. These rules are used to simplify the encoding and decoding process and to increase the accuracy of the results.

Rule – 1 *The maximum distance between two location reference points shall not exceed 15km. The distance is measured along the location reference path. If this condition is not fulfilled for a location reference then a sufficient number of additional LR-points shall be inserted.*

The maximum distance between two consecutive location reference points is restricted in order to speed up shortest-path computation because several short routes can be computed quicker than one large route if the routing algorithm has to take the whole network into account. The restriction also provides the opportunity to form a compact physical format with an acceptable accuracy.

Rule – 2 *All lengths are integer values. If there are float values available then they shall be rounded to an integer representation.*

Different maps might store the length values in different formats and also with different precision. The uniform basis for all is the usage of integer values.

Rule – 3 *Two LR-points are mandatory for all locations being bound to road network.*

A location reference which is bound to the road network must always have at least two location reference points indicating the start and the end of the location. In addition, further intermediate location reference points are to be included if conditions of shortest-path or Rule 1 are not met. If the encoder detects critical situations where the decoder (on a different map) might get into trouble, the location reference might be enhanced with additional (optional) intermediate LR-points. The number of intermediate LR-points is not limited for the line location and closed line location.

Rule – 4 *The coordinates of the LR-points should preferably be chosen on valid network nodes.*

Valid network nodes shall represent junctions in the real world because it is expected that these junctions can be found in different maps with a higher probability than positions somewhere on a line. Additional nodes other than junctions (invalid nodes) shall be avoided if these can be easily skipped during a route search. At these invalid nodes it is not possible to deviate from a route.

Nodes having only one incoming and one outgoing line (as directed connectivity) shall be invalid and therefore avoided since these nodes are not related to junctions (see Figure 17, node in the middle). These nodes can be stepped over during a route search. Nodes which have two incoming and two outgoing lines and there exist only two adjacent nodes shall also be invalid, unless a U-turn is possible at that node (see Figure 18, node in the middle).

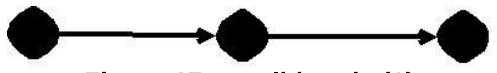

Figure 17: Invalid node (1)

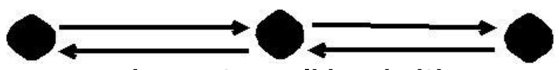

Figure 18: Invalid node (2)

If such node is selected for an LR-point then this LR-point should be shifted along the location reference path in order to find a suitable node. This can be done since a route calculation will step over such invalid nodes without leaving the desired path.

If the encoder cannot determine a valid node at all, then exceptionally an invalid node may be chosen for an LR-Point. Examples are lines which are longer than 15 km (see Rule – 1).

In order to fulfill Rule – 4 it is recommended to check the start node and end node of the location path. If the start or the end of a location is placed on invalid nodes then the encoder might expand the location uniquely and should find a suitable node outside of the location. This expansion must never go into the location because this will shorten the location's spatial extent.

Rule – 5 *Offsets must refer to the location reference path defined by the first two (the last two) location reference points.*

Offsets may be placed only between the first two location reference points (positive offset) and between the last two location reference points (negative offset). Otherwise the first (or last) location reference point is useless as it brings no additional information to describe the location or the offsets. Figure 19 shows where the offset values may be placed along a location and it also indicates that the maximum offset value is 15000 meters (see Rule – 1).

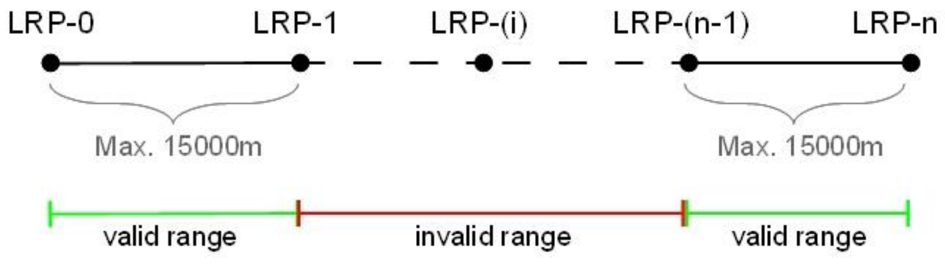

Figure 19: Offset placement and restrictions

Rule – 6 *Area locations should focus on simple (geometric) shapes. The boundary of an area must not cross itself. Mathematical operations like union or intersection of shapes are not supported and need to be implemented by the application using OpenLR™ area locations.*

Area locations in OpenLR™ must be simple. In addition the boundary of an area location must not cross itself. For more complex area locations the encoding and decoding side should agree on common mathematical operations like union or intersection. But these operations are not part of the OpenLR™ specification.

6.1 Overview of the data format rules

The following Table 9 summarizes the data format rules.

Rule	Description	Value
Rule 1	max distance between two consecutive LR-points	15000 m
Rule 2	road length values	treated as integer values
Rule 3	number of LR-points	at least two LR-points (for line locations)
Rule 4	invalid nodes	LR-points shall preferably be placed on valid network nodes
Rule 5	Offsets	offsets must refer to the first two (last two) LR-points
Rule 6	Area locations focus on simple geometric shapes. The boundary of an area must not cross itself.	

Table 9: Data format rules overview

Chapter D

Physical data format (Binary)

7. OpenLR™ - Binary representation

This physical data format describes a byte-oriented stream format of the OpenLR™ data format. It bases on the components described in the logical data format in section 5.

This section defines the binary format version 3.

7.1 Data types

The physical data format uses the following data types. Table 10 gives an overview of all available data types and specifies the name, the type and the designated size of each. In the following sections the data type names are used to indicate the size and type for each data component.

Data type name	Type	Size	Range
Boolean	flag with true=1, false=0	1 bit	0 – 1
uByte	unsigned integer	1 byte	0 – 255
uShort	unsigned integer	2 bytes	0 – 65535
uSmallInt	unsigned integer	3 bytes	0 – 16777215
uInteger	unsigned integer	4 bytes	0 – 4294967295
sShort	signed integer	2 bytes	-32768 – 32767
sSmallInt	signed integer	3 bytes	-8388608 – 8388607
BitField[n]	array of n bits	n bits	variable size

Table 10: Physical format: Data types (Binary)

Negative integer values are stored in the two's complement format.

7.2 Coordinates (COORD)

Each point in a map consists of a coordinate pair "longitude" (lon) and "latitude" (lat) represented in WGS84 coordinates. The lon and lat values are stored in a decamicrodegrees resolution (10^{-5}, five decimals). The coordinate values are transmitted as integer values.

The physical format makes use of an absolute and a relative coordinate format. The absolute format represents the designated values of the geographical position and the relative value is the offset of the coordinates relative to the preceding coordinate.

7.2.1. Absolute format

The absolute format describes geographical positions in a 24-bit resolution. Table 11 shows the data type used for the absolute format.

Data type	Value	Description
sSmallInt	-8388608 – +8388607	24 bit representation

Table 11: Physical format: Coordinate format (absolute) (Binary)

The values are generated using Equation 1 which calculates a 24-bit integer representation. The resolution parameter is set to 24. This translation leads to a coordinate resolution ("error") of about 2.4 meter at most. The backward translation is described in Equation 2. Both equations make use of the signum function which is -1 for negative values, 1 for positive values and 0 otherwise.

$$\text{int} = \left(\operatorname{sgn}(\deg) * 0.5 + \frac{\deg * 2^{Resolution}}{360°} \right)$$

Equation 1: Transformation from decimal coordinates into integer values (Binary)

$$\deg = \left(\frac{(\text{int} - \operatorname{sgn}(\text{int}) * 0.5) * 360°}{2^{\text{Re}\,solution}} \right)$$

Equation 2: Transformation from integer values into decimal coordinates (Binary)

7.2.2. Relative format

The relative format is used to describe differences between two consecutive coordinates. The difference is calculated for each value (lon/lat) separately as shown in Equation 3. The current and previous values represent the latitude (longitude) value in degrees. The difference between these two values is multiplied with 100000 (10^5) in order to resolve an integer value.

$$relative = round(100000 * (currentPo\text{int} - previousPo\text{int}))$$

Equation 3: Relative coordinates calculation (Binary)

Table 12 shows the data type for relative offsets.

Data type	Value	Description
sShort	-32768 – +32767	2 bytes relative coordinates

Table 12: Physical format: Coordinate format (relative) (Binary)

Table 13 shows the maximum distances which are possible to encode within that resolution for a location in the Netherlands. The ranges for relative coordinates are calculated for a given coordinate in the Netherlands at lon = 5° and lat = 52°.

byte	latitude		longitude	
	lower bound	upper bound	lower bound	upper bound
2	-36459 m	36460 m	-22504 m	22504 m

Table 13: Physical format: Longitude/Latitude ranges for relative coordinates (Example)

Important Note
The interval lengths for the latitude are fixed and do not change if the longitude values change. However, the interval lengths for the longitude depend on the latitude values and they change if the latitude values change. Figure 20 shows a chart of the maximum ranges of the longitude values depending on the latitude. The difference between the longitude values is exactly 0.32767°, the maximum value being covered by the 16 bit representation of relative coordinates.

The graph indicates that Rule – 1 of the data format rules is applicable for all latitude values in the range -65.70° < lat < 65.70°. The maximum distance of the longitude values is here greater than the 15km defined in Rule – 1.

For latitude values lower than -65.70° or greater than 65.70° the distance between two longitude values which differ by 0.32767° is less than 15 km. In these areas Rule – 1 needs to be adapted to a lower maximum range. If the encoder still uses the value of 15 km then the binary format cannot be used any further. The available range for relative coordinates will be exceeded.

Therefore it is recommended for such cases to use a lower maximum distance value. Lower values also fulfill the requirements of Rule – 1 and do not break the validity of the OpenLR™ method but the resulting location reference might contain more LR-points. Other physical data formats are not affected by this issue.

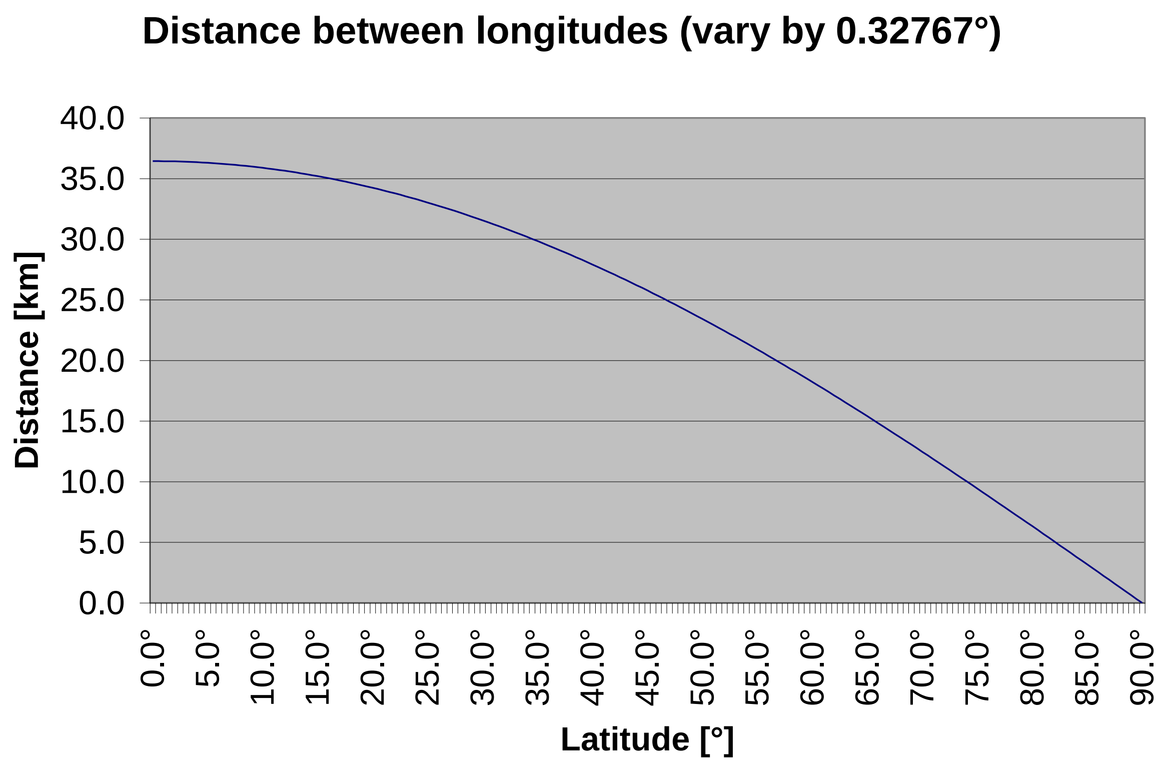

Figure 20: Distances between longitudes

Equation 4 shows the spherical law of cosines formula. This formula can be used to calculate the distance between two coordinates. The latitude and longitude values are expressed in radians and need to be converted from degrees to radians first.

$$d = \arccos(\sin(lat1) * \sin(lat2) + \cos(lat1) * \cos(lat2) * \cos(lon2 - lon1)) * 6371$$

Equation 4: Spherical law of cosines

Equation 4 can be simplified for calculating the maximum range of longitudes. The resulting distance can then be used in Rule – 1 as a new maximum distance between two subsequent LR-points.

The term $\cos(lon2 - lon1)$ can be simplified to the constant value $\cos(rad(0.32767°))$. This expresses the maximum difference in degree which is covered by the 16 bit relative offset representation. The equation can further be simplified by determining the largest latitude value (lowest latitude value for negative numbers) along the location. This ensures that the maximum distance value is valid for the whole location.

The resulting formula is shown in Equation 5: Calculation of the maximum longitude range using "mlat" as the largest (lowest) latitude value along the location.

$$\max_{Distance} = \arccos(\sin^2(rad(mlat)) + \cos^2(rad(mlat)) * \cos(rad(0.32767°))) * 6371$$

Equation 5: Calculation of the maximum longitude range

7.3 Attribute values

The binary format of the attributes follows in this section.

7.3.1. Functional Road Class (FRC)

The functional road class (FRC) can hold eight different values as described in the logical format. These eight values are represented by 3 bits and the mapping is shown in Table 14.

Data type	Value (integer)	Value (binary)	Description
BitField[3]	0	000	FRC 0 – Main road, highest importance
	1	001	FRC 1 – First class road
	2	010	FRC 2 – Second class road
	3	011	FRC 3 – Third class road
	4	100	FRC 4 – Fourth class road
	5	101	FRC 5 – Fifth class road
	6	110	FRC 6 – Sixth class road
	7	111	FRC 7 – Other class road, lowest importance

Table 14: Physical format: Functional road class (Binary)

7.3.2. Form of way (FOW)

The form of way (FOW) can hold eight different values as described in the logical format. These eight values are represented by 3 bits and the mapping is shown in Table 15.

Data type	Value (integer)	Value (binary)	Description
BitField[3]	0	000	UNDEFINED
	1	001	MOTORWAY
	2	010	MULTIPLE_CARRIAGEWAY
	3	011	SINGLE_CARRIAGEWAY
	4	100	ROUNDABOUT
	5	101	TRAFFICSQUARE
	6	110	SLIPROAD
	7	111	OTHER

Table 15: Physical format: Form of way (Binary)

7.3.3. Bearing (BEAR)

The bearing describes the angle between the true North and the road as described in the logical format. The physical data format defines 32 sectors whereby each sector covers 11.25° of the circle. These 32 sectors are represented by 5 bits. Table 16 shows the data type for the bearing attribute and Table 17 shows the mapping from the sectors to the concrete value.

Data type	Value	Description
BitField[5]	0-31	Number of the sector in which the angle between the true North and the line specified in the logical data format is located; the full circle is divided into 32 sectors each covering an angle of 11.25°.

Table 16: Physical format: Bearing (Binary)

Value	Sector	Value	Sector
0	000.00° <= x < 011.25°	16	180.00° <= x < 191.25°
1	011.25° <= x < 022.50°	17	191.25° <= x < 202.50°
2	022.50° <= x < 033.75°	18	202.50° <= x < 213.75°
3	033.75° <= x < 045.00°	19	213.75° <= x < 225.00°
4	045.00° <= x < 056.25°	20	225.00° <= x < 236.25°
5	056.25° <= x < 067.50°	21	236.25° <= x < 247.50°
6	067.50° <= x < 078.75°	22	247.50° <= x < 258.75°
7	078.75° <= x < 090.00°	23	258.75° <= x < 270.00°
8	090.00° <= x < 101.25°	24	270.00° <= x < 281.25°
9	101.25° <= x < 112.50°	25	281.25° <= x < 292.50°
10	112.50° <= x < 123.75°	26	292.50° <= x < 303.75°

11	123.75° <= x < 135.00°	27	303.75° <= x < 315.00°
12	135.00° <= x < 146.25°	28	315.00° <= x < 326.25°
13	146.25° <= x < 157.50°	29	326.25° <= x < 337.50°
14	157.50° <= x < 168.75°	30	337.50° <= x < 348.75°
15	168.75° <= x < 180.00°	31	348.75°<= x < 360.00°

Table 17: Physical format: Bearing value definition (Binary)

Equation 6 outlines the calculation of the bearing value and Figure 21 provides a graphical overview of the sectors.

$$value = \left\lfloor \frac{angle}{11.25°} \right\rfloor \quad , 0° \le angle < 360°$$

Equation 6: Calculation of the bearing value (Binary)

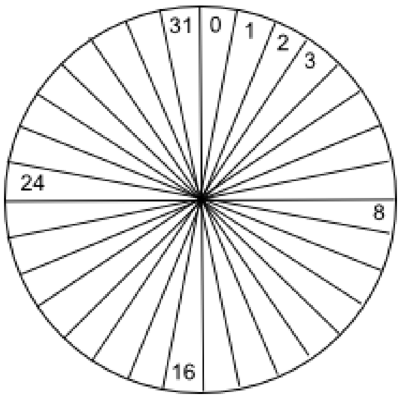

Figure 21: Physical format: Bearing sectors (Binary)

7.3.4. Distance to next LR-point (DNP)

The DNP attribute measures the distance between two consecutive LR-points along the location reference path as described in the logical format.

The physical data format defines an 8-bit representation. Table 18 shows the data type used for DNP. This representation defines 256 intervals and in combination with Rule – 1 of the data format rules (maximum length between two consecutive LR-points is limited by 15000m) each interval has a length of approximately 58.6 meters.

Data type	Value	Description
uByte	0-255	Distance interval according to Equation 7

Table 18: Physical format: Distance to next point (Binary)

Equation 7 shows how the DNP values can be calculated.

$$value = \left\lfloor \frac{length}{58.6m} \right\rfloor$$

Equation 7: Calculation of the DNP value (Binary)

7.3.5. Lowest FRC to next point (LFRCNP)

The lowest FRC to the next point indicates the lowest functional road class used in the location reference path to the next LR-point. This information could be used to limit the number of road classes which need to be scanned during the decoding. See Table 19 for a definition of the data type.

Data type	Value	Description
BitField[3]	0-7	Holds the same values as described in Table 14

Table 19: Physical format: Lowest FRC to next point (Binary)

7.3.6. Orientation information

A point location may have additional information in which direction the information at that point is relevant. Examples for such point locations are speed cams or traffic signs. This additional information is valid for the types “point along line” and “poi with access point”. The orientation information type will be expressed by two bits. The meaning of the allowed values is shown in Table 20.

Data type	Value (binary)	Meaning
BitField[2]	00	Point has no sense of orientation, or determination of orientation is not applicable (default)
	01	Point has orientation from first LRP towards second LRP
	10	Point has orientation from second LRP towards first LRP
	11	Point has orientation in both directions

Table 20: Orientation of a point location

7.3.7. Side of road information

A point location may have additional information on which side of the road the point is located. Examples for such point locations are particular house numbers, speed cams or traffic signs. This additional information is valid for the types “point along line” and “poi with access point”. The side information type will be expressed by two bits. The meaning of the allowed values is shown in Table 21.

Data type	Value (binary)	Meaning
BitField[2]	00	Point is directly on (or above) the road, or determination of right/left side is not applicable (default)
	01	Point is on right side of the road
	10	Point is on the left side of the road
	11	Point is on both sides of the road

Table 21: Side of the road of a point location

7.3.8. Radius

The radius defines the distance between a coordinate (center point) and the border line of a circle. The value is given in meters. The choice of the used data type depends on the concrete value of the radius. It is recommended to choose the data type where the value fits best and no additional and unnecessary bytes are used (see Table 22).

Data type	Value	Description
uByte	0 – 255	Radius (the data type used depends on the concrete value of the radius)
uShort	256 – 65535	
uSmallInt	65536 – 16777215	
uInteger	16777216 – 4294967295	

Table 22: Radius

7.3.9. Number of columns / rows

The number of rows defines the number of cells in a grid in northern direction. The number of columns defines the number of cells in a grid in eastern direction. Both values must be greater than 1 and the maximum number of cells in either direction is 65535 (see Table 23).

Data type	Value	Description
uShort	2 – 65535	Number of columns Number of rows

Table 23: Number of rows / number of columns

7.4 Location Reference header

The Location Reference header contains general information about the reference and is the first information in a binary location reference.

7.4.1. Version (VER)

The version is used to distinguish between several physical and data formats for location references. The version number is represented by 3 bits. The data type is shown in Table 24.

Data type	Value	Description
BitField[3]	0-7	current version number

Table 24: Physical format: Version (Binary)

Note:
The current version of the binary physical data format is 3 so the VER field is constantly set to binary 011.

Note:
The version number of the physical format is independent from the version of the OpenLR™ method. An implementation of a given version of the OpenLR™ method may support multiple versions of the physical format. Likewise, a particular version of the physical format may be used throughout the evolution of OpenLR™ across several versions of the method.

7.4.2. Attribute flag (AF)

The attribute flag indicates whether there are attributes appended to each LR-point or not. The AF value is 0 if no attributes are appended and therefore the location reference only consists of coordinates. Otherwise a value of 1 indicates that attributes are appended to each LR-point. The data type of the AF is shown in Table 25 and Table 26.

Data type	Value	Description
Boolean	0, 1	flag, indicating whether attributes are appended to each LR-point or not

Table 25: Physical format: Attribute flag (Binary)

Value	Description
0	no attributes are appended
1	for each LR-point a set of attributes is appended

Table 26: Physical format: Attribute flag values (Binary)

7.4.3. Area flags (ArF)

The area flags indicate the category of an area location (see Table 27). The flags are only used if the location reference does not correspond to a line or a point location.

Data type	Value (binary)	Description
BitField[2]	00 01 10 11	Circle or no area location Polygon Rectangle or Grid Closed line

Table 27: Physical format: Area flags (Binary)

7.4.4. Point flag (PF)

The point flag indicates whether the location reference describes a point or not. If this flag is set the location describes a point location.

Data type	Value	Description
Boolean	0, 1	flag, indicating whether the location reference describes a point or not

Table 28: Physical format: Point flag (Binary)

Value	Description
0	location reference does not describe a point
1	location reference describes a point

Table 29: Physical format: Point flag values (Binary)

7.5 Offsets

Offsets are used to locate the start and end of a location more precisely than being bound to the nodes in a network. The logical format defines two offsets, one at the start of the location and one at the end of the location. Both offsets operate along the lines of the location and are measured in meters. The offset values are not mandatory. A missing offset value means an offset of 0 meters. Offsets are also only valid for location references which have attributes included.

7.5.1. Offset flags

Offset flags indicate whether the data includes a specific offset information or not. The physical data format deals with two flags corresponding to the two different offset values. The positive offset flag (POffF) and the negative offset flag (NOffF) are described in Table 30 and Table 31.

Data type	Value	Description
Boolean	0, 1	flag, indicating whether the corresponding offset value is included in the data or not

Table 30: Physical format: Offset flag (Binary)

Value	Description
0	location reference data does NOT include the corresponding offset information
1	location reference data includes the corresponding offset information

Table 31: Physical format: Offset flag values (Binary)

7.5.2. Offset values

The value used here is the relation of the offset length to the length of the path between the first two location reference points (last two location reference points for the negative offset). The length between these two LR-points shall be called LRP length.

The relative value (or percentage) will then be equally distributed over the available 256 buckets so that every bucket covers 0.390625% of the LRP length. This method takes the length of a location into account which results in a better accuracy for shorter locations. Even for shorter locations the whole bucket range is used and longer locations as no offset value can be greater than 15000 meter irrespective of the location length.

Table 32 shows the data type used for POFF and NOFF.

Data type	Value	Description
uByte	0-255	offset length interval according to

Table 32: Physical format: Offset (Binary)

Equation 8 shows how the bucket index can be calculated having the offset and location length available. Figure 22 shows some interval lengths (in meters) for selected LRP lengths. It is obvious that the greater the LRP length the less the accuracy of the offset.

$$y = \left\lfloor \frac{256 * offset\_length}{LRP\_length} \right\rfloor$$

Equation 8: Bucket index calculation

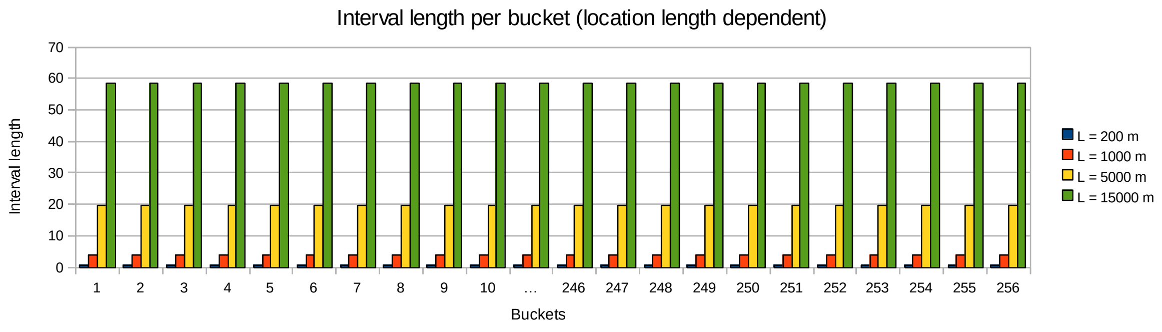

Figure 22: Interval lengths

Recommendation
If the encoder is forced to generate highly accurate offset values, it is recommended to add an additional LR-point. This additional LR-point shall reduce the LRP length value and shall be placed between the first two (last two) LR-points. The path between the first (last) LR-point and the additional LR-point must be at least as long as the offset but shorter than the original distance between the first two (last two) LR-points.

This recommendation might violate Rule – 4 if the additional LR-point cannot be placed on a valid node but the generated location reference shall still be valid. It might even happen that the additional LR-point is placed directly on a line. In such a case the coordinates are taken from a point along the line and the attributes of the LR-point are derived from that line.

The decoder might not find a node for this LR-point so it needs to be ensured that the decoder also implements a strategy where lines can be determined without using nodes. An implementation following this recommendation also increases the data size of the location reference.

8. OpenLR™ - Physical data format binary specification

This section describes the arrangement of the data fields in a byte stream. It is assumed that a byte-oriented stream applies and 8 bits per byte are used. The byte order being used is the big endian format. This means that the most significant byte (MSB) comes first in the sequence of data.

The following tables show the order of the data from top to bottom. Each table describes a special location type and contains a sequence number, a field name and the assignment of each bit.

8.1 Line location

8.1.1. Line Location with 2 location reference points

A line location reference consists of at least 2 LR-points.

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	0 (no point)	0 (ArF0)	1 (has attributes)	011 (version)		
2	Longitude (1)	longitude highest byte							
3	Longitude (2)	longitude middle byte							
4	Longitude (3)	longitude lowest byte							
5	Latitude (1)	latitude highest byte							
6	Latitude (2)	latitude middle byte							
7	Latitude (3)	latitude lowest byte							
8	Attribute 1	RFU	RFU	functional road class			form of way		
9	Attribute 2	lowest frc next point			Bearing				
10	Attribute 3	distance to next point							
11	Longitude (1)	relative longitude highest byte							
12	Longitude (2)	relative longitude lowest byte							
13	Latitude (1)	relative latitude highest byte							
14	Latitude (2)	relative latitude lowest byte							
15	Attributes 1	RFU	RFU	functional road class			form of way		
16	Attributes 4	RFU	0/1 (POffF)	0/1 (NOffF)	Bearing				
17	PosOffset	positive offset (optional)							
18	NegOffset	negative offset (optional)							

Table 33: Line location with 2 LR-Points

Data size: 16 + [0/1/2] bytes

8.1.2. Line location with n location reference points (n > 2)

This type covers the previous type and just shows where intermediate location reference points are placed.

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	0 (no point)	0 (ArF0)	1 (has attributes)	011 (version)		
2	Longitude (1)	longitude highest byte							
3	Longitude (2)	longitude middle byte							
4	Longitude (3)	longitude lowest byte							
5	Latitude (1)	latitude highest byte							
6	Latitude (2)	latitude middle byte							
7	Latitude (3)	latitude lowest byte							
8	Attribute 1	RFU	RFU	functional road class			form of way		
9	Attribute 2	lowest frc next point			Bearing				
10	Attribute 3	distance to next point							
11	Longitude (1)	relative longitude highest byte							

12	Longitude (2)	relative longitude lowest byte			
13	Latitude (1)	relative latitude highest byte			
14	Latitude (2)	relative latitude lowest byte			
15	Attributes 1	RFU	RFU	functional road class	form of way
16	Attributes 2	lowest frc next point		Bearing	
17	Attributes 3	distance to next point			
…					
18 + 7*(i-4)	Longitude (1)	relative longitude highest byte			
19 +7* (i-4)	Longitude (2)	relative longitude lowest byte			
20 + 7* (i-4)	Latitude (1)	relative latitude highest byte			
21 + 7* (i-4)	Latitude (2)	relative latitude lowest byte			
22 + 7* (i-4)	Attributes 1	RFU	RFU	functional road class	form of way
23 + 7* (i-4)	Attributes 2	lowest frc next point		Bearing	
24 + 7* (i-4)	Attributes 3	distance to next point			
…					
18+7*(n-3)	Longitude (1)	relative longitude highest byte			
19+7*(n-3)	Longitude (2)	relative longitude lowest byte			
20+7*(n-3)	Latitude (1)	relative latitude highest byte			
21+7*(n-3)	Latitude (2)	relative latitude lowest byte			
22+7*(n-3)	Attributes 1	RFU	RFU	functional road class	form of way
23+7*(n-3)	Attributes 4	RFU	0/1 (POffF)	0/1 (NOffF)	Bearing
24+7*(n-3)	PosOffset	positive offset (optional)			
25+7*(n-3)	NegOffset	negative offset (optional)			

Table 34: Line location with n location reference points (n>2)

Note: 2 < i < n

Data size: 16 + (n-2)*7 + [0/1/2] bytes

8.1.3. Code size calculation

The code size of a line location reference depends on the number of LR-points included in the location reference. There must be at least two LR-points in the location reference. Also mandatory is the header with the status information. The following Table 35 shows a calculation of the message sizes depending on the number of LR-points.

# LR-points	Message size
2	16 bytes (+1 or +2 bytes offset, if included)
3	23 bytes (+1 or +2 bytes offset, if included)
4	30 bytes (+1 or +2 bytes offset, if included)
5	37 bytes (+1 or +2 bytes offset, if included)
6	44 bytes (+1 or +2 bytes offset, if included)
7	51 bytes (+1 or +2 bytes offset, if included)
8	58 bytes (+1 or +2 bytes offset, if included)
…	…

n (n>1)	16 + (n-2)*7 bytes (+1 or +2 bytes offset, if included)

Table 35: Message sizes depending on the number of LR-points

8.2 Point locations

8.2.1. Geo-coordinate

The simplest form of a point location is a single geo-coordinate.

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	1 (is point)	0 (ArF0)	0 (no attributes)	011 (version)		
2	Longitude (1)	longitude highest byte							
3	Longitude (2)	longitude middle byte							
4	Longitude (3)	longitude lowest byte							
5	Latitude (1)	latitude highest byte							
6	Latitude (2)	latitude middle byte							
7	Latitude (3)	latitude lowest byte							

Table 36: Point location – geo-coordinate

Data size: 7 bytes

8.2.2. Point along line

A point along line is referenced by a line and an offset value (measured from the start of the line).

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	1 (is point)	0 (ArF0)	1 (has attributes)	011 (version)		
2	Longitude (1)	longitude highest byte							
3	Longitude (2)	longitude middle byte							
4	Longitude (3)	longitude lowest byte							
5	Latitude (1)	latitude highest byte							
6	Latitude (2)	latitude middle byte							
7	Latitude (3)	latitude lowest byte							
8	Attribute 5	Orientation		functional road class			form of way		
9	Attribute 2	lowest frc next point			bearing				
10	Attribute 3	distance to next point							
11	Longitude (1)	relative longitude highest byte							
12	Longitude (2)	relative longitude lowest byte							
13	Latitude (1)	relative latitude highest byte							
14	Latitude (2)	relative latitude lowest byte							
15	Attributes 6	side of the road		functional road class			form of way		
16	Attributes 4	RFU	0/1 (POffF)	0 (NOffF)	bearing				
17	PosOffset	positive offset							

Table 37: Point location – Point along line

Data size: 16/17 bytes

8.2.3. Point of interest (POI) with access point

A point of interest with access point combines a point of interest located somewhere on the surface and an access point within a network.

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	1 (is point)	0 (ArF0)	1 (has attributes)	011 (version)		
2	Longitude (1)	longitude highest byte							
3	Longitude (2)	longitude middle byte							

4	Longitude (3)	longitude lowest byte							
5	Latitude (1)	latitude highest byte							
6	Latitude (2)	latitude middle byte							
7	Latitude (3)	latitude lowest byte							
8	Attribute 5	Orientation		functional road class			form of way		
9	Attribute 2	lowest frc next point			bearing				
10	Attribute 3	distance to next point							
11	Longitude (1)	relative longitude highest byte							
12	Longitude (2)	relative longitude lowest byte							
13	Latitude (1)	relative latitude highest byte							
14	Latitude (2)	relative latitude lowest byte							
15	Attributes 6	side of the road		functional road class			form of way		
16	Attributes 4	RFU	0/1 (POffF)	0 (NOffF)	bearing				
17	PosOffset	positive offset							
18	Longitude (1)	relative longitude highest byte (poi coord, relative to abs lon)							
19	Longitude (2)	relative longitude lowest byte (poi coord, relative to abs lon)							
20	Latitude (1)	relative latitude highest byte (poi coord, relative to abs lon)							
21	Latitude (2)	relative latitude lowest byte (poi coord, relative to abs lon)							

Table 38: Point location – Point of interest with access point

Data size: 20/21 bytes

8.3 Area locations

8.3.1. Circle

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	0 (no point)	0 (ArF0)	0 (no attri-butes)	011 (version)		
2	Longitude (1)	longitude center point highest byte							
3	Longitude (2)	longitude center point middle byte							
4	Longitude (3)	longitude center point lowest byte							
5	Latitude (1)	latitude center point highest byte							
6	Latitude (2)	latitude center point middle byte							
7	Latitude (3)	latitude center point lowest byte							
8	Radius (1)	small: lowest byte							
9	Radius (2)	medium: third byte (optional)							
10	Radius (3)	large: second byte (optional)							
11	Radius (4)	very large: highest byte (optional)							

Table 39: Circle location

Data Size: 7 + [1/2/3/4] bytes
Area category: $(00)_2$

8.3.2. Rectangle

A rectangle location reference consists of the lower left corner point as a pair of WGS84 coordinates in absolute format and the upper right corner point, given in absolute format (large rectangle) or relative format (standard rectangle). The choice between these two flavours is done in the binary encoder and depends on the distance between the lower left and upper right corner. The more compact relative coordinates are used whenever this distance is small enough (see also section 7.2). Otherwise, absolute coordinates are used.

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	1 (ArF1)	0 (no point)	0 (ArF0)	0 (no attri-butes)	011 (version)		
2	Longitude (1)	longitude lower left corner point highest byte							
3	Longitude (2)	longitude lower left corner point middle byte							
4	Longitude (3)	longitude lower left corner point lowest byte							
5	Latitude (1)	latitude lower left corner point highest byte							

6	Latitude (2)	latitude lower left corner point middle byte
7	Latitude (3)	latitude lower left corner point lowest byte
8	Longitude (1)	relative longitude upper right corner point highest byte
9	Longitude (2)	relative longitude upper right corner point lowest byte
10	Latitude (1)	relative latitude upper right corner point highest byte
11	Latitude (2)	relative latitude upper right corner point lowest byte

Table 40: Rectangle location with relative coordinates

Next, the table shows the representation for a rectangle with absolute coordinates:

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	1 (ArF1)	0 (no point)	0 (ArF0)	0 (no attri-butes)	011 (version)		
2	Longitude (1)	longitude lower left corner point highest byte							
3	Longitude (2)	longitude lower left corner point middle byte							
4	Longitude (3)	longitude lower left corner point lowest byte							
5	Latitude (1)	latitude lower left corner point highest byte							
6	Latitude (2)	latitude lower left corner point middle byte							
7	Latitude (3)	latitude lower left corner point lowest byte							
8	Longitude (1)	longitude upper right corner point highest byte							
9	Longitude (2)	longitude upper right corner point middle byte							
10	Longitude (3)	longitude upper right point lowest byte							
11	Latitude (1)	latitude upper right point highest byte							
12	Latitude (2)	latitude upper right point middle byte							
13	Latitude (3)	latitude upper right point lowest byte							

Table 41: Rectangle location with absolute coordinates

Data Size: 11/13 bytes (11 bytes: standard rectangle location, 13 bytes: large rectangle location reference)
Area category: $(10)_2$

8.3.3. Grid

The binary data format of a grid location reference is essentially that of a rectangle location which has been extended by two integer values, the number of columns and the number of rows. Analogous to the two different types of rectangle location references (standard and large), there are also two different types of grid location references, standard grid and large grid.

In the following, both possible binary representations are illustrated in their respective tables.

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	1 (ArF1)	0 (no point)	0 (ArF0)	0 (no attri-butes)	011 (version)		
2	Longitude (1)	longitude lower left corner point highest byte							
3	Longitude (2)	longitude lower left corner point middle byte							
4	Longitude (3)	longitude lower left corner point lowest byte							
5	Latitude (1)	latitude lower left corner point highest byte							
6	Latitude (2)	latitude lower left corner point middle byte							
7	Latitude (3)	latitude lower left corner point lowest byte							
8	Longitude (1)	relative longitude upper right corner point highest byte							
9	Longitude (2)	relative longitude upper right corner point lowest byte							
10	Latitude (1)	relative latitude upper right corner point highest byte							
11	Latitude (2)	relative latitude upper right corner point lowest byte							
12	NumCols (1)	number of columns highest byte							
13	NumCols (2)	number of columns lowest byte							
14	NumRows (1)	number of rows highest byte							
15	NumRows (2)	number of rows lowest byte							

Table 42:Grid location with relative coordinates

Next, the table shows the representation for a large grid:

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	1 (ArF1)	0 (no point)	0 (ArF0)	0 (no attri-butes)	011 (version)		
2	Longitude (1)	longitude lower left corner point highest byte							
3	Longitude (2)	longitude lower left corner point middle byte							
4	Longitude (3)	longitude lower left corner point lowest byte							
5	Latitude (1)	latitude lower left corner point highest byte							
6	Latitude (2)	latitude lower left corner point middle byte							
7	Latitude (3)	latitude lower left corner point lowest byte							
8	Longitude (1)	longitude upper right corner point highest byte							
9	Longitude (2)	longitude upper right corner point middle byte							
10	Longitude (3)	longitude upper right point lowest byte							
11	Latitude (1)	latitude upper right point highest byte							
12	Latitude (2)	latitude upper right point middle byte							
13	Latitude (3)	latitude upper right point lowest byte							
14	NumCols (1)	number of columns highest byte							
15	NumCols (2)	number of columns lowest byte							
16	NumRows (1)	number of rows highest byte							
17	NumRows (2)	number of rows lowest byte							

Table 43: Grid location with absolute coordinates

Data Size: 15/17 bytes (15 bytes: standard grid location, 17 bytes: large grid location reference)
Area category: $(10)_2$

8.3.4. Polygon

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	0 (ArF1)	0 (no point)	1 (ArF0)	0 (no attri-butes)	011 (version)		
2	Longitude (1)	longitude first corner point highest byte							
3	Longitude (2)	longitude first corner point middle byte							
4	Longitude (3)	longitude first corner point lowest byte							
5	Latitude (1)	latitude first corner point highest byte							
6	Latitude (2)	latitude first corner point middle byte							
7	Latitude (3)	latitude first corner point lowest byte							
8	Longitude (1)	relative longitude second corner point highest byte							
9	Longitude (2)	relative longitude second corner point lowest byte							
10	Latitude (1)	relative latitude second corner point highest byte							
11	Latitude (2)	relative latitude second corner point lowest byte							
…									
12+(i-4)*4	Longitude (1)	relative longitude ith corner point highest byte							
13+(i-4)*4	Longitude (2)	relative longitude ith corner point lowest byte							
14+(i-4)*4	Latitude (1)	relative latitude ith corner point highest byte							
15+(i-4)*4	Latitude (2)	relative latitude ith corner point lowest byte							
…									
12+(n-3)*4	Longitude (1)	relative longitude nth corner point highest byte							
13+(n-3)*4	Longitude (2)	relative longitude nth corner point lowest byte							
14+(n-3)*4	Latitude (1)	relative latitude nth corner point highest byte							
15+(n-3)*4	Latitude (2)	relative latitude nth corner point lowest byte							

Table 44: Polygon location with n corner points (n > 2)

Note: 2<i<n
Data Size: $15+(n-3)^*4$ bytes
Area category: $(01)_2$

8.3.5. ClosedLine

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	1 (ArF1)	0 (no point)	1 (ArF0)	1 (has attri-	011 (version)		

						butes)			
2	Longitude (1)	longitude lower left corner point highest byte							
3	Longitude (2)	longitude lower left corner point middle byte							
4	Longitude (3)	longitude lower left corner point lowest byte							
5	Latitude (1)	latitude lower left corner point highest byte							
6	Latitude (2)	latitude lower left corner point middle byte							
7	Latitude (3)	latitude lower left corner point lowest byte							
8	Attribute 1	RFU	RFU	functional road class			form of way		
9	Attribute 2	lowest frc next point			bearing				
10	Attribute 3	distance to next point							
11	Attribute 1	RFU	RFU	functional road class			form of way		
12	Attribute 7	RFU			bearing				

Table 45: Closed line location with two location reference points

Data Size: 12 bytes
Area category: $(11)_2$

Byte number	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	Status	RFU	1 (ArF1)	0 (no point)	1 (ArF0)	1 (has attri-butes)	011 (version)		
2	Longitude (1)	longitude lower left corner point highest byte							
3	Longitude (2)	longitude lower left corner point middle byte							
4	Longitude (3)	longitude lower left corner point lowest byte							
5	Latitude (1)	latitude lower left corner point highest byte							
6	Latitude (2)	latitude lower left corner point middle byte							
7	Latitude (3)	latitude lower left corner point lowest byte							
8	Attribute 1	RFU	RFU	functional road class			form of way		
9	Attribute 2	lowest frc next point			bearing				
10	Attribute 3	distance to next point							
11	Longitude (1)	relative longitude upper right corner point highest byte							
12	Longitude (2)	relative longitude upper right corner point lowest byte							
13	Latitude (1)	relative latitude upper right corner point highest byte							
14	Latitude (2)	relative latitude upper right corner point lowest byte							
15	Attribute 1	RFU	RFU	functional road class			form of way		
16	Attribute 2	lowest frc next point			bearing				
17	Attribute 3	distance to next point							
…									
18+7*(n-4)	Longitude (1)	relative longitude upper right corner point highest byte							
19+7*(n-4)	Longitude (2)	relative longitude upper right corner point lowest byte							
20+7*(n-4)	Latitude (1)	relative latitude upper right corner point highest byte							
21+7*(n-4)	Latitude (2)	relative latitude upper right corner point lowest byte							
22+7*(n-4)	Attribute 1	RFU	RFU	functional road class			form of way		
23+7*(n-4)	Attribute 2	lowest frc next point			bearing				
24+7*(n-4)	Attribute 3	distance to next point							
25+7*(n-4)	Attribute 1	RFU	RFU	functional road class			form of way		
26+7*(n-4)	Attribute 7	RFU			bearing				

Table 46: Closed line location with n location reference points (n>2)

Note: 2<i<n
Data Size: 19+7*(n-3) bytes
Area category: $(11)_2$

8.4 Identification of the location type

All location types need to be identified during decoding. This identification is based on the data size and the combination of the flags point, area and attributes. The Table 47 shows a mapping from location type to the possible data sizes and the settings of the flags. Table 48 shows a mapping from data size to the possible location types.

Location type	Data size	Point flag	Area flag	Attributes flag
Line	16 + (n-2) * 7 + [0/1/2] bytes	0	00	1
Geo-coordinate	7 bytes	1	00	0
Point along line	16/17 bytes	1	00	1

POI with access point	20/21 bytes	1	00	1
Circle	7 + [1/2/3/4] bytes	0	00	0
Rectangle	11/13 bytes	0	10	0
Grid	15/17 bytes	0	10	0
Polygon	15 + (n-3) * 4 bytes	0	01	0
ClosedLine	19 + (n-3) * 7 bytes	0	11	1

Table 47: Identification depending on flags

Number of bytes	Location type
1	- not allowed -
2	- not allowed -
3	- not allowed -
4	- not allowed -
5	- not allowed -
6	- not allowed -
7	*Geo-coordinate*
8	*Circle*
9	*Circle*
10	*Circle*
11	*Circle or Rectangle*
12	*Closed line*
13	*Rectangle*
14	- not allowed -
15	*Grid or Polygon*
16	*Line location (2 LRP) or Point along line*
17	*Line location (2 LRP, 1 offset) or Point along line or Grid*
18	*Line location (2 LRP, 2 offsets)*
19	*Polygon or Closed line*
20	*POI with access point*
21	*POI with access point*
22	- not allowed -
23	*Line location (3 LRP) or Polygon*
24	*Line location (3 LRP, 1 offset)*

Table 48: Identification depending on data size

8.5 Changes

The following lists summarize the changes of the binary physical format.

8.5.1. Version 2 -> Version 3

- Offsets are using the relative format (calculating the relation between the offset length and the location length)
- Version bits (0-2) in the status byte indicate version 3 = (011b)
    - Important note: the value 0 = (000b) is not used yet and might indicate in a later version of the format that a second status byte is used (representing a version number higher than 7!)
- Bit 5 of the status byte indicates whether or not the location is a point location (0: no point location, 1: point location)
- Bits 7 and 6 of the first attribute of the first location reference point are used for the orientation of a point location (only valid for point locations)
- Bits 7 and 6 of the first attribute of the last location reference point are used for the side of the road of a point location (only valid for point locations)

8.5.2. Version 3 (adding area location support)

- Area flag consists of two bits ArF1 and ArF0
- Support for area locations

Chapter E

Physical data format (XML)

9. OpenLR™ - XML representation

This physical data format describes an XML format for the OpenLR™ data format. It uses the components described in the logical data format in section 5.

9.1 Data types

The physical data format uses the following data types. Table 49 gives an overview of all available data types and specifies the name and the type for each data type. In the following sections the data type names are used to indicate the type for each data component.

Data type name	Type
integer	The integer data type is used to specify a numeric value without a fractional component.
double	The double data type is used to specify a numeric value with a fractional component.
short	The short data type is used to specify a numeric value without a fractional component.
string	The string data type can contain characters, line feeds, carriage returns, space and tab characters.
DATA	The data type can hold any kind of XML data not specified by the schema. It allows documents to contain additional elements.
Base64Data	Binary data type to express binary-formatted data (Base64-encoded).

Table 49: Physical format: Data types (XML)

9.2 Coordinates

Each point in a map consists of a coordinate pair “longitude” (lon) and “latitude” (lat) represented in WGS84 coordinates. The directions north and east are represented by positive values (longitude and latitude respectively).

The coordinate values will be transmitted as double values. The physical format makes use of absolute coordinates which represent the designated values of the geographical position. The longitude and latitude fields are mandatory.

```
<openlr:Coordinates>
  <openlr:Longitude> [double] </openlr:Longitude>
  <openlr:Latitude> [double] </openlr:Latitude>
</openlr:Coordinates>
```

9.3 Line attribute values

Line attributes are part of a location reference point and consist of functional road class (FRC), form of way (FOW) and bearing (BEAR) data. All components are mandatory fields.

```
<openlr:LineAttributes>
  <openlr:FRC> [FRC value] </openlr:FRC>
  <openlr:FOW> [FOW value] </openlr:FOW>
  <openlr:BEAR> [integer] </openlr:BEAR>
</openlr:LineAttributes>
```

9.3.1. Functional Road Class (FRC)

The functional road class (FRC) can hold eight different values as described in the logical format. These eight values are shown in Table 50.

```
<openlr:FRC> [FRC value] </openlr:FRC>
```

Data type	Value	Description
FRC value	FRC0	Main road, highest importance
	FRC 1	First class road
	FRC 2	Second class road
	FRC 3	Third class road
	FRC 4	Fourth class road
	FRC 5	Fifth class road
	FRC 6	Sixth class road
	FRC 7	Other class road, lowest importance

Table 50: Physical format: Functional road class (XML)

9.3.2. Form of way (FOW)

The form of way (FOW) can hold eight different values as described in the logical format. These eight values are shown in Table 51.

```
<openlr:FOW> [FOW value] </openlr:FOW>
```

Data type	Value
FOW value	UNDEFINED
	MOTORWAY
	MULTIPLE_CARRIAGEWAY
	SINGLE_CARRIAGEWAY
	ROUNDABOUT
	TRAFFICSQUARE
	SLIPROAD
	OTHER

Table 51: Physical format: Form of way (XML)

9.3.3. Bearing (BEAR)

The bearing describes the angle between the true North and the road as described in the logical format. The physical data format defines the bearing field as an integer value between 0 and 360 whereby “0” is included and “360” is excluded from that range. The values should be rounded if not available as an integer.

```
<openlr:BEAR> [integer] </openlr:BEAR>
```

9.4 Path attribute values

The field path attributes is part of a location reference point (except for the last location reference point) and consists of lowest functional road class (LFRCNP) and distance to next point (DNP) data. All components are mandatory fields.

```
<openlr:PathAttributes>
  <openlr:LFRCNP> [FRC value] </openlr:LFRCNP>
```

```
    <openlr:DNP> [integer] </openlr:DNP>
  </openlr:PathAttributes>
```

9.4.1. Distance to next LR-point (DNP)

The DNP attribute measures the distance in meters between two consecutive LR-points along the location reference path as described in the logical format.

```
  <openlr:DNP> [integer] </openlr:DNP>
```

9.4.2. Lowest FRC to next point (LFRCNP)

The lowest FRC to the next point indicates the lowest functional road class used in the location reference path to the next LR-point. This information could be used to limit the number of road classes which need to be scanned during the decoding. See Table 50 for a definition of the data type.

```
    <openlr:LFRCNP> [FRC value] </openlr:LFRCNP>
```

9.5 Point attribute values

9.5.1. Side of road

A point location may have additional information on which side of the road the point is located. Examples for such point locations are particular house numbers, speed cams or traffic signs. This additional information is valid for the types “point along line” and “poi with access point”.

```
  <openlr:SideOfRoad> [SideOfRoad value] </openlr:SideOfRoad>
```

SideOfRoad value	Meaning
ON_ROAD_OR_UNKNOWN	Point is directly on (or above) the road, or determination of right/left side is not applicable (default)
RIGHT	Point is on right side of the road
LEFT	Point is on the left side of the road
BOTH	Point is on both sides of the road

Table 52: Physical format: SideOfRoad values (XML)

9.5.2. Orientation

A point location may have additional information in which direction the information at that point is relevant. Examples for such point locations are speed cams or traffic signs. This additional information is valid for the types “point along line” and “poi with access point”.

```
  <openlr:Orientation> [Orientation value] </openlr:Orientation>
```

Orientation value	Meaning
NO_ORIENTATION_OR_UNKNOWN	Point has no sense of orientation, or determination of orientation is not applicable (default)
WITH_LINE_DIRECTION	Point has orientation from first LRP towards second

	LRP
AGAINST_LINE_DIRECTION	Point has orientation from second LRP towards first LRP
BOTH	Point has orientation in both directions

Table 53: Orientation of a point location

9.6 Area attribute values

9.6.1. Radius

The radius attribute described the radius of the circle location in meters. The radius is a positive integer.

```
<openlr:Radius> [integer] </openlr:Radius>
```

9.6.2. Number of columns / rows

The dimension attribute is used to define the number of columns as well as the number of rows for a grid location. The dimension is a positive value and must be representable by a short integer.

```
<openlr:Dimension> [short] </openlr:Dimension>
```

9.6.3. Rectangle

The rectangle describes the bounding box which covers the location reference path for rectangle locations as well as grid locations. The rectangle is defined by the coordinates of its lower left and upper right corner. Both coordinates are mandatory fields and consist of a longitude and latitude pair.

```
<openlr:Rectangle>
  <openlr:LowerLeft>
   <openlr:Longitude> [double] </openlr:Longitude>
   <openlr:Latitude> [double] </openlr:Latitude>
  </openlr:LowerLeft>
  <openlr:UpperRight>
   <openlr:Longitude> [double] </openlr:Longitude>
   <openlr:Latitude> [double] </openlr:Latitude>
  </openlr:UpperRight>
</openlr:Rectangle>
```

9.7 Line location reference

A line location reference describes a path within a map and consists of location reference point(s), a last location reference point and offset data. There must be at least one location reference point and exactly one last location reference point. The offset field is optional.

```
  <openlr:LineLocationReference>
    [LocationReferencePoint(s)]
    [LastLocationReferencePoint]
    [Offsets]
  </openlr:LineLocationReference>
```

9.7.1. Location reference point

A location reference point consists of coordinate, line attribute and path attribute data. All fields are mandatory.

```
  <openlr:LocationReferencePoint>
    [Coordinates]
    [LineAttributes]
    [PathAttributes]
  <openlr:LocationReferencePoint>
```

9.7.2. Last location reference point

A last location reference point consists of coordinate and line attribute data. All fields are mandatory.

```
  <openlr:LastLocationReferencePoint>
    [Coordinates]
    [LineAttributes]
  </openlr:LastLocationReferencePoint>
```

9.7.3. Offsets

Offsets are used to locate the start and end of a location more precisely than bounding to the nodes in a network. The logical format defines two offsets, one at the start of the location and one at the end of the location. Both offsets operate along the lines of the location and are measured in meters. The offset values are optional and a missing offset value means an offset of 0 meters. If there are no offsets at all the offset field can be omitted.

```
  <openlr:Offsets>
    <openlr:PosOff> [integer] </openlr:PosOff>
    <openlr:NegOff> [integer] </openlrNegOff>
  </openlr:Offsets>
```

9.8 Point location reference

A point location reference consists of exactly one out of GeoCoordinate, PointAlongLine or PoiWithaccessPoint.

```
  <openlr:PointLocationReference>
    [GeoCoordinate]
    [PointAlongLine]
    [PoiWithAccessPoint]
  </openlr:PointlocationReference>
```

9.8.1. GeoCoordinate

A GeoCoordinate consists of a single coordinate pair.

```
  <openlr:GeoCoordinate>
    [Coordinates]
  </openlr:GeoCoordinate>
```

9.8.2. PointAlongLine

A PointAlongLine consists of a location reference point, a last location reference point, optional offsets, and information about the side of the road and the orientation.

```
  <openlr:PointAlongLine>
    [LocationReferencePoint]
    [LastLocationReferencePoint]
    [Offsets]
    <openlr:SideOfRoad> [SideOfRoad] </openlr:SideOfRoad>
    <openlr:Orientation> [Orientation] </openlr:Orientation>
  </openlr:PointAlongLine>
```

9.8.3. PoiWithAccessPoint

A PoiWithAccessPoint consists of a location reference point, a last location reference point, a coordinate, optional offsets, and information about the side of the road and the orientation.

```
  <openlr:PointAlongLine>
    [LocationReferencePoint]
    [LastLocationReferencePoint]
    [Coordinates]
    [Offsets]
    <openlr:SideOfRoad> [SideOfRoad] </openlr:SideOfRoad>
    <openlr:Orientation> [Orientation] </openlr:Orientation>
  </openlr:PointAlongLine>
```

9.9 Area location reference

An area location reference consists of the following five location references of type area location: circle, polygon, rectangle, grid, and closed line.

```
<openlr:AreaLocationReference>
  <openlr:CircleLocationReference> [CircleLocationReference]
  </openlr:CircleLocationReference>
  <openlr:PolygonLocationReference> [PolygonLocationReference]
  </openlr:PolygonLocationReference>
  <openlr:RectangleLocationReference> [RectangleLocationReference]
  </openlr:RectangleLocationReference>
  <openlr:GridLocationReference> [GridLocationReference]
  </openlr:GridLocationReference>
  <openlr:ClosedLineLocationReference> [ClosedlineLocationReference]
  </openlr:ClosedLineLocationReference>
</openlr:AreaLocationReference>
```

9.9.1. Circle Location Reference

A circle location reference consists of a geo coordinate pair describing the center point and the radius.

```
<openlr:CicleLocationReference>
  [Coordinates]
  [Radius]
</openlr:CircleLocationReference>
```

9.9.2. Polygon Location Reference

A polygon location reference consists of a list of coordinates describing the corner points of the polygon.

```
<openlr:PolygonLocationReference>
  [Coordinates](3…n)
</openlr:PolygonLocationReference>
```

9.9.3. Rectangle Location Reference

A Rectangle location reference consists of a rectangle.

```
<openlr:RectangleLocationReference>
  [Rectangle]
</openlr:RectangleLocationReference>
```

9.9.4. Grid Location Reference

A Grid location reference consists of a rectangle, the number of columns and the number of rows.

```
<openlr:RectangleLocationReference>
  [Rectangle]
  <openlr:Dimension> [columns] </openlr:Dimension>
  <openlr:Dimension> [rows] </openlr:Dimension>
</openlr:RectangleLocationReference>
```

9.9.5. Closed Line Location Reference

A closed line location describes a path within a map and consists of at least two location reference points (LRPs).

```
<openlr:ClosedLineLocationReference>
  [LocationReferencepoint(s)]
  [LineAttributes]
</openlr:ClosedLineLocationReference>
```

9.10 XML location reference

An XML location reference consists of a line location, point location or area location reference. This inner field is mandatory.

```
  <openlr:XMLLocationReference>
    <openlr:LineLocationReference> [LineLocationReference]
    </openlr:LineLocationReference>
    <openlr:PointLocationReference> [PointLocationReference]
    </openlr:PointLocationReference>
    <openlr:AreaLocationReference> [AreaLocationReference]
    </openlr:AreaLocationReference>
  </openlr:XMLLocationReference>
```

9.11 Binary location references

The field “binary location references” holds the binary representation of the same location reference. There may exist several different versions of the binary format but at least one binary location reference is mandatory.

```
<openlr:BinaryLocationReferences>
  [BinaryLocationReference(s)]
</openlr:BinaryLocationReferences>
```

9.11.1. Binary location reference

A binary location reference represents the binary format of the same location reference as described in this XML representation. This field contains the Base64-encoded binary data. The attributes id and version are mandatory whereby the id identifies this specific binary location reference in the list of possibly several binary location references and the version expresses the binary format version being used.

```
<openlr:BinaryLocationReference id="[name]" version="[version]">
  [Base64Data]
</openlr:BinaryLocationReference>
```

9.12 Additional information

The additional information field consists of extra information not being relevant for the OpenLR™ location reference data. This field is optional and also all inner fields are optional. If no additional data is specified this field can be omitted. This field consists of bounding box, map database and any other kind of XML data.

```
<openlr:AdditionalInformation>
  <openlr:BoundingBox> [Rectangle] </openlr:BoundingBox>
  <openlr:MapDatabase> [string] </openlr:MapDatabase>
  <openlr:AdditionalData> [DATA] </openlr:AdditionalData>
</openlr:AdditionalInformation>
```

9.12.1. Bounding box

The (optional) bounding box describes a rectangle which covers the location reference path completely. The bounding box is defined by the coordinates of its lower left and upper right corner. Both coordinates are mandatory fields and consist of a longitude and latitude pair.

```
<openlr:BoundingBox> [Rectangle] </openlr:BoundingBox>
```

9.12.2. Map database

The (optional) map database field provides information about the map being used during decoding. The information can be of any type but must be expressible as a string.

```
<openlr:MapDatabase> [string] </openlr:MapDatabase>
```

9.12.3. Additional data

The (optional) additional data field makes the XML format extendable with any kind of other XML data not specified in the XML format schema definition. This field may be used to transmit additional data like debug data or decoding hints.

```
  <openlr:AdditionalData> [DATA] </openlr:AdditionalData>
```

9.13 Root tag: OpenLR

The root tag of the XML format is OpenLR. This tag includes the namespace definitions and schema locations. The tag consists of a location ID and XML location reference, binary location reference and additional information data. The location ID is mandatory; all other data fields are optional. But there should be at least one of the data fields XML location reference or binary location references.

```
  <?xml version="1.0" encoding="UTF-8"?>
  <openlr:OpenLR xmlns:openlr=http://www.openlr.org/openlr
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.openlr.org/openlr openlr.xsd ">
    [LocationID]
    [XMLLocationReference]
    [BinaryLocationReferences]
    [AdditionalInformation]
  <openlr:OpenLR>
```

9.13.1. Location ID

The location ID identifes this location reference uniquely. The identifier can be of any type but must be expressible as a string.

```
  <openlr:LocationID> [string] </openlr:LocationID>
```

10. OpenLR™ - Physical data format XML specification

The XML format for OpenLR™ location references is defined by the following XML schema definition. The schema includes additional documentation.
Figure 23 shows the structure of such XML documents in a graphical way.

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:openlr="http://www.openlr.org/openlr" attributeFormDefault="unqualified"
    elementFormDefault="qualified" targetNamespace="http://www.openlr.org/openlr">

    <xs:annotation>
        <xs:documentation xml:lang="en">
            This schema describes the XML format for OpenLR location references. This format includes a
            XMLLocationReference but also supports BinaryLocationReference. The XMLLocationReference shall make use of
            the raw data for any type so that no compression techniques are used (e.g. use of intervals for the description of
            length values). The BinaryLocationReference shall represent the compressed format of the location reference. For
            further information on the relevant fields see the OpenLR white paper at http://www.openlr.org.

            OpenLR is a trade mark of TomTom International B.V.
            author: TomTom International B.V.
        </xs:documentation>
    </xs:annotation>


    <xs:simpleType name="FRC_Type">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The functional road class is a road classification based on the importance of a road.
            </xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="FRC0" />
            <xs:enumeration value="FRC1" />
            <xs:enumeration value="FRC2" />
            <xs:enumeration value="FRC3" />
            <xs:enumeration value="FRC4" />
            <xs:enumeration value="FRC5" />
            <xs:enumeration value="FRC6" />
            <xs:enumeration value="FRC7" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="FOW_Type">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The form of way describes the physical road type.
            </xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="UNDEFINED" />
            <xs:enumeration value="MOTORWAY" />
            <xs:enumeration value="MULTIPLE_CARRIAGEWAY" />
            <xs:enumeration value="SINGLE_CARRIAGEWAY" />
            <xs:enumeration value="ROUNDABOUT" />
            <xs:enumeration value="TRAFFICSQUARE" />
            <xs:enumeration value="SLIPROAD" />
            <xs:enumeration value="OTHER" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="Bearing_Type">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The bearing describes the angle between the true North and a line. The values range from 0 to 359 degree.
            </xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="0" />
            <xs:maxExclusive value="360" />
        </xs:restriction>
    </xs:simpleType>
```

```
<xs:simpleType name="Radius_Type">
     <xs:annotation>
          <xs:documentation xml:lang="en">
               The radius of the circle
               area location. The values range
               from 0 to max Integer.
          </xs:documentation>
     </xs:annotation>
     <xs:restriction base="xs:integer">
          <xs:minInclusive value="0" />
     </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Dimension_Type">
     <xs:annotation>
          <xs:documentation xml:lang="en">
               A dimension (number of
               columns or number of rows) of the
               grid area location. The values
               range
               from 0 to max Short.
          </xs:documentation>
     </xs:annotation>
     <xs:restriction base="xs:short">
          <xs:minInclusive value="0" />
     </xs:restriction>
</xs:simpleType>

<xs:simpleType name="SideOfRoad_Type">
     <xs:annotation>
          <xs:documentation xml:lang="en">
               The side of the road is valid only for point locations and indicates on which side of the referenced road the point
               location is located.
          </xs:documentation>
     </xs:annotation>
     <xs:restriction base="xs:string">
          <xs:enumeration value="ON_ROAD_OR_UNKNOWN" />
          <xs:enumeration value="RIGHT" />
          <xs:enumeration value="LEFT" />
          <xs:enumeration value="BOTH" />
     </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Orientation_Type">
     <xs:annotation>
          <xs:documentation xml:lang="en">
               The orientation is only valid for point locations and indicates for which direction the point information is relevant.
          </xs:documentation>
     </xs:annotation>
     <xs:restriction base="xs:string">
          <xs:enumeration value="NO_ORIENTATION_OR_UNKNOWN" />
          <xs:enumeration value="WITH_LINE_DIRECTION" />
          <xs:enumeration value="AGAINST_LINE_DIRECTION" />
          <xs:enumeration value="BOTH" />
     </xs:restriction>
</xs:simpleType>

<xs:complexType name="LineAttributes">
     <xs:annotation>
          <xs:documentation xml:lang="en">
            Line attributes describe features of a line in a network. The attributes the functional road class, form of way and
            the bearing of a line.
          </xs:documentation>
     </xs:annotation>
     <xs:all>
          <xs:element name="FRC" type="openlr:FRC_Type" />
          <xs:element name="FOW" type="openlr:FOW_Type" />
          <xs:element name="BEAR" type="openlr:Bearing_Type" />
     </xs:all>
</xs:complexType>

<xs:complexType name="PathAttributes">
     <xs:annotation>
          <xs:documentation xml:lang="en">
```

```
                Path attributes describes features of a path. The LFRCNP attribute is the lowest functional road class value
                which appears in the path. The highest FRC value is FRC0 and the lowest possible FRC value is FRC7.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element name="LFRCNP" type="openlr:FRC_Type" />
            <xs:element name="DNP" type="xs:integer" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="Coordinates">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                Coordinates stands for a pair of WGS84 longitude and latitude values. This coordinate pair specifies a
                geometric point in a digital map.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element name="Longitude" type="xs:double" />
            <xs:element name="Latitude" type="xs:double" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="Rectangle">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The bounding box describes a rectangle which covers the location completely. The rectangle is defined by the
                coordinates of the lower left and upper right corners.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element name="LowerLeft" type="openlr:Coordinates" />
            <xs:element name="UpperRight" type="openlr:Coordinates" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="Offsets">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                Offsets are used to shorten the location reference path at its start and end. The new positions along the location
                reference path indicate the real start and end of the location. The positive offset is the difference of the start
                point of the location reference and the start point of the desired location along the location reference path. The
                negative offset is the difference of the end point of the desired location and the end point of the location
                reference along the location reference path. Both values are measured in meter.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="0" name="PosOff" type="xs:integer" />
            <xs:element minOccurs="0" name="NegOff" type="xs:integer" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="AdditionalData">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                AdditionalData can be used to add any other kind of information needed like debug information or additional
                encoder/decoder flags.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:any minOccurs="0" maxOccurs="unbounded" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="AdditionalInformation">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                AdditionalInformation can be added if necessary. This includes the bounding box covering the location and
                information on the map used for encoding. The AdditionalData section also provides the freedom to add any
                other information needed.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="0" name="BoundingBox" type="openlr:Rectangle" />
            <xs:element minOccurs="0" name="MapDatabase" type="xs:string" />
            <xs:element minOccurs="0" name="AdditionalData" type="openlr:AdditionalData" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="LocationReferencePoint">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The basis of a location reference is a sequence of location reference points (LRPs). Such a LRP contains a
                coordinate pair, specified in WGS84 longitude and latitude values and additionally several line and path
                attributes.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" name="Coordinates" type="openlr:Coordinates" />
            <xs:element minOccurs="1" name="LineAttributes" type="openlr:LineAttributes" />
            <xs:element minOccurs="1" name="PathAttributes" type="openlr:PathAttributes" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="LastLocationReferencePoint">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The sequence of location reference points is terminated by a last location reference point. This LRP does not
                need to have any path attributes.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" name="Coordinates" type="openlr:Coordinates" />
            <xs:element minOccurs="1" name="LineAttributes" type="openlr:LineAttributes" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="BinaryLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                The BinaryLocationReference is the Base64-encoded binary representation of the location reference. Each
                binary location reference needs to have an identifier and a version. The version is defined by the version of the
                binary physical format.
            </xs:documentation>
        </xs:annotation>
        <xs:simpleContent>
            <xs:extension base="xs:base64Binary">
                <xs:attribute name="id" type="xs:string" use="required" />
                <xs:attribute name="version" type="xs:string" use="required" />
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="LineLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A LineLocationReference is defined by an ordered sequence of location reference points and a terminating last
                location reference point.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="unbounded"
                name="LocationReferencePoint" type="openlr:LocationReferencePoint" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="LastLocationReferencePoint" type="openlr:LastLocationReferencePoint" />
            <xs:element minOccurs="0" name="Offsets" type="openlr:Offsets" />
        </xs:sequence>
    </xs:complexType>

<xs:complexType name="ClosedLineLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A LineLocationReference   is defined by an ordered sequence of location reference points and a terminating last
                 location reference point.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="unbounded"
                name="LocationReferencePoint" type="openlr:LocationReferencePoint" />
            <xs:element minOccurs="1" maxOccurs="1" name="LastLine"
                type="openlr:LineAttributes" />
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="GeoCoordinate">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                GeoCoordinate is a point location which consists of exactly one oordinate pair.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" maxOccurs="1" name="Coordinates"
                type="openlr:Coordinates" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="PointAlongLine">
        <xs:annotation>
            <xs:documentation xml:lang="en">
            Point along line is a point location which is defined by a line and an offset value. The line will be referenced by two
            location reference points and the concrete position on that line is referenced using the positive offset. Additionally
            information about the side of the road where the point is located and the orientation with respect to the direction of
            the line can be added.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" maxOccurs="1"
                name="LocationReferencePoint" type="openlr:LocationReferencePoint" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="LastLocationReferencePoint" type="openlr:LastLocationReferencePoint" />
            <xs:element minOccurs="0" maxOccurs="1" name="Offsets"
                type="openlr:Offsets" />
            <xs:element minOccurs="0" maxOccurs="1" name="SideOfRoad"
                type="openlr:SideOfRoad_Type" default="ON_ROAD_OR_UNKNOWN" />
            <xs:element minOccurs="0" maxOccurs="1" name="Orientation"
                type="openlr:Orientation_Type" default="NO_ORIENTATION_OR_UNKNOWN" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="PoiWithAccessPoint">
        <xs:annotation>
            <xs:documentation xml:lang="en">
            Point along line with access is a point location which is defined by a line, an offset value and a coordinate. The
            line will be referenced by two location reference points and the concrete position of the access point on that line is
            referenced using the positive offset. The point of interest is identified by the coordinate pair. Additionally information
            about the side of the road where the point is located and the orientation with respect to the direction of the line can
            be added.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" maxOccurs="1"
                name="LocationReferencePoint" type="openlr:LocationReferencePoint" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="LastLocationReferencePoint" type="openlr:LastLocationReferencePoint" />
            <xs:element minOccurs="0" maxOccurs="1" name="Offsets"
                type="openlr:Offsets" />
            <xs:element minOccurs="1" maxOccurs="1" name="Coordinates"
                type="openlr:Coordinates" />
            <xs:element minOccurs="0" maxOccurs="1" name="SideOfRoad"
                type="openlr:SideOfRoad_Type" default="ON_ROAD_OR_UNKNOWN" />
            <xs:element minOccurs="0" maxOccurs="1" name="Orientation"
                type="openlr:Orientation_Type" default="NO_ORIENTATION_OR_UNKNOWN" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="PointLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A PointLocationReference represents a point of interest. The point may be bound to the underlying network.
            </xs:documentation>
        </xs:annotation>
        <xs:choice>
            <xs:element minOccurs="1" maxOccurs="1" name="GeoCoordinate"
                type="openlr:GeoCoordinate" />
            <xs:element minOccurs="1" maxOccurs="1" name="PointAlongLine"
                type="openlr:PointAlongLine" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="PoiWithAccessPoint" type="openlr:PoiWithAccessPoint" />
        </xs:choice>
    </xs:complexType>

    <xs:complexType name="BinaryLocationReferences">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                It may exist several different formats for binary location references and all formats or a subset can be included.
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="unbounded"
                name="BinaryLocationReference" type="openlr:BinaryLocationReference" />
        </xs:sequence>
    </xs:complexType>

<xs:complexType name="AreaLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                An AreaLocationReference represents a set of area location like circle polygon rectangle and grid location.
            </xs:documentation>
        </xs:annotation>
        <xs:choice>
            <xs:element minOccurs="1" maxOccurs="1"
                name="CircleLocationReference" type="openlr:CircleLocationReference" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="PolygonLocationReference" type="openlr:PolygonLocationReference" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="RectangleLocationReference" type="openlr:RectangleLocationReference" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="GridLocationReference" type="openlr:GridLocationReference" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="ClosedLineLocationReference" type="openlr:ClosedLineLocationReference" />
        </xs:choice>
    </xs:complexType>

    <xs:complexType name="CircleLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A CircleLocationReference represents a circle area location. The circle area may be bound to the underlying
                network.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" maxOccurs="1" name="GeoCoordinate"
                type="openlr:GeoCoordinate" />
            <xs:element minOccurs="1" maxOccurs="1" name="Radius"
                type="openlr:Radius_Type" />
        </xs:all>
    </xs:complexType>


    <xs:complexType name="RectangleLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A RectangleLocationReference represents a rectangular area location.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" maxOccurs="1" name="Rectangle"
                type="openlr:Rectangle" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="GridLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A GridLocationReference represents a rectangular area location with a grid of ncols columns and nrows rows.
            </xs:documentation>
        </xs:annotation>
        <xs:all>
            <xs:element minOccurs="1" maxOccurs="1" name="Rectangle"
                type="openlr:Rectangle" />
            <xs:element minOccurs="1" maxOccurs="1" name="NumColumns"
                type="openlr:Dimension_Type" />
            <xs:element minOccurs="1" maxOccurs="1" name="NumRows"
                type="openlr:Dimension_Type" />
        </xs:all>
    </xs:complexType>

    <xs:complexType name="PolygonLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A PolygonLocationReference represents a polygonal area location
            </xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element minOccurs="1" maxOccurs="1" name="PolygonCorners">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="3" maxOccurs="unbounded" name="Coordinates"
                                    type="openlr:Coordinates" />
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="XMLLocationReference">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                A XMLLocationReference describes an OpenLR location reference. Currently only line and point locations are
                defined but other location types may follow.
            </xs:documentation>
        </xs:annotation>
        <xs:choice>
            <xs:element minOccurs="1" maxOccurs="1"
                name="LineLocationReference" type="openlr:LineLocationReference" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="PointLocationReference" type="openlr:PointLocationReference" />
            <xs:element minOccurs="1" maxOccurs="1"
                name="AreaLocationReference" type="openlr:AreaLocationReference" />
        </xs:choice>
    </xs:complexType>

    <xs:element name="OpenLR">
        <xs:annotation>
            <xs:documentation xml:lang="en">
                An OpenLR location reference consists of a unique identifier and a location reference as a
                XMLLocationReference, BinaryLocationReference or both. The schema accepts OpenLR tags without any
                location reference but these tags will be ignored. The XMLLocationReference shall make use of the raw data for
                any type so that no compression techniques are used (e.g. use of intervals for the description of length values).
                The BinaryLocationReference shall represent the compressed format of the location reference. If both types are
                used the location reference must represent the same location. Additional information may be added if
                necessary.
            </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element minOccurs="1" maxOccurs="1" name="LocationID"
                    type="xs:string" />
                <xs:element minOccurs="0" maxOccurs="1"
                    name="XMLLocationReference" type="openlr:XMLLocationReference" />
                <xs:element minOccurs="0" maxOccurs="1"
                    name="BinaryLocationReferences" type="openlr:BinaryLocationReferences" />
                <xs:element minOccurs="0" maxOccurs="1"
                    name="AdditionalInformation" type="openlr:AdditionalInformation" />
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

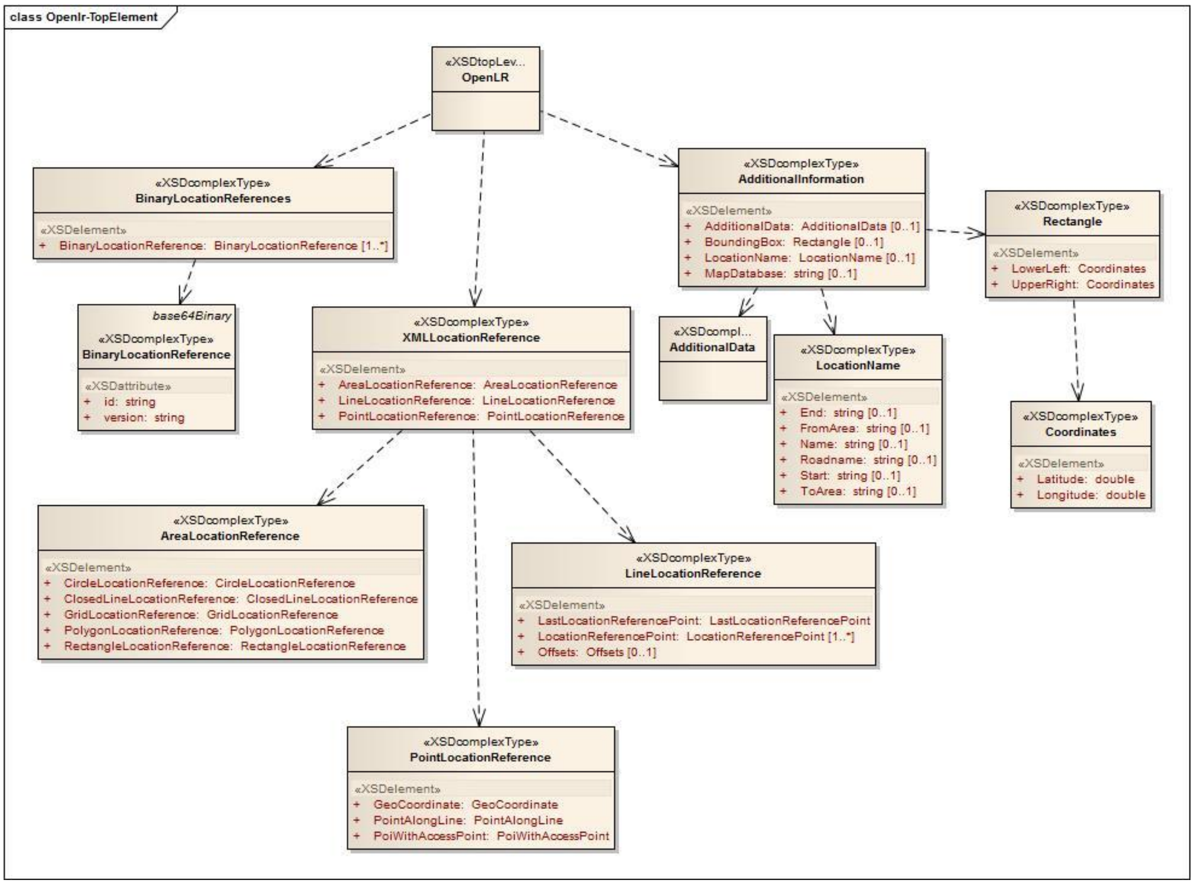
class Openlr-TopElement
«XSDtopLev... OpenLR
«XSDcomplexType» BinaryLocationReferences
«XSDelement» + BinaryLocationReference: BinaryLocationReference [1..*]
base64Binary «XSDcomplexType» BinaryLocationReference
«XSDattribute» + id: string + version: string
«XSDcomplexType» XMLLocationReference
«XSDelement» + AreaLocationReference: AreaLocationReference + LineLocationReference: LineLocationReference + PointLocationReference: PointLocationReference
«XSDcomplexType» AdditionalInformation
«XSDelement» + AdditionalData: AdditionalData [0..1] + BoundingBox: Rectangle [0..1] + LocationName: LocationName [0..1] + MapDatabase: string [0..1]
«XSDcomplexType» Rectangle
«XSDelement» + LowerLeft: Coordinates + UpperRight: Coordinates
«XSDcompl... AdditionalData
«XSDcomplexType» LocationName
«XSDelement» + End: string [0..1] + FromArea: string [0..1] + Name: string [0..1] + Roadname: string [0..1] + Start: string [0..1] + ToArea: string [0..1]
«XSDcomplexType» Coordinates
«XSDelement» + Latitude: double + Longitude: double
«XSDcomplexType» AreaLocationReference
«XSDelement» + CircleLocationReference: CircleLocationReference + ClosedLineLocationReference: ClosedLineLocationReference + GridLocationReference: GridLocationReference + PolygonLocationReference: PolygonLocationReference + RectangleLocationReference: RectangleLocationReference
«XSDcomplexType» LineLocationReference
«XSDelement» + LastLocationReferencePoint: LastLocationReferencePoint + LocationReferencePoint: LocationReferencePoint [1..*] + Offsets: Offsets [0..1]
«XSDcomplexType» PointLocationReference
«XSDelement» + GeoCoordinate: GeoCoordinate + PointAlongLine: PointAlongLine + PoiWithAccessPoint: PoiWithAccessPoint

Figure 23: XML structure overview

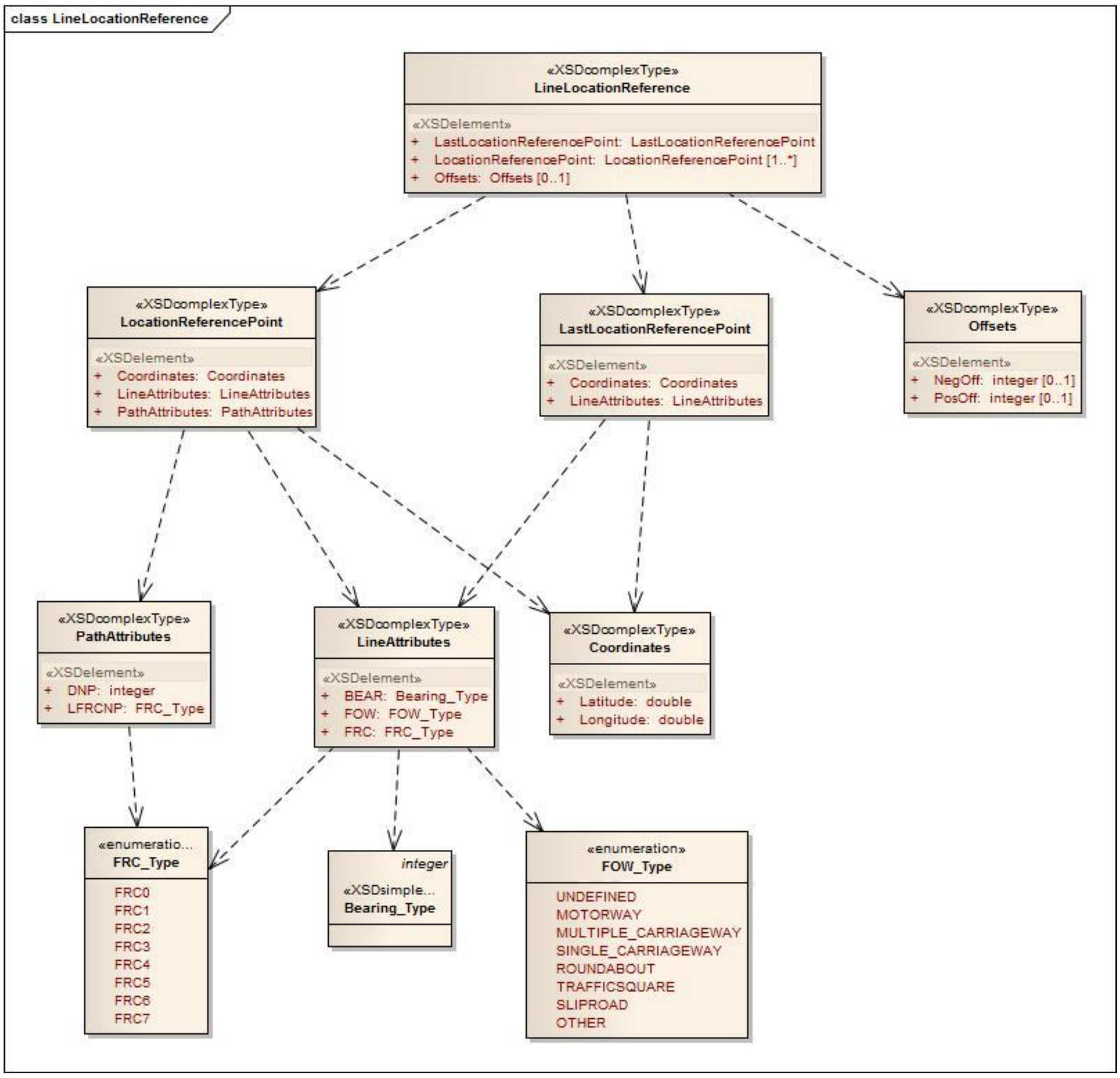
class LineLocationReference
«XSDcomplexType» LineLocationReference
«XSDelement»
+ LastLocationReferencePoint: LastLocationReferencePoint
+ LocationReferencePoint: LocationReferencePoint [1..*]
+ Offsets: Offsets [0..1]
«XSDcomplexType» LocationReferencePoint
«XSDelement»
+ Coordinates: Coordinates
+ LineAttributes: LineAttributes
+ PathAttributes: PathAttributes
«XSDcomplexType» LastLocationReferencePoint
«XSDelement»
+ Coordinates: Coordinates
+ LineAttributes: LineAttributes
«XSDcomplexType» Offsets
«XSDelement»
+ NegOff: integer [0..1]
+ PosOff: integer [0..1]
«XSDcomplexType» PathAttributes
«XSDelement»
+ DNP: integer
+ LFRCNP: FRC_Type
«XSDcomplexType» LineAttributes
«XSDelement»
+ BEAR: Bearing_Type
+ FOW: FOW_Type
+ FRC: FRC_Type
«XSDcomplexType» Coordinates
«XSDelement»
+ Latitude: double
+ Longitude: double
«enumeratio... FRC_Type
FRC0
FRC1
FRC2
FRC3
FRC4
FRC5
FRC6
FRC7
integer
«XSDsimple... Bearing_Type
«enumeration» FOW_Type
UNDEFINED
MOTORWAY
MULTIPLE_CARRIAGEWAY
SINGLE_CARRIAGEWAY
ROUNDABOUT
TRAFFICSQUARE
SLIPROAD
OTHER

Figure 24: XML structure - Line location reference

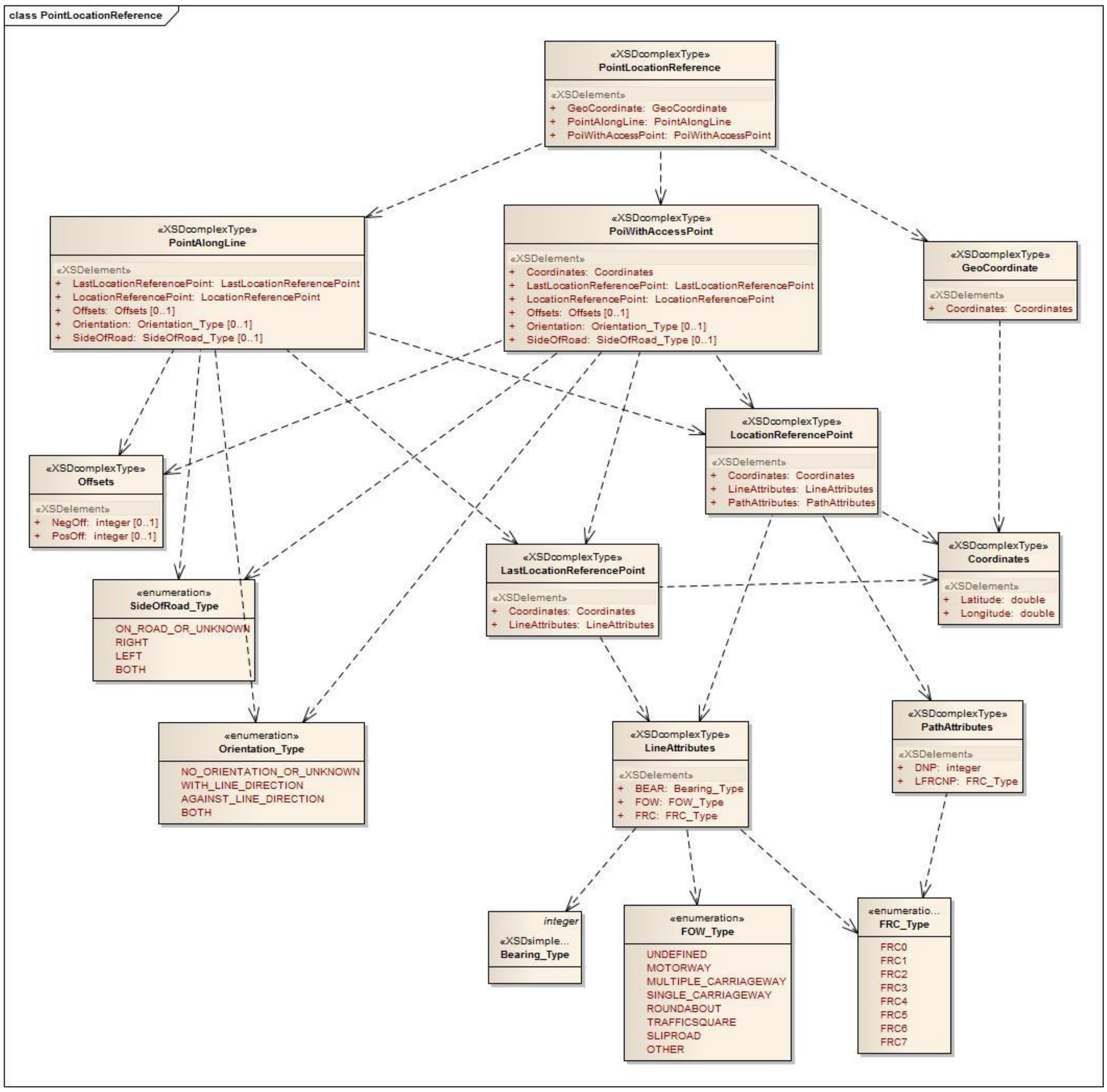
class PointLocationReference
«XSDcomplexType» PointLocationReference
«XSDelement»
+ GeoCoordinate: GeoCoordinate
+ PointAlongLine: PointAlongLine
+ PoiWithAccessPoint: PoiWithAccessPoint
«XSDcomplexType» PointAlongLine
«XSDelement»
+ LastLocationReferencePoint: LastLocationReferencePoint
+ LocationReferencePoint: LocationReferencePoint
+ Offsets: Offsets [0..1]
+ Orientation: Orientation_Type [0..1]
+ SideOfRoad: SideOfRoad_Type [0..1]
«XSDcomplexType» PoiWithAccessPoint
«XSDelement»
+ Coordinates: Coordinates
+ LastLocationReferencePoint: LastLocationReferencePoint
+ LocationReferencePoint: LocationReferencePoint
+ Offsets: Offsets [0..1]
+ Orientation: Orientation_Type [0..1]
+ SideOfRoad: SideOfRoad_Type [0..1]
«XSDcomplexType» GeoCoordinate
«XSDelement»
+ Coordinates: Coordinates
«XSDcomplexType» LocationReferencePoint
«XSDelement»
+ Coordinates: Coordinates
+ LineAttributes: LineAttributes
+ PathAttributes: PathAttributes
«XSDcomplexType» Offsets
«XSDelement»
+ NegOff: integer [0..1]
+ PosOff: integer [0..1]
«XSDcomplexType» LastLocationReferencePoint
«XSDelement»
+ Coordinates: Coordinates
+ LineAttributes: LineAttributes
«XSDcomplexType» Coordinates
«XSDelement»
+ Latitude: double
+ Longitude: double
«enumeration» SideOfRoad_Type
ON_ROAD_OR_UNKNOWN
RIGHT
LEFT
BOTH
«enumeration» Orientation_Type
NO_ORIENTATION_OR_UNKNOWN
WITH_LINE_DIRECTION
AGAINST_LINE_DIRECTION
BOTH
«XSDcomplexType» LineAttributes
«XSDelement»
+ BEAR: Bearing_Type
+ FOW: FOW_Type
+ FRC: FRC_Type
«XSDcomplexType» PathAttributes
«XSDelement»
+ DNP: integer
+ LFRCNP: FRC_Type
integer
«XSDsimple... Bearing_Type
«enumeration» FOW_Type
UNDEFINED
MOTORWAY
MULTIPLE_CARRIAGEWAY
SINGLE_CARRIAGEWAY
ROUNDABOUT
TRAFFICSQUARE
SLIPROAD
OTHER
«enumeratio... FRC_Type
FRC0
FRC1
FRC2
FRC3
FRC4
FRC5
FRC6
FRC7

Figure 25: XML structure - Point location reference

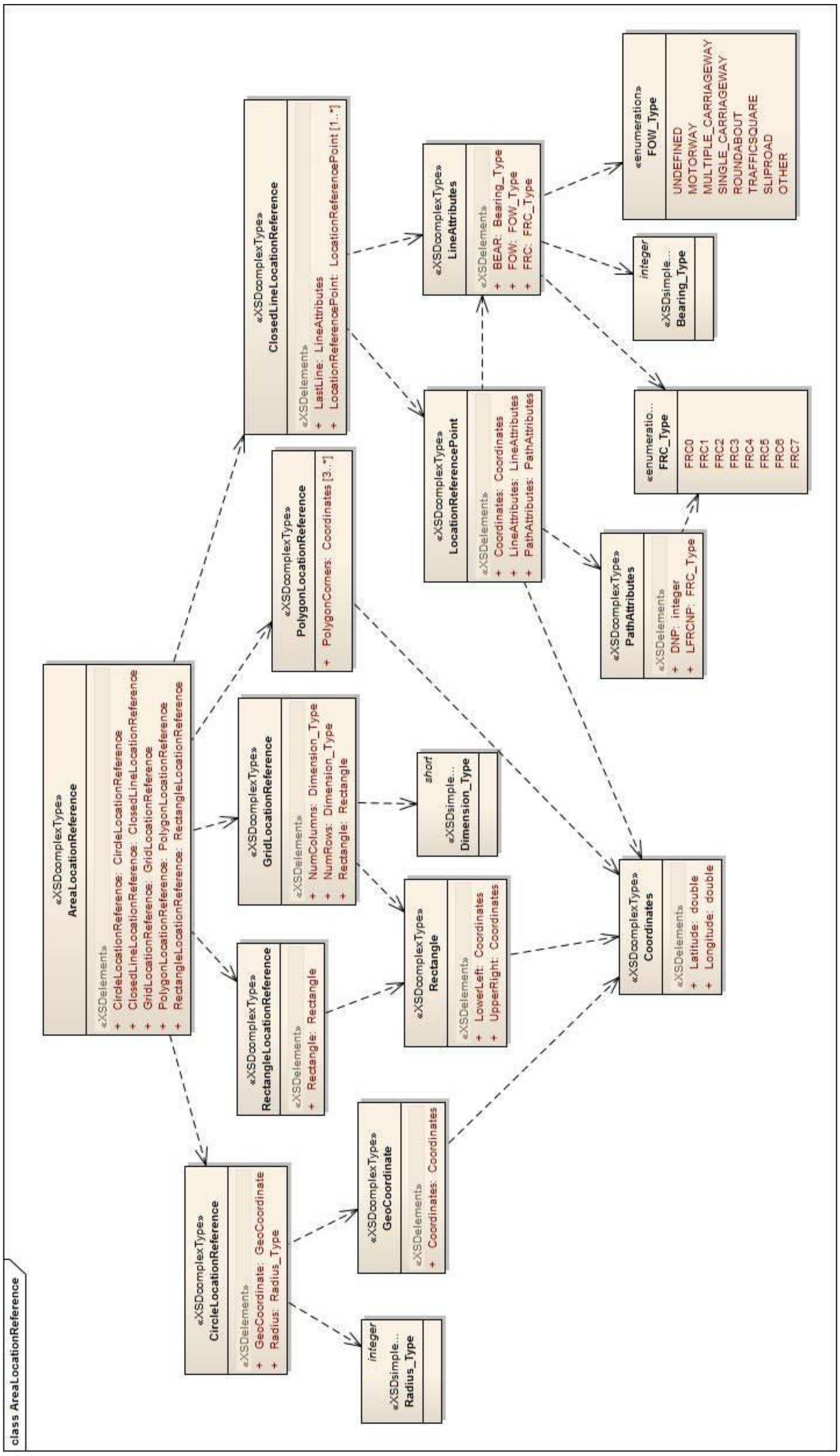

Figure 26: XML structure - Area location reference

Chapter F

Encoder

11. Encoding locations

The OpenLR™ encoder generates a map-independent location reference for a (map-dependent) location. The output of this encoder can be used to distribute location information to other parties. The decoder side will be able to resolve the location on its own map. If the encoding of such a location fails and no location reference can be created then the encoder should report on this.

Note:
This section provides recommendations how an OpenLR™ encoder might be implemented. There might be other implementations which are also compliant with the OpenLR™ standard.

11.1 Line location

The following encoding steps describe the process of generating a location reference for a line location. The location shall consist of an ordered list of connected lines and (optional) offset values.

Step – 1 *Check validity of the location and offsets to be encoded.*

A line location is valid if the following constraints are fulfilled:

- The location is a connected path.
- The location is traversable from its start to its end.

The offsets must fulfill the following constraints:

- The sum of the positive and negative offset cannot be greater than the total length of the location lines.
- Positive offset value shall be less than the length of the first line
    - Otherwise the first line can be removed from the list of location lines and the offset value must be reduced in the same way
    - This procedure shall be repeated until this constraint is fulfilled
- Negative offset value shall be less than the length of the last line
    - Otherwise the last line can be removed from the list of location lines and the offset value must be reduced in the same way
    - This procedure shall be repeated until this constraint is fulfilled

If it is intended to use the binary physical format this step should additionally calculate the maximum (minimum) latitude values along the location and adjust the maximum distance between two LR-points in Rule – 1, if necessary. The value defined in Rule – 1 is not applicable for locations above the latitude value 65.70° (or below the latitude value -65.70°).

If the location is not valid the encoder should fail.

Step – 2 *Adjust start and end node of the location to represent valid map nodes.*

Rule – 4 of the data format rules recommends to place location reference points on valid nodes. Valid nodes are such nodes where a shortest-path calculation needs to decide between several different ways. Invalid nodes, on the contrary, are such nodes where a shortest path calculation can step over (see also section 6).

Since the start and end of a location will become a location reference point these nodes need to be adjusted to valid nodes if necessary (expansion of location). The expansion shall take Rule – 1 (see section 6) into account so that the maximum distance between two location reference points will not be exceeded.

For line locations the real start of the location can then be referenced using offsets (positive offset for the start node and negative offset for the end node, respectively). See also section 11.10 for a detailed explanation of the composition of offset values.

If adjusting the nodes to valid nodes fails then the encoder could proceed with the next steps and does not need to fail.

Step – 3 *Determine coverage of the location by a shortest-path.*

The encoder calculates a shortest-path between the current start line and the end line of the location. This step will be executed until coverage of the location has been found.

If no intermediate location reference point was detected so far, the current start line is identical to the start line of the location. If an intermediate location reference point was detected in a previous step, then the line corresponding to the intermediate location reference point acts as current start line. The start line is always part of the location.

The shortest path algorithm should take the whole network or a well-defined subset of the network into account in order to calculate a shortest path between the current start and end. Additionally it should fulfill the following constraints:

- All lengths of the lines should be measured in meters and should also be converted to integer values, so that float values need to be rounded correctly.
- The search is node based and starts at the start node of the first line and ends at the end node of the last line.
- The algorithm shall return an ordered list of lines representing the calculated shortest-path.

If no shortest-path can be calculated the encoding should fail. But this should never happen as the location consists of concatenated lines.

Step – 4 *Check whether the calculated shortest-path covers the location completely. Go to step 5 if the location is not covered completely, go to step 7 if the location is covered.*

Compare the calculated shortest-path against the part of the location which is currently not covered by a shortest path in a previous step. The shortest-path covers the location if the following constraints are fulfilled:

- Every line of the location (between current start and end) needs to be part of the shortest-path.
- The order of the lines in the location (between current start and end) needs to be the same order as the lines in the shortest-path.

If one of the constraints is not fulfilled go to step 5 in order to determine a proper intermediate location reference point. If all constraints are fulfilled then go to step 7.

Step – 5 *Determine the position of a new intermediate location reference point so that the part of the location between the start of the shortest-path calculation and the new intermediate is covered completely by a shortest-path.*

If the location (between current start and end) is not fully part of the shortest-path or the order of the lines is mixed up then a proper intermediate location reference point needs to be determined. This intermediate must fulfill the following constraints:

- The shortest-path between the current start and the line indicated by the intermediate location reference point must cover the corresponding part of the location completely.
- The start node of the line indicated by the intermediate location reference point shall be positioned on a valid node (if no valid node can be determined, an invalid node may be chosen).

Step – 6 *Go to step 3 and restart shortest path calculation between the new intermediate location reference point and the end of the location.*

The remaining part of the location still needs to be covered by a shortest-path. Go to step 3 and calculate a shortest-path between the last intermediate point found and the end of the location.

Step – 7 *Concatenate the calculated shortest-paths for a complete coverage of the location and form an ordered list of location reference points (from the start to the end of the location).*

The calculated shortest-paths, if multiple were required, need to be concatenated so that the resulting location reference path covers the location completely. According to the concatenation of the path an ordered list of the location reference points needs to be formed, starting with the start of the location, followed by all intermediates, ending with the end of the location.

All relevant attributes for defining a location reference point need to be collected and calculated. This includes the functional road class, the form of way, the bearing of the corresponding line, and the distances and lowest functional road class between two subsequent location reference points.

This step may also include a check whether an intermediate LR-point is required to create a unique location reference. If the encoder uses a heuristic function during the route search it might happen that this heuristic leads the search to the end of the location but not along the location itself. The decoder might also use a heuristic function but this will lead the search towards the end of the recent part of the location and not to the end of the whole location. This difference might end up in intermediate LR-points added by the encoder but not required by the decoder. An intermediate LR-point check might determine these unneeded LR-points and remove them.

Step – 8 *Check validity of the location reference path. If the location reference path is invalid then go to step 9, if the location reference path is valid then go to step 10.*

The location reference as a list of ordered location reference points is checked for fulfilling the following constraints:

- The distance between two subsequent location reference points shall not exceed the maximum distance value as defined in the data format rules (see 6).
- The offset values must match the path between the first two (last two) location reference points. These values cannot be longer than the length of the path between the corresponding LR-points.

If the location reference does not fulfill all constraints then go to step 9, otherwise go to step 10.

The encoder may detect situations where it might be difficult for a decoder to choose the correct path. In this case additional intermediate location reference points are allowed. The intermediate must fulfill the same constraints as defined in step 5.

Step – 9 *Add a sufficient number of additional intermediate location reference points if the distance between two location reference points exceeds the maximum distance. Remove the start/ end LR-point if the positive/ negative offset value exceeds the length of the corresponding path.*

In order to fulfill the constraints in step 8 additional intermediate location reference points need to be added or removed.

If the maximum distance between two subsequent location reference points is exceeded additional location reference points shall be placed at valid nodes along the location reference path between these two location reference points. If placing on valid nodes is not possible an invalid node shall be used. Examples for such a case are lines which are longer than 15 km. In such a case the coordinates of additional LRP may be resolved from a point being less than 15 km away from the start node of that line. Such a point is directly on the line and is not represented by a node. All LRP attributes need to be calculated relative to this point. The calculated attributes need to be updated and the new intermediate location reference point must be added to the list without breaking the order from start to end.

If the positive/negative offset value is greater than the length of the path between the first/last two LR-points, then the first/last LR-point needs to be removed from the list of LR-points. The offset value needs to be adjusted accordingly.

The process of adding/removing LR-points needs to be repeated until all constraints in step 8 are fulfilled.

Step – 10 *Create physical representation of the location reference.*

The calculated location reference points (raw location reference data) need to be transformed into a physical representation (e.g. binary or XML).

11.1.1. Overview of the encoding steps

The following Table 54 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location and offsets to be encoded*
Step – 2	*Adjust start and end node of the location to represent valid map nodes*
Step – 3	*Determine coverage of the location by a shortest-path*
Step – 4	*Check whether the calculated shortest-path covers the location completely or. Go to step 5 if the location is not covered completely, go to step 7 if the location is covered*
Step – 5	*Determine the position of a new intermediate location reference point so that the part of the location between the start of the shortest-path calculation and the new intermediate is covered completely by a shortest-path.*
Step – 6	*Go to step 3 and restart shortest path calculation between the new intermediate location reference point and the end of the location.*
Step – 7	*Concatenate the calculated shortest-path(s) for a complete coverage of the location and form an ordered list of location reference points (from the start to the end of the location)*
Step – 8	*Check validity of the location reference path. If location reference path is invalid then go to step 9, if location reference path is valid then go to step 10.*
Step – 9	*Add a sufficient number of additional intermediate location reference points if the distance between two location reference points exceeds the maximum distance.*
Step – 10	*Create physical representation of the location reference*

Table 54: Encoding steps overview

11.2 Geo-coordinates location

The following encoding steps describe the process of generating a location reference for a geo-coordinate location.

Step – 1 *Check validity of the location to be encoded.*

A geo-coordinate location is valid if the following constraints are fulfilled:

- The coordinate values must be valid
  - boundaries for the longitude value: -180° <= lon <= 180°
  - boundaries for the latitude value: -90° <= lat <= 90°

If the location is not valid the encoder should fail.

Step – 2 *Create physical representation of the location reference.*

The calculated location reference point data needs to be transformed into a physical representation.

11.2.1. Overview of the encoding steps

The following Table 55 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location to be encoded*
Step – 2	*Create physical representation of the location reference*

Table 55: Encoding steps overview

11.3 Point along line location

The following encoding steps describe the process of generating a location reference for a point along line location. The location consists of a single line, one (optional) offset value and optionally information on the side of the road and orientation. The location needs to be checked whether it is valid for encoding or not. The following steps act as a guideline for the encoding process.

Step – 1 *Check validity of the location and offsets to be encoded.*

A point location is valid if the following constraints are fulfilled:

- The location contains only a single line

For all location types the offsets must fulfill the following constraints:

- Positive offset value shall be less than the length of the first line

If it is intended to use the binary physical format this step should additionally calculate the maximum (minimum) latitude values along the location and adjust the maximum distance between two LR-points in Rule – 1, if necessary. The value defined in Rule – 1 is not applicable for locations above the latitude value 65.70° (or below the latitude value -65.70°).

If the location is not valid the encoder should fail.

Step – 2 *Adjust start and end node of the location to represent valid map nodes.*

Rule – 4 of the data format rules recommends placing location reference points on valid nodes. Valid nodes are such nodes where a shortest-path calculation needs to decide between several different ways. Otherwise invalid nodes are such nodes where a shortest path calculation can step over (see also section 6).

Since the start and end of a location will become a location reference point these nodes need to be adjusted to valid nodes if necessary (expansion of location). The expansion shall take Rule – 1 (see section 6) into account so that the maximum distance between two location reference points will not be exceeded.

For point locations start and end of the line shall be expanded but only the positive offset value needs to be adjusted. See also section 11.10 for a detailed explanation of the composition of offset values.

If adjusting the nodes to valid nodes fails then the encoder could proceed with the next steps and does not need to fail.

Step – 3 *Create physical representation of the location reference.*

The calculated location reference points need to be transformed into a physical representation.

11.3.1. Overview of the encoding steps

The following Table 56 summarizes the encoding.

Step	Action
Step – 1	*Check validity of the location and offsets to be encoded*
Step – 2	*Adjust start and end node of the location to represent valid map nodes*
Step – 3	*Create physical representation of the location reference*

Table 56: Encoding steps overview

11.4 POI with access point location

The following encoding steps describe the process of generating a location reference for POI with access point locations. The location consists of a single line, a geo-coordinate, one (optional) offset value and optionally information on the side of the road and orientation. The location needs to be checked whether it is valid for encoding or not. The following steps act as a guideline for the encoding process.

Step – 1 *Check validity of the location and offsets to be encoded.*

A point location is valid if the following constraints are fulfilled:

- The coordinate values must be valid
    - boundaries for the longitude value: -180° <= lon <= 180°
    - boundaries for the latitude value: -90° <= lat <= 90°
- The location contains only a single line

For all location types the offsets must fulfill the following constraints:

- Positive offset value shall be less than the length of the first line

If it is intended to use the binary physical format this step should additionally calculate the maximum (minimum) latitude values along the location and adjust the maximum distance between two LR-points in Rule – 1, if necessary. The value defined in Rule – 1 is not applicable for locations above the latitude value 65.70° (or below the latitude value -65.70°).

If the location is not valid the encoder should fail.

Step – 2 *Adjust start and end node of the location to represent valid map nodes.*

Rule – 4 of the data format rules recommends placing location reference points on valid nodes. Valid nodes are such nodes where a shortest-path calculation needs to decide between several different ways. Otherwise invalid nodes are such nodes where a shortest path calculation can step over (see also section 6).

Since the start and end of a location will become a location reference point these nodes need to be adjusted to valid nodes if necessary (expansion of location). The expansion shall take Rule – 1 (see section 6) into account so that the maximum distance between two location reference points will not be exceeded.

For point locations start and end of the line shall be expanded but only the positive offset value needs to be adjusted. See also section 11.10 for a detailed explanation of the composition of offset values.

If adjusting the nodes to valid nodes fails then the encoder could proceed with the next steps and does not need to fail.

Step – 3 *Create physical representation of the location reference.*

The calculated location reference points need to be transformed into a physical representation.

11.4.1. Overview of the encoding steps

The following Table 57 summarizes the encoding.

Step	Action
Step – 1	*Check validity of the location and offsets to be encoded*
Step – 2	*Adjust start and end node of the location to represent valid map nodes*
Step – 3	*Create physical representation of the location reference*

Table 57: Encoding steps overview

11.5 Circle location

The following encoding steps describe the process of generating a location reference for a circle location.

Step – 1 *Check validity of the location to be encoded.*

A circle location is valid if the following constraints are fulfilled:

- The coordinate values must be valid
    - boundaries for the longitude value: -180° <= lon <= 180°
    - boundaries for the latitude value: -90° <= lat <= 90°
- The radius is a positive integer value.

If the location is not valid the encoder should fail.

Step – 2 *Create physical representation of the location reference.*

The calculated location reference point data needs to be transformed into a physical representation.

11.5.1. Overview of the encoding steps

The following Table 58 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location to be encoded*
Step – 2	*Create physical representation of the location reference*

Table 58: Encoding steps overview

11.6 Polygon location

The following encoding steps describe the process of generating a location reference for a polygon location.

Step – 1 *Check validity of the location to be encoded.*

A polygon location is valid if the following constraints are fulfilled:

- The location contains at least three coordinate pairs.
- The coordinate values must be valid
  - boundaries for the longitude value: -180° <= lon <= 180°
  - boundaries for the latitude value: -90° <= lat <= 90°
- The border of the polygon must not cross itself.
  - The border is the concatenation of all direct line connections between two subsequent coordinate pairs and the direct line connection between the last and the first coordinate pair.

If the location is not valid the encoder should fail.

Step – 2 *Create physical representation of the location reference.*

The calculated location reference point data needs to be transformed into a physical representation.

11.6.1. Overview of the encoding steps

The following Table 59 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location to be encoded*
Step – 2	*Create physical representation of the location reference*

Table 59: Encoding steps overview

11.7 Rectangle location

The following encoding steps describe the process of generating a location reference for a rectangle location.

Step – 1 *Check validity of the location to be encoded.*

A rectangle location is valid if the following constraints are fulfilled:

- All coordinate values must be valid (if a coordinate is present)
  - boundaries for the longitude value: -180° <= lon <= 180°
  - boundaries for the latitude value: -90° <= lat <= 90°
- The lower left coordinate and the upper right coordinate must differ in their longitude and latitude values.
- The lower left coordinate must be southwestern of the upper right coordinate.

If the location is not valid the encoder should fail.

Step – 2 *Create physical representation of the location reference.*

The calculated location reference point data needs to be transformed into a physical representation.

11.7.1. Overview of the encoding steps

The following Table 60 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location to be encoded*
Step – 2	*Create physical representation of the location reference*

Table 60: Encoding steps overview

11.8 Grid location

The following encoding steps describe the process of generating a location reference for a grid location.

Step – 1 *Check validity of the location to be encoded.*

A grid location is valid if the following constraints are fulfilled:

- All coordinate values must be valid
  - boundaries for the longitude value: -180° <= lon <= 180°
  - boundaries for the latitude value: -90° <= lat <= 90°
- The lower left coordinate and the upper right coordinate must differ in their longitude and latitude values.
- The lower left coordinate must be southwestern of the upper right coordinate.
- The number of columns is greater than one.
- The number of rows is greater than one.

If the location is not valid the encoder should fail.

Step – 2 *Create physical representation of the location reference.*

The calculated location reference point data needs to be transformed into a physical representation.

11.8.1. Overview of the encoding steps

The following Table 61 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location to be encoded*
Step – 2	*Create physical representation of the location reference*

Table 61: Encoding steps overview

11.9 Closed line location

The following encoding steps describe the process of generating a location reference for a closed line location. The location shall consist of an ordered list of connected lines.

Step – 1 *Check validity of the location and offsets to be encoded.*

A closed line location is valid if the following constraints are fulfilled:

- The location is a connected path.
- The location is traversable from its start to its end.
- The last line in the list is a predecessor of the start line.

If it is intended to use the binary physical format this step should additionally calculate the maximum (minimum) latitude values along the location and adjust the maximum distance between two LR-points in Rule – 1, if necessary. The value defined in Rule – 1 is not applicable for locations above the latitude value 65.70° (or below the latitude value -65.70°).

If the location is not valid the encoder should fail.

Step – 2 *Determine coverage of the location by a shortest-path.*

The encoder calculates a shortest-path between the current start line and the end line of the location. This step will be executed until coverage of the location has been found.

If no intermediate location reference point was detected so far, the current start line is identical to the start line of the location. If an intermediate location reference point was detected in a previous step, then the line corresponding to the intermediate location reference point acts as current start line. The start line is always part of the location.

The shortest path algorithm should take the whole network or a well-defined subset of the network into account in order to calculate a shortest path between the current start and end. Additionally it should fulfill the following constraints:

- All lengths of the lines should be measured in meters and should also be converted to integer values, so that float values need to be rounded correctly.
- The search is node based and starts at the start node of the first line and ends at the end node of the last line.
- The algorithm shall return an ordered list of lines representing the calculated shortest-path.

If no shortest-path can be calculated the encoding should fail. But this should never happen as the location consists of concatenated lines.

Step – 3 *Check whether the calculated shortest-path covers the location completely. Go to step 4 if the location is not covered completely, go to step 6 if the location is covered.*

Compare the calculated shortest-path against the part of the location which is currently not covered by a shortest path in a previous step. The shortest-path covers the location if the following constraints are fulfilled:

- Every line of the location (between current start and end) needs to be part of the shortest-path.
- The order of the lines in the location (between current start and end) needs to be the same order as the lines in the shortest-path.

If one of the constraints is not fulfilled go to step 4 in order to determine a proper intermediate location reference point. If all constraints are fulfilled then go to step 6.

Step – 4 *Determine the position of a new intermediate location reference point so that the part of the location between the start of the shortest-path calculation and the new intermediate is covered completely by a shortest-path.*

If the location (between current start and end) is not fully part of the shortest-path or the order of the lines is mixed up then a proper intermediate location reference point needs to be determined. This intermediate must fulfill the following constraints:

- The shortest-path between the current start and the line indicated by the intermediate location reference point must cover the corresponding part of the location completely.
- The start node of the line indicated by the intermediate location reference point shall be positioned on a valid node (if no valid node can be determined, an invalid node may be chosen).

Step – 5 *Go to step 2 and restart shortest path calculation between the new intermediate location reference point and the end of the location.*

The remaining part of the location still needs to be covered by a shortest-path. Go to step 2 and calculate a shortest-path between the last intermediate found and the end of the location.

Step – 6 *Concatenate the calculated shortest-path) for a complete coverage of the location and form an ordered list of location reference points (from the start to the end of the location).*

The calculated shortest-paths, if multiple were required, need to be concatenated so that the resulting location reference path covers the location completely. According to the concatenation of the path an ordered list of the location reference points needs to be formed, starting with the start of the location, followed by all intermediates, ending with the end of the location.

All relevant attributes for defining a location reference point need to be collected and calculated. This includes the functional road class, the form of way, the bearing of the corresponding line, and the distances and lowest functional road class between two subsequent location reference points.

This step may also include a check whether an intermediate LR-point is required to create a unique location reference. If the encoder uses a heuristic function during the route search it might happen that this heuristic leads the search to the end of the location but not along the location itself. The decoder might also use a heuristic function but this will lead the search towards the end of the recent part of the location and not to the end of the whole location. This difference might end up in intermediate LR-points added by the encoder but not required by the decoder. An intermediate LR-point check might determine these unneeded LR-points and remove them.

Step – 7 *Check validity of the location reference path. If the location reference path is invalid then go to step 8, if the location reference path is valid then go to step 9.*

The location reference as a list of ordered location reference points is checked for fulfilling the following constraints:

- The distance between two subsequent location reference points shall not exceed the maximum distance value as defined in the data format rules (see section 6).

If the location reference does not fulfill all constraints then go to step 8, otherwise go to step 9.

The encoder may detect situations where it might be difficult for a decoder to choose the correct path. In this case additional intermediate location reference points are allowed. The intermediate must fulfill the same constraints as defined in step 4.

Step – 8 *Add a sufficient number of additional intermediate location reference points if the distance between two location reference points exceeds the maximum distance.*

In order to fulfill the constraints in step 7 additional intermediate location reference points need to be added or removed.

If the maximum distance between two subsequent location reference points is exceeded additional location reference points shall be placed at valid nodes along the location reference path between these two location reference points. If placing on valid nodes is not possible an invalid node shall be used. Examples for such a case are lines which are longer than 15 km. In such a case the coordinates of additional LRP may be resolved from a point being less than 15 km away from the start node of that line. Such a point is directly on the line and is not represented by a node. All LRP attributes need to be calculated relative to this point. The calculated attributes need to be updated and the new intermediate location reference point must be added to the list without breaking the order from start to end.

The process of adding LR-points needs to be repeated until all constraints in step 7 are fulfilled.

Step – 9 *Create physical representation of the location reference.*

The calculated location reference points (raw location reference data) need to be transformed into a physical representation (e.g. binary or XML).

11.9.1. Overview of the encoding steps

The following Table 62 summarizes the encoding rules.

Step	Action
Step – 1	*Check validity of the location and offsets to be encoded*
Step – 2	*Determine coverage of the location by a shortest-path*
Step – 3	*Check whether the calculated shortest-path covers the location completely or. Go to step 5 if the location is not covered completely, go to step 7 if the location is covered*
Step – 4	*Determine the position of a new intermediate location reference point so that the part of the location between the start of the shortest-path calculation and the new intermediate is covered completely by a shortest-path.*
Step – 5	*Go to step 3 and restart shortest path calculation between the new intermediate location reference point and the end of the location.*
Step – 6	*Concatenate the calculated shortest-path(s) for a complete coverage of the location and form an ordered list of location reference points (from the start to the end of the location)*
Step – 7	*Check validity of the location reference path. If location reference path is invalid then go to step 9, if location reference path is valid then go to step 10.*
Step – 8	*Add a sufficient number of additional intermediate location reference points if the distance between two location reference points exceeds the maximum distance.*
Step – 9	*Create physical representation of the location reference*

Table 62: Encoding steps overview

11.10 Composition of offset values

The following example shows how the offset values are composed. Figure 27 shows a part of an abstract network with only 6 nodes and 5 lines. The first and the last node shall be valid nodes according to the definition of valid nodes. All intermediate nodes shall be invalid and therefore these nodes shall not be used for location reference points.

The situation a) shows the line location in red. The location is not bounded by any node. The user needs to choose the blue lines in situation b) as location lines. These location lines need to be encoded. The differences between the start node and the start of the location and the end of the location and the end node are the location offsets. Situation c) shows the positive offset at the start in dark green and the negative offset at the end in light green.

The location lines and the location offsets form the input for the OpenLR™ encoder. The encoding process starts with validating the location. If the location is valid the start and end of the location lines shall use valid nodes. The location lines will be extended so far that the start and end nodes are valid. This step results in situation d). The yellow lines indicate the expanded location lines. The additional lines are shown in situation e) where the positive expansion offset is shown in dark brown and the negative expansion offset is shown in light brown.

The expanded location will be encoded and together with the concatenated offsets compiled into a physical format. The concatenated offsets are shown in situation f) where orange indicates the positive offset and purple the negative offset.

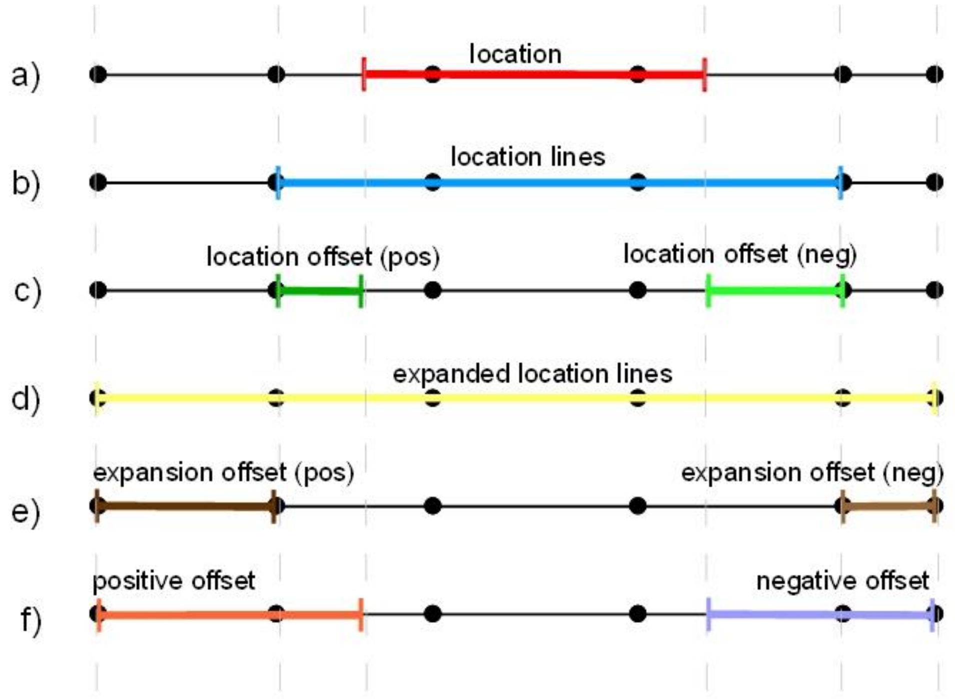

Figure 27: Offset calculation

Note:
In order to ensure a high accuracy for the offset values with the binary physical format additional intermediate location reference points may be used. These need to be placed directly after the first location reference point for the positive offset or directly before the last location reference point for the negative offset. The "LRP_length" (see Equation 8) then refers to the length between the first LRP and the inserted intermediate (inserted intermediate and last LRP respectively).

Chapter G

Decoder

12. Decoding location references

The OpenLR™ decoder retrieves a map-independent location reference which was encoded by an OpenLR™ encoder described in section 11. The decoder resolves a (map-dependent) location using its own map. This map might differ from the one used during encoding.

Note:
This section provides recommendations on how an OpenLR™ decoder might be implemented. There might be other implementations which are also compliant with the OpenLR™ standard.

12.1 Line location

The following decoding steps describe the process of resolving a location from a line location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *For each location reference point find candidate nodes*

Each location reference point contains coordinates specifying a node in the encoder map. The decoder should try to find so called candidate nodes in the decoder map whereby the coordinates of the candidate nodes are close to the coordinates of the location reference point coordinates. The straight line distance should be used to identify close-by nodes. Nodes in the decoder map which are far away from the coordinates of the location reference point should not be considered as candidate nodes in the further processing. It might happen that several candidate nodes for one location reference point exist.

If no candidate node has been determined for a location reference point the decoder should try to determine a candidate line directly. The LRP coordinate can be projected onto lines which are not far away from that coordinate. The following steps should take then the projection points into account instead of one of the start or end node of the determined lines.

Step – 3 *For each location reference point find candidate lines*

For each location reference point the decoder tries to determine lines which should fulfill the following constraints:

- The start node, end node for the last location reference point or projection point shall be close to the coordinates of the location reference point.
- The candidate lines should be outgoing lines (incoming lines for the last location reference point) of the candidate nodes or projection points determined in the previous step.
- The candidate lines should match the attributes functional road class, form of way and bearing as extracted from the physical data. Slight variances are allowed and shall be taken into account in step 4.

The direct search of lines using a projection point may also be executed even if candidate nodes are found. This might increase the number of candidate nodes but it could help to determine the correct candidate line in the next step if the nodes in the encoder and decoder map differ significantly.

If no candidate line can be found for a location reference point, the decoder should report an error and stop further processing.

Step – 4 *Rate candidate lines for each location reference point*

All candidate lines for a location reference point shall be rated according to the following criteria:

- The start node, end node for the last location reference point or projection point shall be as close as possible to the coordinates of the location reference point
- The functional road class of the candidate line should match the functional road class of the location reference point
- The form of way of the candidate line should match the form of way of the location reference point
- The bearing of the candidate line should match indicated bearing angles of the location reference point
- Slight variances in the concrete values are allowed and shall be considered in the rating function.

The candidate lines should be ordered in a way that the best matching line comes first.

Step – 5 *Determine shortest-path(s) between two subsequent location reference points*

The decoder needs to compute a shortest-path between each pair of subsequent location reference points. For each pair of location reference points suitable candidate lines must be chosen. The candidate line of the first LRPs of this pair acts as start of the shortest-path calculation. The candidate line of the second location reference point of this pair is the end of the shortest-path calculation. If the chosen lines are equal no shortest-path calculation needs to be started.

The shortest path algorithm should take the part of the network into account which contains all lines having a functional road class lower than or equal to the lowest functional road class of the first location reference point of the pair. This value might be altered if the decoder anticipates having different functional road class values than the encoder map.

Additionally the shortest-path algorithm should fulfill the following constraints:

- All lengths of the lines should be measured in meters and should also be converted to integer values, float values need to be rounded correctly.
- The search is node based and will start at the start node of the first line and will end at the end node of the last line.
- The algorithm shall return an ordered list of lines representing the calculated shortest-path.

If no shortest-path can be calculated for two subsequent location reference points, the decoder might try a different pair of candidate lines or finally fail and report an error. If a different pair of candidate lines is tried it might happen that the start line needs to be changed. In such a case this also affects the end line of the previous shortest-path and this path also needs to be re-calculated and checked again. The number of retries of shortest-path calculations should be limited in order to guarantee a fast decoding process.

Step – 6 *Check validity of the calculated shortest-path(s)*

After the shortest-path calculation the length of such a path should be checked against the distance to next point information of the first location reference point of a pair. If the length information differ too much the decoder could decide to try a different pair of candidate lines (see also Step – 5) or to fail and report an error.

Step – 7 *Concatenate shortest-path(s) to form the location and trim path according to the offsets*

If all shortest-paths are calculated and checked the decoder concatenates the shortest-path(s) according to the order of the location reference points. This concatenation shall be trimmed using the positive and negative offset of the physical data. Some hints on how to trim the location using the different offset implementations are given in section 12.10. The resulting path forms the decoder location and the decoder returns the location.

12.1.1. Overview of the decoding steps

The following Table 63 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*For each location reference point find candidate nodes*
Step – 3	*For each location reference point find candidate lines*
Step – 4	*Rate candidate lines for each location reference point*
Step – 5	*Determine shortest-path(s) between two subsequent location reference points*
Step – 6	*Check validity of the calculated shortest-path(s)*
Step – 7	*Concatenate shortest-paths to form the location and trim path according to the offsets*

Table 63: Decoding steps overview

12.2 Geo-coordinate location

The following decoding steps describe the process of resolving a location from a geo-coordinate location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *Return the geo-coordinate*

The decoder should return the geo-coordinate.

12.2.1. Overview of the decoding steps

The following Table 64 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*Return the geo-coordinate*

Table 64: Decoding steps overview

12.3 Point along line location

The following decoding steps describe the process of resolving a location from a point along line location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *For each location reference point find candidate nodes*

Each location reference point contains coordinates specifying a node in the encoder map. The decoder should try to find so called candidate nodes in the decoder map whereby the coordinates of the candidate nodes are close to the coordinates of the location reference point coordinates. The straight line distance should be used to identify close-by nodes. Nodes in the decoder map which are far away from the coordinates of the location reference point should not be considered as candidate nodes in the further processing. It might happen that several candidate nodes for one location reference point exist.

If no candidate node has been determined for a location reference point the decoder should try to determine a candidate line directly. The LRP coordinate can be projected onto lines which are not far away from that coordinate. The following steps should take then the projection points into account instead of one of the start or end node of the determined lines.

Step – 3 *For each location reference point find candidate lines*

For each location reference point the decoder tries to determine lines which should fulfill the following constraints:

- The start node (end node for the last location reference point) or projection point shall be close to the coordinates of the location reference point.
- The candidate lines should be outgoing lines (incoming lines for the last location reference point) of the candidate nodes or projection points determined in the previous step.
- The candidate lines should match the attributes functional road class, form of way and bearing as being extracted from the physical data. Slight variances are allowed and shall be taken into account in step 4.

The direct search of lines using a projection point may also be executed even if candidate nodes are found. This might increase the number of candidate nodes but it could help to determine the correct candidate line in the next step if the nodes in the encoder and decoder map differ significantly.

If no candidate line can be found for a location reference point, the decoder should report an error and stop further processing.

Step – 4 *Rate candidate lines for each location reference point*

All candidate lines for a location reference point shall be rated according to the following criteria:

- The start node (end node for the last location reference point) or projection point shall be as close as possible to the coordinates of the location reference point
- The functional road class of the candidate line should match the functional road class of the location reference point
- The form of way of the candidate line should match the form of way of the location reference point
- The bearing of the candidate line should match indicated bearing angles of the location reference point
- Slight variances in the concrete values are allowed and shall be considered in the rating function.

The candidate lines should be ordered in a way that the best matching line comes first.

Step – 5 *Determine shortest-paths between the location reference points*

The decoder needs to compute a shortest-path between the first and last location reference point.and suitable candidate lines must be chosen. The candidate line of the first LRP of this pair acts as start of the shortest-path calculation. The candidate line of the last location reference point of is the end of the shortest-path calculation. If the chosen lines are equal no shortest-path calculation needs to be started.

The shortest path algorithm should take the part of the network into account which contains all lines having a functional road class lower than or equal to the lowest functional road class of the first location reference point. This value might be altered if the decoder anticipates having different functional road class values than the encoder map.

Additionally the shortest-path algorithm should fulfill the following constraints:

- All lengths of the lines should be measured in meters and should also be converted to integer values, so that float values need to be rounded correctly.
- The search is node based and will start at the start node of the first line and will end at the end node of the last line.
- The algorithm shall return an ordered list of lines representing the calculated shortest-path.

If no shortest-path can be calculated, the decoder might try a different pair of candidate lines or finally fail and report an error. The number of retries of shortest-path calculations should be limited in order to guarantee a fast decoding process.

Step – 6 *Check validity of the calculated shortest-path*

After the shortest-path calculation the length of such a path should be checked against the distance to next point information of the first location reference point. If the length information differ too much the decoder could decide to try a different pair of candidate lines (see also Step – 5) or to fail and report an error.

Step – 7 *Determine point on line according to the positive offset and resolve information on side of road and orientation*

The positive offset value must now be used to determine the point on the line. Together with this point in question the decoder returns the information on the side of road and the orientation.

12.3.1. Overview of the decoding steps

The following Table 65 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*For each location reference point find candidate nodes*
Step – 3	*For each location reference point find candidate lines*
Step – 4	*Rate candidate lines for each location reference point*
Step – 5	*Determine shortest-path between the location reference points*
Step – 6	*Check validity of the calculated shortest-path*
Step – 7	*Determine point on line according to the positive offset and resolve information on side of road and orientation*

Table 65: Decoding steps overview

12.4 POI with access point location

The following decoding steps describe the process of resolving a location from POI with access point location references. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *For each location reference point find candidate nodes*

Each location reference point contains coordinates specifying a node in the encoder map. The decoder should try to find so called candidate nodes in the decoder map whereby the coordinates of the candidate nodes are close to the coordinates of the location reference point coordinates. The airline distance should be used to identify close-by nodes. Nodes in the decoder map which are far away from the location reference point coordinates should not be considered as candidate nodes in the further processing. It might happen that several candidate nodes for one location reference point exist.

Step – 3 *For each location reference point find candidate lines*

For each location reference point the decoder tries to determine lines which should fulfill the following constraints:

- The start node, end node for the last location reference point or projection point shall be close to the coordinates of the location reference point.
- The candidate lines should be outgoing lines (incoming lines for the last location reference point) of the candidate nodes or projection points determined in the previous step.
- The candidate lines should match the attributes functional road class, form of way and bearing as being extracted from the physical data. Slight variances are allowed and shall be taken into account in step 4.

If no candidate node has been determined for a location reference point the decoder should try to determine a candidate line directly. The LRP coordinate can be projected onto lines which are not far away from that coordinate. The following steps should take then the projection points into account instead of one of the start or end node of the determined lines.

The direct search of lines using a projection point may also be executed even if candidate nodes are found. This might increase the number of candidate nodes but it could help to determine the correct candidate line in the next step if the nodes in the encoder and decoder map differ significantly.

If no candidate line can be found for a location reference point, the decoder should report an error and stop further processing.

Step – 4 *Rate candidate lines for each location reference point*

All candidate lines for a location reference point shall be rated according to the following criteria:

- The start node, end node for the last location reference point or projection point shall be as close as possible to the coordinates of the location reference point
- The functional road class of the candidate line should match the functional road class of the location reference point
- The form of way of the candidate line should match the form of way of the location reference point

- The bearing of the candidate line should match indicated bearing angles of the location reference point
- Slight variances in the concrete values are allowed and shall be considered in the rating function.

The candidate lines should be ordered in a way that the best matching line comes first.

Step – 5 *Determine shortest-paths between the location reference points*

The decoder needs to compute a shortest-path between the first and last location reference point.and suitable candidate lines must be chosen. The candidate line of the first LRP of this pair acts as start of the shortest-path calculation. The candidate line of the last location reference point of is the end of the shortest-path calculation. If the chosen lines are equal no shortest-path calculation needs to be started.

The shortest path algorithm should take the part of the network into account which contains all lines having a functional road class lower than or equal to the lowest functional road class of the first location reference point. This value might be altered if the decoder anticipates having different functional road class values than the encoder map.

Additionally the shortest-path algorithm should fulfill the following constraints:
- All lengths of the lines should be measured in meters and should also be converted to integer values, so that float values need to be rounded correctly.
- The search is node based and will start at the start node of the first line and will end at the end node of the last line.
- The algorithm shall return an ordered list of lines representing the calculated shortest-path.

If no shortest-path can be calculated, the decoder might try a different pair of candidate lines or finally fail and report an error. The number of retries of shortest-path calculations should be limited in order to guarantee a fast decoding process.

Step – 6 *Check validity of the calculated shortest-path*

After the shortest-path calculation the length of such a path should be checked against the distance to next point information of the first location reference point. If the length information differ too much the decoder could decide to try a different pair of candidate lines (see also Step – 5) or to fail and report an error.

Step – 7 *Determine access point on line according to the positive offset and resolve information on side of road and orientation. Resolve the coordinates of the POI.*

The positive offset value must now be used to determine the access point on the line. Then the coordinates of the POI must be resolved. Together with these points in question the decoder returns the information on the side of road and the orientation.

12.4.1. Overview of the decoding steps

The following Table 66 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*For each location reference point find candidate nodes*
Step – 3	*For each location reference point find candidate lines*
Step – 4	*Rate candidate lines for each location reference point*
Step – 5	*Determine shortest-path between the location reference points*
Step – 6	*Check validity of the calculated shortest-path*

Step – 7	*Determine access point on line according to the positive offset and resolve information on side of road and orientation. Resolve the coordinates of the POI.*

Table 66: Decoding steps overview

12.5 Circle location

The following decoding steps describe the process of resolving a location from a circle location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *(Optional) Resolve lines being fully covered by the area or partly covered by the area*

The decoder may resolve all lines which are fully covered by the area and may also determine the lines which are partly covered by the area. This step is optional as the computational effort may be high depending on the size of the area and the road network.

Step – 3 *Return the circle*

The decoder should return the circle by specifying the center point and the radius. Optionally the affected lines can be returned.

12.5.1. Overview of the decoding steps

The following Table 67 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*(Optional) Resolve lines being fully covered by the area or partly covered by the area*
Step – 3	*Return the circle*

Table 67: Decoding steps overview

12.6 Polygon location

The following decoding steps describe the process of resolving a location from a polygon location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference.It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *(Optional) Resolve lines being fully covered by the area or partly covered by the area*

The decoder may resolve all lines which are fully covered by the area and may also determine the lines which are partly covered by the area. This step is optional as the computational effort may be high depending on the size of the area and the road network.

Step – 3 *Return the polygon*

The decoder should return the polygon by specifying the sequence of corner points. Optionally the affected lines can be returned.

12.6.1. Overview of the decoding steps

The following Table 68 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*(Optional) Resolve lines being fully covered by the area or partly covered by the area*
Step – 3	*Return the polygon*

Table 68: Decoding steps overview

12.7 Rectangle location

The following decoding steps describe the process of resolving a location from a rectangle location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *(Optional) Resolve lines being fully covered by the area or partly covered by the area*

The decoder may resolve all lines which are fully covered by the area and may also determine the lines which are partly covered by the area. This step is optional as the computational effort may be high depending on the size of the area and the road network.

Step – 3 *Return the rectangle*

The decoder should return the rectangle by specifying the lower left and upper right coordinate spanning the rectangle area. Optionally the affected lines can be returned.

12.7.1. Overview of the decoding steps

The following Table 69 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*(Optional) Resolve lines being fully covered by the area or partly covered by the area*
Step – 3	*Return the rectangle*

Table 69: Decoding steps overview

12.8 Grid location

The following decoding steps describe the process of resolving a location from a grid location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *(Optional) Resolve lines being fully covered by the area or partly covered by the area*

The decoder may resolve all lines which are fully covered by the area and may also determine the lines which are partly covered by the area. This step is optional as the computational effort may be high depending on the size of the area and the road network.

Step – 3 *Return the grid*

The decoder should return the grid by specifying the lower left and upper right coordinates spanning the basis rectangle and the number of rows and number of columns. Optionally the affected lines can be returned.

12.8.1. Overview of the decoding steps

The following Table 70 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*(Optional) Resolve lines being fully covered by the area or partly covered by the area*
Step – 3	*Return the grid*

Table 70: Decoding steps overview

12.9 Closed line location

The following decoding steps describe the process of resolving a location from a closed line location reference. The location reference comes in a physical representation. The decoder resolves a location in the decoder map or it reports on errors if no location can be resolved. The following steps act as a guideline for the decoding process.

Step – 1 *Decode physical data and check its validity*

The decoder receives a physical representation of a location reference. It has to check whether the data is valid. If a check fails the decoder should report an error and stop the decoding process. The decoder should also extract the physical data into location reference points.

Step – 2 *For each location reference point find candidate nodes*

Each location reference point contains coordinates specifying a node in the encoder map. The decoder should try to find so called candidate nodes in the decoder map whereby the coordinates of the candidate nodes are close to the coordinates of the location reference point coordinates. The airline distance should be used to identify close-by nodes. Nodes in the decoder map which are far away from the location reference point coordinates should not be considered as candidate nodes in the further processing. It might happen that several candidate nodes for one location reference point exist.

If no candidate node has been determined for a location reference point the decoder should try to determine a candidate line directly. The LRP coordinate can be projected onto lines which are not far away from that coordinate. The following steps should take then the projection points into account instead of one of the start or end node of the determined lines.

Step – 3 *For each location reference point find candidate lines*

For each location reference point the decoder tries to determine lines which should fulfill the following constraints:

- The start node (end node for the last location reference point) or projection point shall be close to the coordinates of the location reference point.
- The candidate lines should be outgoing lines (incoming lines for the last location reference point) of the candidate nodes or projection points determined in the previous step.
- The candidate lines should match the attributes functional road class, form of way and bearing as being extracted from the physical data. Slight variances are allowed and shall be taken into account in step 4.

The direct search of lines using a projection point may also be executed even if candidate nodes are found. This might increase the number of candidate nodes but it could help to determine the correct candidate line in the next step if the nodes in the encoder and decoder map differ significantly.

If no candidate line can be found for a location reference point, the decoder should report an error and stop further processing.

Step – 4 *Rate candidate lines for each location reference point*

All candidate lines for a location reference point shall be rated according to the following criteria:

- The start node (end node for the last location reference point) or projection point shall be as close as possible to the coordinates of the location reference point
- The functional road class of the candidate line should match the functional road class of the location reference point
- The form of way of the candidate line should match the form of way of the location reference point
- The bearing of the candidate line should match indicated bearing angles of the location reference point
- Slight variances in the concrete values are allowed and shall be considered in the rating function.

The candidate lines should be ordered in a way that the best matching line comes first.

Step – 5 *Determine shortest-path(s) between two subsequent location reference points*

The decoder needs to compute a shortest-path between each pair of subsequent location reference points. For each pair of location reference points suitable candidate lines must be chosen. The candidate line of the first LRPs of this pair acts as start of the shortest-path calculation. The candidate line of the second location reference point of this pair is the end of the shortest-path calculation. If the chosen lines are equal no shortest-path calculation needs to be started.

The shortest path algorithm should take the part of the network into account which contains all lines having a functional road class lower than or equal to the lowest functional road class of the first location reference point of the pair. This value might be altered if the decoder anticipates having different functional road class values than the encoder map.

Additionally the shortest-path algorithm should fulfill the following constraints:

- All lengths of the lines should be measured in meters and should also be converted to integer values, so that float values need to be rounded correctly.
- The search is node based and will start at the start node of the first line and will end at the end node of the last line.
- The algorithm shall return an ordered list of lines representing the calculated shortest-path.

If no shortest-path can be calculated for two subsequent location reference points, the decoder might try a different pair of candidate lines or finally fail and report an error. If trying a different pair of candidate lines it might happen that the start line needs to be changed. In such a case this also affects the end line of the previous shortest-path and this path also needs to be re-calculated and checked again. The number of retries of shortest-path calculations should be limited in order to guarantee a fast decoding process.

Step – 6 *Check validity of the calculated shortest-path(s)*

After the shortest-path calculation the length of such a path should be checked against the distance to next point information of the first location reference point of a pair. If the length information differ too much the decoder could decide to try a different pair of candidate lines (see also Step – 5) or to fail and report an error.

Step – 7 *Concatenate shortest-paths to form the location*

If all shortest-paths are calculated and checked the decoder concatenates the shortest-paths according to the order of the location reference points.

Step – 8 *(Optional) Resolve lines being fully covered by the area or partly covered by the area*

The decoder may resolve all lines which are fully covered by the area and may also determine the lines which are partly covered by the area. This step is optional as the computational effort may be high depending on the size of the area and the road network.

12.9.1. Overview of the decoding steps

The following Table 71 summarizes the decoding steps.

Step	Action
Step – 1	*Decode physical data and check its validity*
Step – 2	*For each location reference point find candidate nodes*
Step – 3	*For each location reference point find candidate lines*
Step – 4	*Rate candidate lines for each location reference point*
Step – 5	*Determine shortest-path(s) between two subsequent location reference points*
Step – 6	*Check validity of the calculated shortest-path(s)*
Step – 7	*Concatenate shortest-paths to form the location*
Step – 8	*(Optional) Resolve lines being fully covered by the area or partly covered by the area*

Table 71: Decoding steps overview

12.10 Decoding offsets

Decoding the offsets and trimming the location depend on the physical format. The XML format allows to transmit the real values, the binary format version 3 defines a relative value and needs the distance between the first two location reference points for the calculation of the positive offset (last two location reference points for negative offset).

The offset values defined in XML are more accurate than the values defined in the binary format. The decoder implementation must take these differences into account.

Chapter H

Examples

13. OpenLR™ examples

OpenLR™ shall now be explained using abstract examples. The examples use two different maps and they execute the encoder steps as described in section 11 outline the binary and XML representations and they execute the decoder steps as described in section 12.

Section 13.1 explains an example of encoding and decoding a line location, section 13.2 explains an example of encoding and decoding the different types of point locations and section 13.3 shows how area locations are encoded and decoded.

13.1 Line Location

13.1.1. Example setup

The encoder map is shown in Figure 28 and consists of 15 nodes and 23 lines (two-way lines are counted twice). The nodes are numbered from 1 to 15. The necessary line attributes are shown beside every line using the format: <FRC>, <FOW>, <Length in meters>. The arrows indicate the driving direction for each line. The coordinates are chosen exemplary and do not correspond to any real world objects!

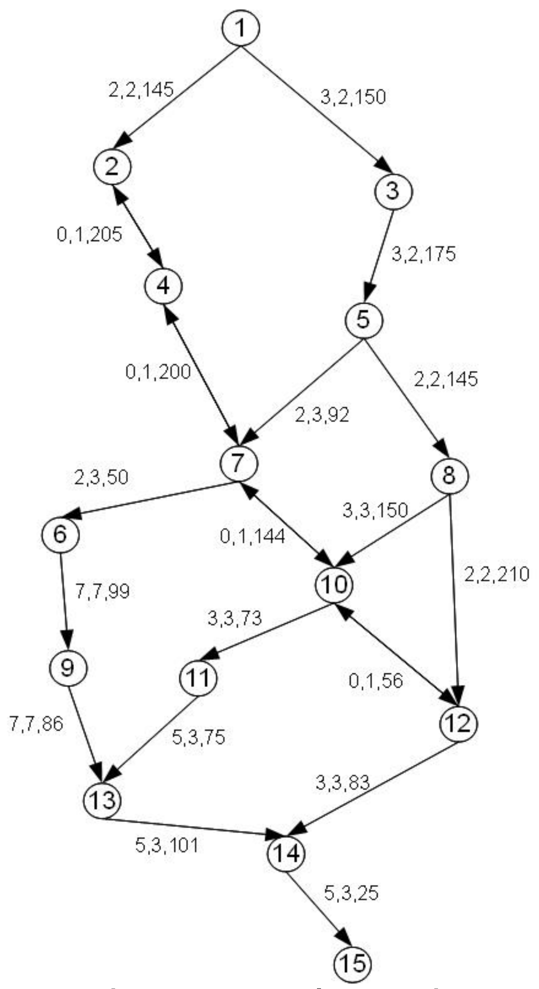

Figure 28: Example network

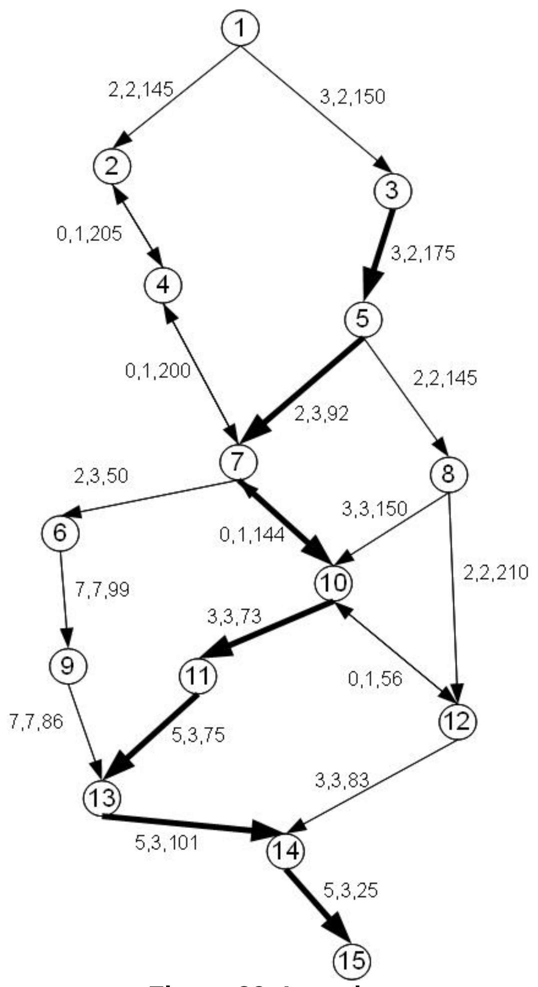

Figure 29: Location

The location to be encoded is shown in Figure 29 using bold lines. The location starts at node 3 and continues over the nodes 5, 7, 10, 11, 13, 14, and ends at node 15. Its total length in the encoder map is 685 meters. The ordered list of lines and the map to be used during encoding serves as input for the OpenLR™ encoder.

13.1.2. Encoding

In step 1 of the encoding process the location will first be checked for validity. Since the location is connected and drivable and all functional road classes along the location are between 0 and 7, this location is valid. Given that turn restrictions are omitted from the map data used for this example, the encoder can ignore this check.

The encoder step 2 checks the start and end node of the location being valid nodes according to the data format rules in section 6. The end node 15 has only one incoming line and is therefore valid.

The start node 3 also has two incident lines but here it is one outgoing and one incoming line. Therefore this node is not valid and the encoder searches for a valid node outside the location. The encoder will find node 1 to be a valid node and it also expands the location uniquely. Node 1 is chosen as the new start node for the location reference and there will be a positive offset of 150 meters. The total length of the location reference path results in 835 meters.

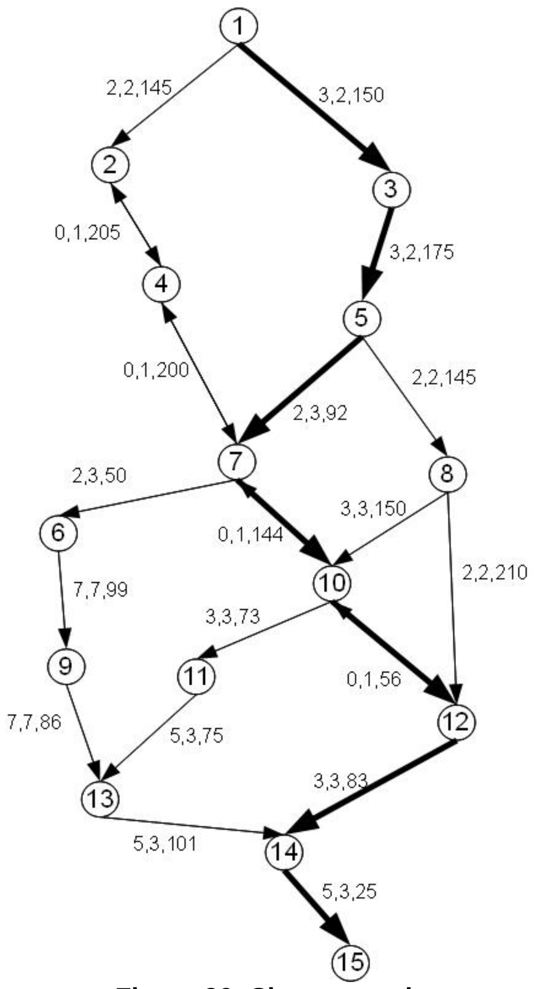

Figure 30: Shortest path

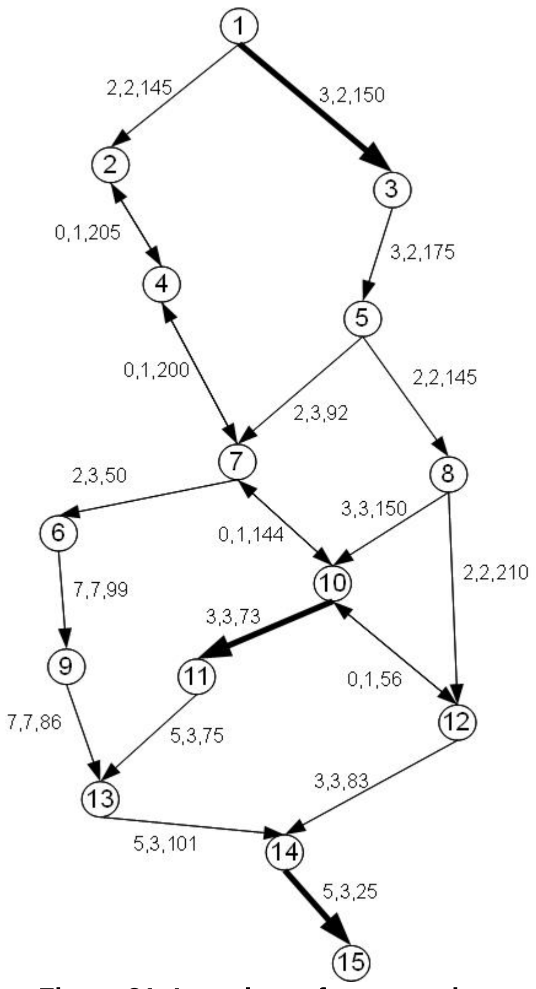

Figure 31: Location reference points

The encoder proceeds with step 3 and calculates a shortest-path between the start line (line between nodes 3 and 5) and the end line (line between nodes 14 and 15) of the location. The line between the nodes 1 and 3 may be ignored because this line was added during the extension phase. The resulting shortest-path (including the extended line) is outlined in Figure 30 using bold lines. The shortest-path has a length of 725 meters (without the extended line: 675 meters).

Step 4 of the encoding process now checks whether the location is covered by the calculated shortest-path. It will detect that this is not the case and there is a deviation after node 10. Continuing with step 5 the encoder will determine the line from node 10 to 11 to become a new intermediate location reference point. Node 10 is a valid node since it cannot be stepped over during route search and the shortest-path up to this line covers the corresponding part of the location completely. The length of the location being covered after this first shortest-path calculation is 561 meters.

The following step 6 prepares the route calculation in order to determine a shortest-path for the remaining part of the location (from node 10 over 11 and 14 to 15). The shortest-path calculation will therefore start at the line from 10 to 11 and ends at the line from 14 to 15.

Step 3 will determine a shortest path (length: 274 meters) between 10 and 15 and step 4 will return that the location is now completely covered by the calculated shortest paths.

In step 7 the location reference path will be composed of the two shortest-paths and the ordered list of location reference points will be formed. Figure 31 shows the lines being selected for the location reference points. The first LRP points to the line from node 1 to 3 and indicates the start of the location reference path, the second LRP points to the line from node 10 to 11 and this line was necessary to avoid the deviation from the location. The last LRP points to the line from node 14 to 15 and indicates the end of the location reference path.

The following step 8 checks the validity of the location reference. Since all lengths between two subsequent location reference points are less than the maximum distance, the location reference emphasizes as being valid.

The last step of the encoding process (step 10) generates the binary and XML representation of the location reference.

13.1.3. Binary location reference

The location reference consists of three location reference points and Table 72 shows the coordinates for the nodes 1, 10 and 15. These nodes are the corresponding nodes to the location reference points. In preparation of the binary format this table also shows the relative coordinates. The node 1 corresponds to the location reference point 1 and will have coordinates in absolute format. Node 10 corresponding to location reference point 2 will have relative coordinates to the location reference point 1. Node 15 corresponding to location reference point 3 will also have relative coordinates but now referencing to location reference point 2.

Node ID	LRP index	Longitude	Latitude	Relative longitude	Relative latitude
1	1	6.12683°	49.60851°	--	--
10	2	6.12838°	49.60398°	155	-453
15	3	6.12817°	49.60305°	-21	-93

Table 72: Example coordinates

The offsets being calculated in step 2 of the encoding process are shown in Table 73. In the binary data only the positive offset will appear because the negative offset is 0 and a missing offset will be treated as 0. The length between the first two location reference points is 561m.

Field	Value	%
positive offset	150	26,74
negative offset	0	--

Table 73: Example offset values

Table 74 collects the relevant data for each location reference point. This includes the functional road class, the form of way and the bearing of the corresponding line. The needed information about the path between two subsequent location reference points is also shown (lowest functional road class and distance to the next location reference point).

LRP index	FRC	FOW	BEAR	LFRCNP	DNP

1	FRC3	MULTIPLE_CARRIAGEWAY	135°	FRC3	561
2	FRC3	SINGLE_CARRIAGEWAY	227°	FRC5	274
3	FRC5	SINGLE_CARRIAGEWAY	290°	--	--

Table 74: Location reference points determined during encoding

These tables above hold all relevant information for creating the binary data. The following tables outline the binary data according to section 8:

• Status byte:	see Table 75
• LRP 1:	see Table 76 to Table 80
• LRP 2	see Table 81 to Table 85
• LRP 3	see Table 86 to Table 89
• Offset	see Table 90

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	0	0	0	1	0	1	1

Table 75: Binary example: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	1	0	1	1	0	1	1

Table 76: Binary example: LRP 1 – absolute longitude

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	1	1	1	1	0	1	0	1

Table 77: Binary example: LRP1 – absolute latitude

Bit	7	6	5	4	3	2	1	0
Description	RFU	RFU	FRC			FOW		
Value	0	0	0	1	1	0	1	0

Table 78: Binary example: LRP1 – attribute 1

Bit	7	6	5	4	3	2	1	0
Description	LFRCNP			Bearing				
Value	0	1	1	0	1	1	0	0

Table 79: Binary example: LRP1 – attribute 2

Bit	7	6	5	4	3	2	1	0
Description	DNP							
Value	0	0	0	0	1	0	0	1

Table 80: Binary example: LRP1 – attribute 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1

Table 81: Binary example: LRP2 – relative longitude

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	0	0	0	1	1	1	0	1	1

Table 82: Binary example: LRP2 – relative latitude

Bit	7	6	5	4	3	2	1	0
Description	RFU	RFU	FRC			FOW		
Value	0	0	0	1	1	0	1	1

Table 83: Binary example: LRP2 – attribute 1

Bit	7	6	5	4	3	2	1	0
Description	LFRCNP			Bearing				
Value	1	0	1	1	0	1	0	0

Table 84: Binary example: LRP2 – attribute 2

Bit	7	6	5	4	3	2	1	0
Description	DNP							
Value	0	0	0	0	0	1	0	0

Table 85: Binary example: LRP2 – attribute 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1

Table 86: Binary example: LRP3 – relative longitude

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	0	1	0	0	0	1	1

Table 87: Binary example: LRP3 – relative latitude

Bit	7	6	5	4	3	2	1	0
Description	RFU	RFU	FRC			FOW		
Value	0	0	1	0	1	0	1	1

Table 88: Binary example: LRP3 – attribute 1

Bit	7	6	5	4	3	2	1	0
Description	RFU	PoffF	NoffF	Bearing				
Value	0	1	0	1	1	0	0	1

Table 89: Binary example: LRP3 – attribute 4

Bit	7	6	5	4	3	2	1	0
Description	POFF							
Value	0	1	0	0	0	1	0	0

Table 90: Binary example: positive Offset

The full binary data stream will have a length of 24 bytes and consists of the following (ordered as bytes from left to right and top to down, see integer values in parentheses):

00001010(11)	00000100(4)	01011011(91)	01011011(91)	00100011(35)	01000110(70)
11110101 (-11)	00011010(26)	01101100(108)	00001001(9)	00000000(0)	10011011(-101)
11111110(-2)	00111011(59)	00011011(27)	10110100(-76)	00000100(4)	11111111(-1)
11101011(-21)	11111111(-1)	10100011(-93)	00101011(43)	01011001(89)	01000100(68)

A Base64 encoded representation of the binary data is the following string:

CwRbWyNG9RpsCQCb/jsbtAT/6/+jK1lE

13.1.4. XML location reference

The location reference consists of three location reference points in total. The XML format therefore contains two location reference points and a last location reference points. The offset field contains only the positive offset value as there is no negative offset.

The XML representation also contains the Base64-encoded binary representation of the location reference.

```
<?xml version="1.0" encoding="UTF-8"?>
<openlr:OpenLR xmlns:openlr="http://www.openlr.org/openlr"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.openlr.org/openlr openlr.xsd ">
  <openlr:LocationID>ExampleLocation</openlr:LocationID>
  <openlr:XMLLocationReference>
    <openlr:LineLocationReference>
      <openlr:LocationReferencePoint>
        <openlr:Coordinates>
          <openlr:Longitude>6.12683</openlr:Longitude>
          <openlr:Latitude>49.60851</openlr:Latitude>
        </openlr:Coordinates>
        <openlr:LineAttributes>
          <openlr:FRC>FRC3</openlr:FRC>
          <openlr:FOW>MULTIPLE_CARRIAGEWAY</openlr:FOW>
          <openlr:BEAR>135</openlr:BEAR>
        </openlr:LineAttributes>
        <openlr:PathAttributes>
          <openlr:LFRCNP>FRC3</openlr:LFRCNP>
          <openlr:DNP>561</openlr:DNP>
        </openlr:PathAttributes>
      </openlr:LocationReferencePoint>
      <openlr:LocationReferencePoint>
        <openlr:Coordinates>
          <openlr:Longitude>6.12838</openlr:Longitude>
          <openlr:Latitude>49.60398</openlr:Latitude>
        </openlr:Coordinates>
        <openlr:LineAttributes>
          <openlr:FRC>FRC3</openlr:FRC>
          <openlr:FOW>SINGLE_CARRIAGEWAY</openlr:FOW>
          <openlr:BEAR>227</openlr:BEAR>
        </openlr:LineAttributes>
        <openlr:PathAttributes>
          <openlr:LFRCNP>FRC5</openlr:LFRCNP>
          <openlr:DNP>274</openlr:DNP>
        </openlr:PathAttributes>
      </openlr:LocationReferencePoint>
      <openlr:LastLocationReferencePoint>
        <openlr:Coordinates>
          <openlr:Longitude>6.12817</openlr:Longitude>
          <openlr:Latitude>49.60305</openlr:Latitude>
        </openlr:Coordinates>
        <openlr:LineAttributes>
          <openlr:FRC>FRC5</openlr:FRC>
          <openlr:FOW>SINGLE_CARRIAGEWAY</openlr:FOW>
          <openlr:BEAR>290</openlr:BEAR>
        </openlr:LineAttributes>
      </openlr:LastLocationReferencePoint>
      <openlr:Offsets>
        <openlr:PosOff>150</openlr:PosOff>
      </openlr:Offsets>
    </openlr:LineLocationReference>
  </openlr:XMLLocationReference>
  <openlr:BinaryLocationReferences>
    <openlr:BinaryLocationReference id="BinaryExample" version="3">
      CgRbWyNG9BpsCQCb/jsbtAT/6/+jK1kC
    </openlr:BinaryLocationReference>
  </openlr:BinaryLocationReferences>
  <openlr:AdditionalInformation>
    <openlr:BoundingBox>
      <openlr:LowerLeft>
        <openlr:Longitude>6.12601</openlr:Longitude>
        <openlr:Latitude>49.60243</openlr:Latitude>
      </openlr:LowerLeft>
      <openlr:UpperRight>
        <openlr:Longitude>6.12951</openlr:Longitude>
        <openlr:Latitude>49.60891</openlr:Latitude>
      </openlr:UpperRight>
    </openlr:BoundingBox>
    <openlr:MapDatabase>ExampleMap Version 0.1</openlr:MapDatabase>
    <openlr:AdditionalData/>
  </openlr:AdditionalInformation>
</openlr:OpenLR>
```

13.1.5. Decoding

This section outlines the decoding of the location reference being prepared in the sections above.

The step 1 of the decoding process extracts the physical data (binary format) and checks the validity of the location reference. The information extracted from the binary data will not be as accurate as before creating the binary stream. Due to the usage of intervals for the bearing and the distance to next point the concrete value cannot be extracted but a small interval containing the concrete value. The offset value is chosen as the mid point of the interval (calculation: 0.390625 * 68 * 1.5 = 26.76).

The information being extracted from the binary data example is shown in Table 91, Table 92 and Table 93.

LRP index	Longitude	Latitude
1	6.12682°	49.60850°
2	6.12838°	49.60397°
3	6.12817°	49.60304°

Table 91: Decoded coordinates

LRP index	FRC	FOW	Bearing	LFRCNP	DNP
1	FRC3	MULTIPLE_CARRIAGEWAY	135.00° - 146.25°	FRC3	527.4m – 586.0m
2	FRC3	SINGLE_CARRIAGEWAY	225.00° - 236.25°	FRC5	234.4m – 293.0m
3	FRC5	SINGLE_CARRIAGEWAY	281.25° - 292.50°	--	0m

Table 92: Decoded LRP information

Offset	Value
Positive offset	26,76%
Negative offset	- no offset available -

Table 93: Decoded offset information

This information is sufficient to resolve the location on the decoder map shown in Figure 32. The map consists of 17 nodes and 26 lines (two-way lines are counted twice).

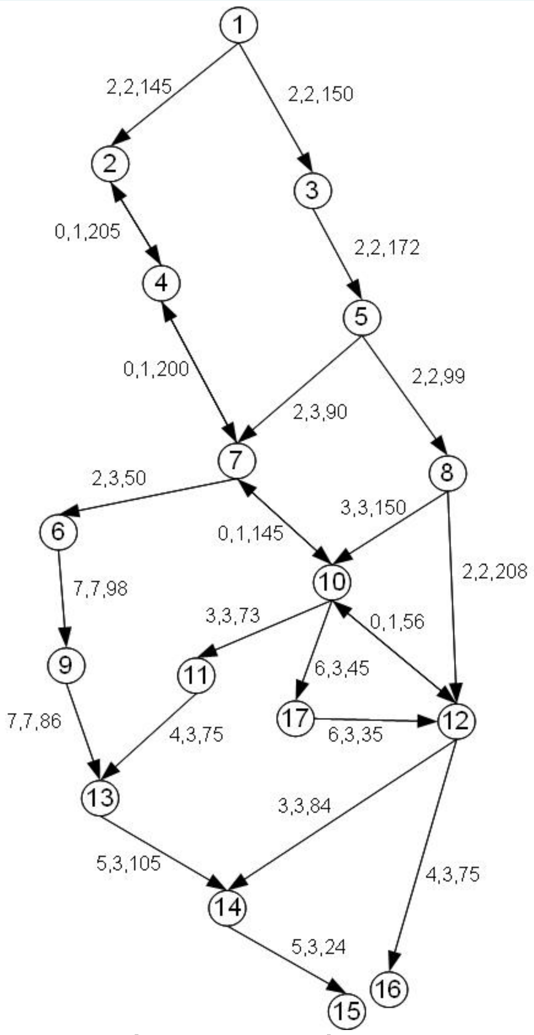
Figure 32: Decoder map

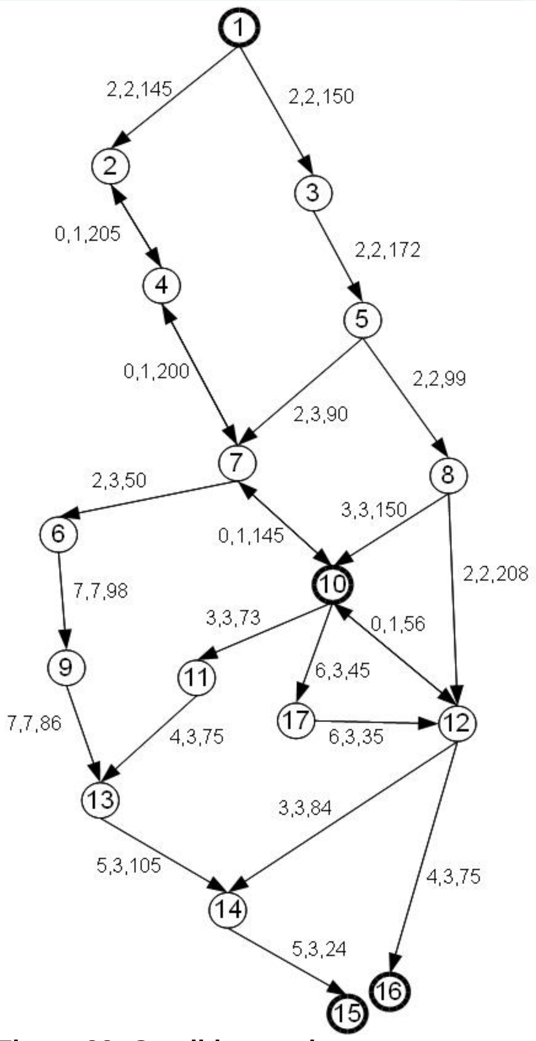
Figure 33: Candidate nodes

This map differs from the encoder map (see Figure 28) in several ways. Some length values are different (e.g. line from node 3 to 5), some functional road class values have changed (e.g. line from node 3 to 5) and there are two more nodes 16 and 17 and also additional lines connecting these new nodes. The challenge of the decoder is to resolve the location in this different map.

After validating the data the decoder starts determining candidate nodes for each location reference point (step 2). Figure 33 shows the candidate nodes (bold circle) which are positioned close to the coordinates of the location reference points. For the location reference point 1 and 2 exist only one candidate node but for the last location reference point two candidate nodes are in the run.

Step 3 deals with determining candidate lines for each location reference point. The bold lines in Figure 34 are the candidate lines for this example. The location reference point 1 has two outgoing lines as candidates, location reference point 2 has three outgoing lines in the run and the last location reference point has two incoming lines (one for each candidate node).

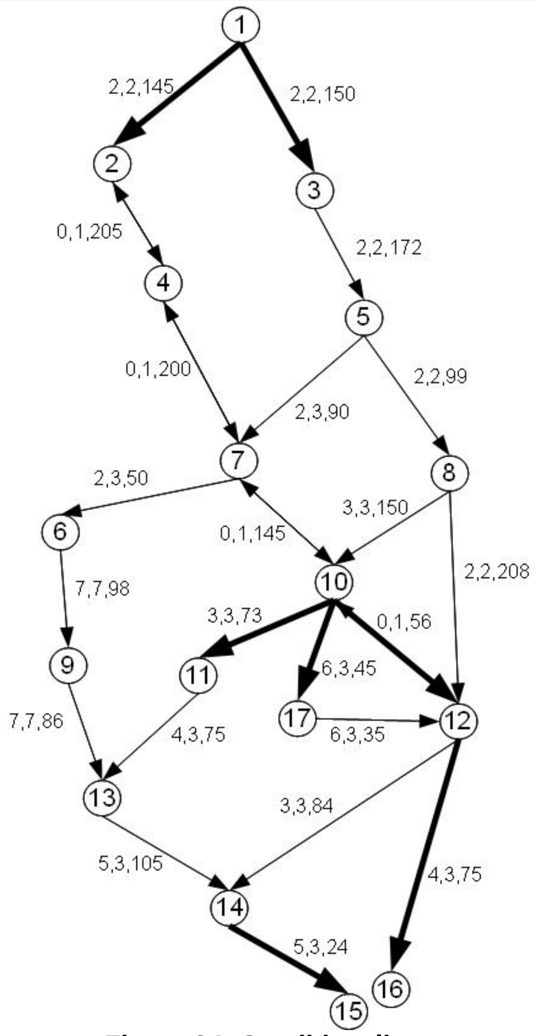

Figure 34: Candidate lines

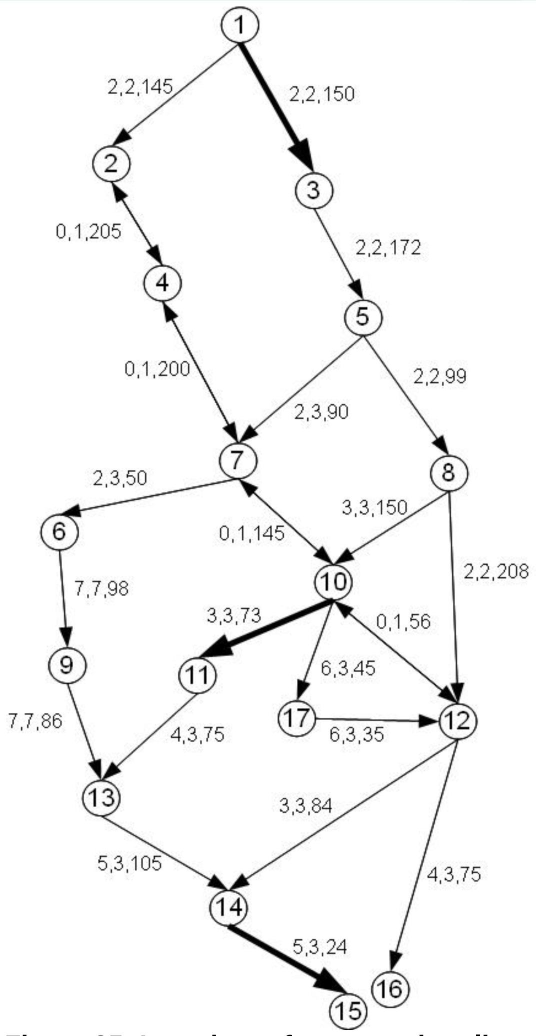

Figure 35: Location reference point - lines

Each candidate line is rated in step 4 according to its compliance with the attributes of the location reference point and its distance to the location reference coordinate. The bold lines in Figure 35 show the lines which are rated best. These lines are used for the following shortest-path calculation in step 5 of the decoding process.

The shortest-path calculation is run for each subsequent pair of location reference points and the concatenation of the two shortest-paths in this example are shown in Figure 36. These shortest-paths are validated in step 6. The length of the first shortest-path (from node 1 to node 10) is 557 meters and this value fits into the distance to next point interval of the first location reference point (527.4 meters – 586.0 meters). The length of the second shortest-path (from node 10 to node 15) is 277 meters and this value also fits into the distance to next point interval of the location reference point 2 (234.4 meters – 293.0 meters). The shortest-paths are validated and the decoder does not fail.

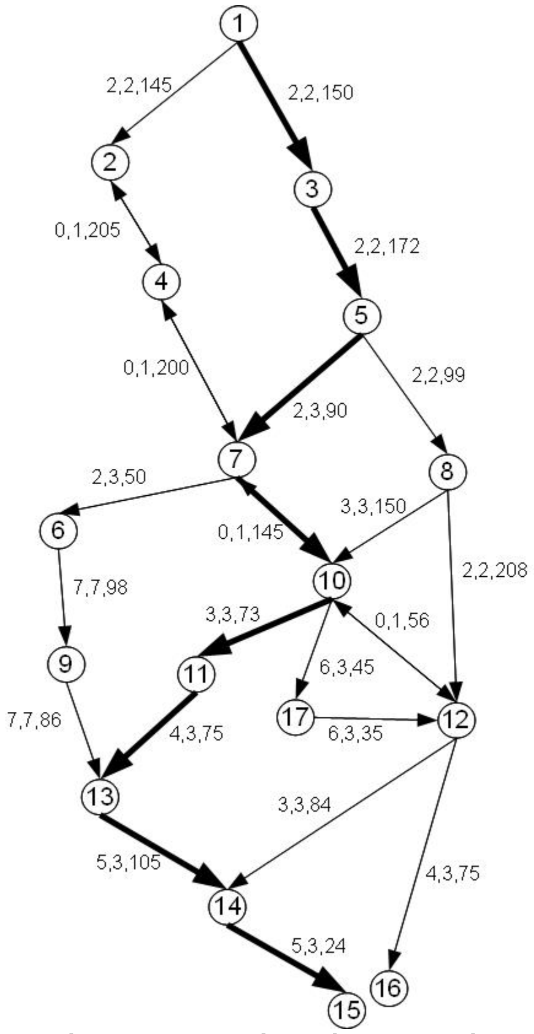

Figure 36: Decoder - shortest path

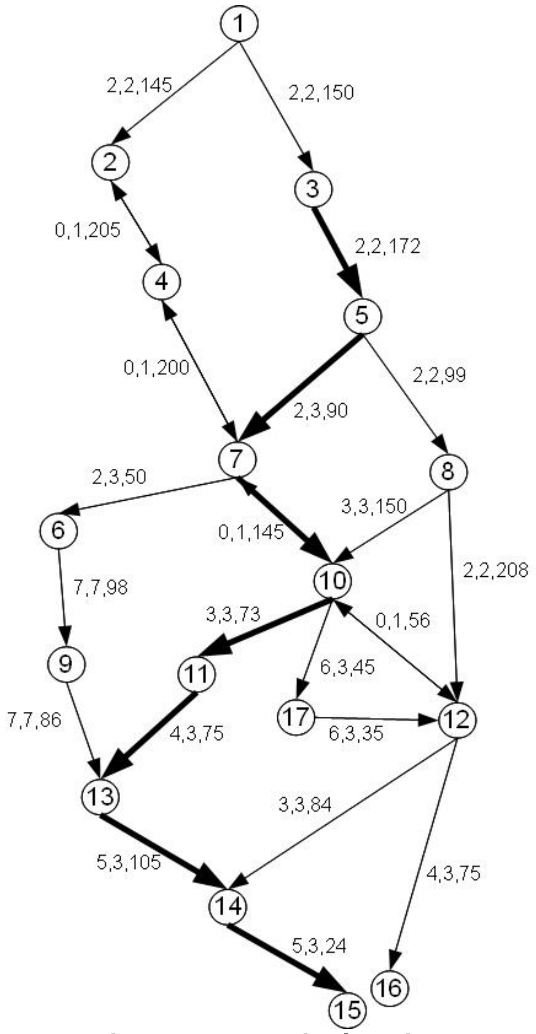

Figure 37: Decoder location

The last step of the decoding process (step 7) now trims the concatenated shortest-path according to the offsets. The example only holds a positive offset and therefore the decoder trims the shortest path at its start. The length between the first two location reference points is 557 meters. The offset value is therefore 557 * 0.2676 = 149 meters. It points out that node 3 is the real start of the location and the decoded location is finally shown in Figure 37.

13.2 Point location

13.2.1. Example setup

The encoder and decoder maps (see Figure 38 and Figure 39) are extracts of the maps being used for the line location example. Both maps consist of 6 nodes and 8 lines. The necessary line attributes are shown beside every line using the format: <FRC>, <FOW>, <Length in meter>. The arrows indicate the driving direction for each line. The coordinates chosen are exemplary and do not correspond to any real world objects!

All nodes in the network shall be valid even there are only two lines attached to the nodes!!

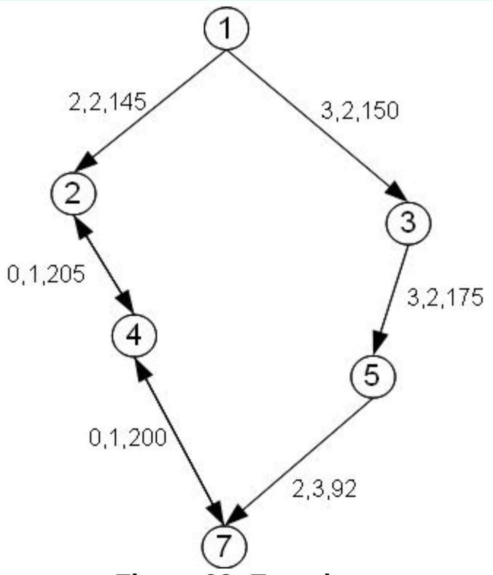

Figure 38: Encoder map

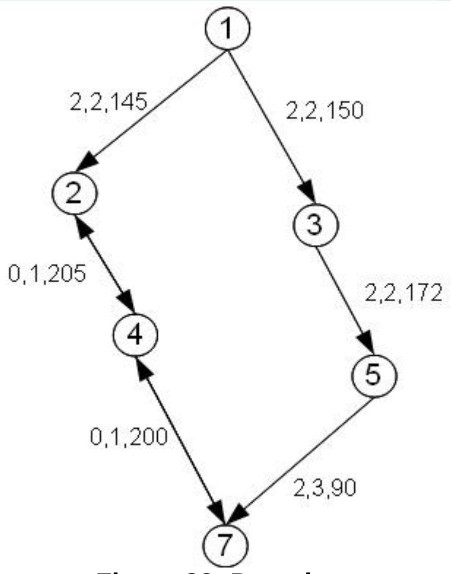

Figure 39: Decoder map

The following examples uses the coordinates for nodes 5 and 7 as shown in Table 94.

Node ID	Longitude	Latitude
5	6.12829°	49.60597°
7	6.12779°	49.60521°

Table 94: Example data

13.2.2. GeoCoordinate

The geo coordinate example shows the simplest type of a point location. Figure 40 shows the geo coordinates to be encoded with a black cross and the coordinates used in this example are shown in Table 95.

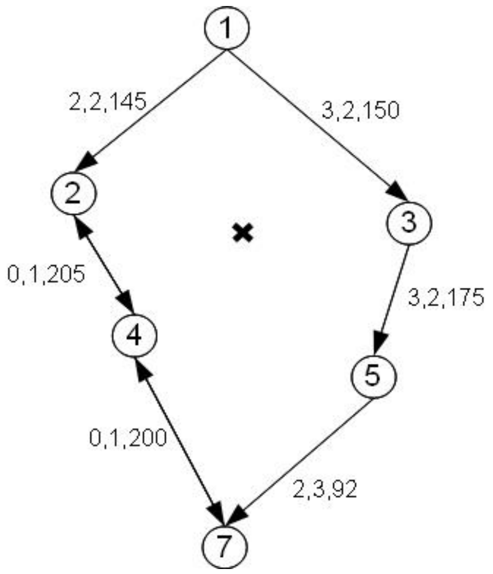

Figure 40: Example – GeoCoordinate (encoder)

	Longitude	Latitude
X	6.12699°	49.60728°

Table 95: Example coordinates

13.2.2.1 Encoding

The encoding of geo coordinates starts with a validation of the coordinates itself. These must fit into the valid ranges for longitude and latitude coordinates. If the geo coordinates are invalid then the encoder shall report an error. If the longitude and latitude values are correct then the encoder can immediately create the physical format.

Binary location reference
The size of the binary location reference for a GeoCoordinate is 7 bytes. The following tables (Table 96 to Table 98) explain the layout for each single byte.

The status byte indicates that the location reference describes a point location using version 3 of the binary format. The geo coordinates (longitude and latitude values) are transformed into integer values according to Equation 1 and then translated into the binary format.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	0	1	0	0	0	1	1

Table 96: Binary example - GeoCoordinate: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	1	1	0	0	0	1	1

Table 97: Binary example - GeoCoordinate: Absolute longitude

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	1	0	1	1	1	0	1	1

Table 98: Binary example - GeoCoordinate: Absolute latitude

A Base64 encoded representation of the binary data is the following string:

IwRbYyNGuw==

XML location reference
The XML location reference shows a point location reference which consists of a GeoCoordinate. This special type of point location contains only a pair of longitude and latitude values referring to the geo coordinates.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<openlr:OpenLR xmlns="http://www.openlr.org/openlr">
    <openlr:LocationID>TestID01</openlr:LocationID>
    <openlr:XMLLocationReference>
        <openlr:PointLocationReference>
            <openlr:GeoCoordinate>
                <openlr:Coordinates>
                    <openlr:Longitude>6.12699</openlr:Longitude>
                    <openlr:Latitude>49.60728</openlr:Latitude>
                </openlr:Coordinates>
            </openlr:GeoCoordinate>
        </openlr:PointLocationReference>
    </openlr:XMLLocationReference>
</openlr:OpenLR>
```

13.2.2.2 Decoding

The decoding starts with the validation of the physical format. If the format is invalid and cannot be read then the decoder should fail.

If the physical format is valid the decoder needs to extract the geo coordinates. For the binary physical format this will result in the longitude and latitude values shown in Table 99. The decoded GeoCoordinate is shown in Figure 41.

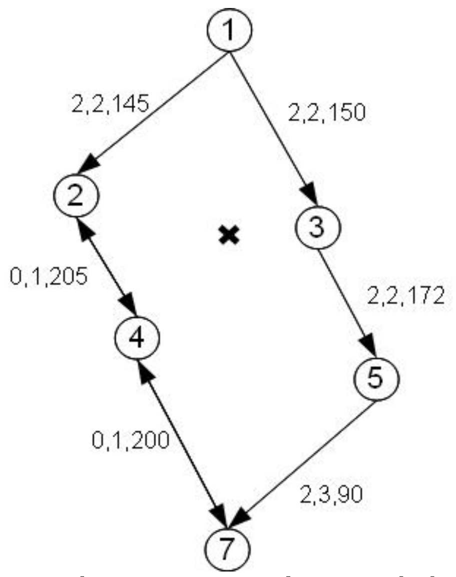

Figure 41: Example: Decoded GeoCoordinate

	Longitude	Latitude
X	6.12699°	49.60727°

Table 99: Decoded GeoCoordinate values

13.2.3. PointAlongLine

The PointAlongLine defines a point location directly on a line or near by a line. The point of interest is identified by a network line an offset value. In this example the point location marked with a black cross in Figure 42 shall be encoded. The distance between the start of the line and the point of interest shall be 28 meter or 30.43% of the line length. The additional information about the side of the road and the orientation are mentioned in Table 100.

Information type	Value
SideOfRoad	LEFT
Orientation	NO_ORIENTATION_OR_UNKNOWN

Table 100: Additional information for point along line

Please note that all nodes in the example shall be considered as valid even if there are only two lines attached to a node!

13.2.3.1 Encoding

The encoding of a PointAlongLine location starts with validating the input data. The point location shown in Figure 42 needs to be translated into a network line and an offset value. The point location is on the determined line and the offset describes the distance between the start node of that line and the point location (see Figure 43).

The line including the offset value will be the input for the OpenLR™ encoder and needs to be checked. The offset value shall not exceed the length of the line. If the validation of the input data fails then the encoder shall report an error.

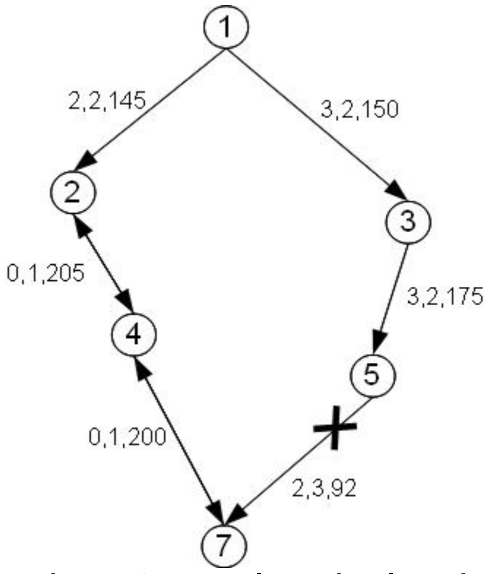

Figure 42: Example - PointAlongLine

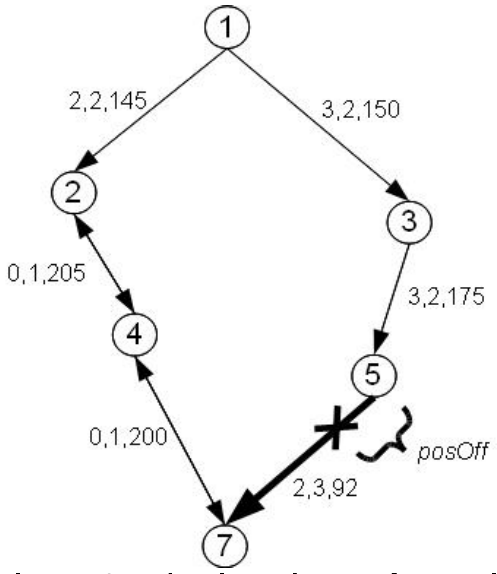

Figure 43: PointAlongLine - Reference line and offset

If the input data is valid then the encoder would try to expand the referenced line in order to find valid nodes as start and end of the location. As all nodes are defined as valid for this example setup the encoder can directly create two location reference points describing the reference line. The first LRP points to the start of the line (node 5 in Figure 43) and the second LRP (last LRP) points to the end of the reference line (node 7 in Figure 43). The line will therefore be referenced twice but one time in the direction from node 5 to node 7 and the second time against that direction.

Using two LRPs also provides the opportunity to use an artificial location reference point (not to be bound to a network node) in order to reduce the distance between the two location reference points and therefore to increase the accuracy of the offset.

In the last step the two location reference points and the offset value needs to be physically encoded.

Binary location reference

The following tables (Table 101 to Table 111) show the details of the binary physical format (version 3). The data size in this example is 17 bytes. The positive offset value of 30.43% falls into the bucket number 77.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	0	1	0	1	0	1	1

Table 101: Binary example: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	0	0

Table 102: Binary example: LRP 1 – absolute longitude

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0	1	1	1	1	1	1	0

Table 103: Binary example: LRP1 – absolute latitude

Bit	7	6	5	4	3	2	1	0
Description	Orientation		FRC			FOW		
Value	0	0	0	1	0	0	1	0

Table 104: Binary example: LRP1 – attribute 5

Bit	7	6	5	4	3	2	1	0
Description	LFRCNP			Bearing				
Value	0	1	0	1	0	0	0	1

Table 105: Binary example: LRP1 – attribute 2

Bit	7	6	5	4	3	2	1	0
Description	DNP							
Value	0	0	0	0	0	0	0	1

Table 106: Binary example: LRP1 – attribute 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	0

Table 107: Binary example: LRP2 – relative longitude

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	0

Table 108: Binary example: LRP2 – relative latitude

Bit	7	6	5	4	3	2	1	0
Description	SideOfRoad		FRC			FOW		
Value	1	0	0	1	0	0	1	0

Table 109: Binary example: LRP2 – attribute 6

Bit	7	6	5	4	3	2	1	0
Description	RFU	PoffF	NoffF	Bearing				
Value	0	1	0	1	0	1	0	0

Table 110: Binary example: LRP3 – attribute 4

Bit	7	6	5	4	3	2	1	0
Description	POFF							
Value	0	1	0	0	1	1	0	1

Table 111: Binary example: positive Offset

A Base64 encoded representation of the binary data is the following string:

KwRboCNGfhJRAf/O/7SSQ00=

XML location reference

The following XML location reference shows a PointAlongLine location consisting of exactly two location reference points and one (positive) offset.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<openlr:OpenLR xmlns="http://www.openlr.org/openlr">
    <openlr:LocationID>TestID02</openlr:LocationID>
    <openlr:XMLLocationReference>
        <openlr:PointLocationReference>
            <openlr:PointAlongLine>
                <openlr:LocationReferencePoint>
                    <openlr:Coordinates>
                        <openlr:Longitude>6.12829</openlr:Longitude>
                        <openlr:Latitude>49.60597</openlr:Latitude>
                    </openlr:Coordinates>
                    <openlr:LineAttributes>
                        <openlr:FRC>FRC2</openlr:FRC>
                        <openlr:FOW>MULTIPLE_CARRIAGEWAY</openlr:FOW>
                        <openlr:BEAR>202</openlr:BEAR>
                    </openlr:LineAttributes>
                    <openlr:PathAttributes>
                        <openlr:LFRCNP>FRC2</openlr:LFRCNP>
                        <openlr:DNP>92</openlr:DNP>
                    </openlr:PathAttributes>
                </openlr:LocationReferencePoint>
                <openlr:LastLocationReferencePoint>
                    <openlr:Coordinates>
                        <openlr:Longitude>6.12779</openlr:Longitude>
                        <openlr:Latitude>49.60521</openlr:Latitude>
                    </openlr:Coordinates>
                    <openlr:LineAttributes>
                        <openlr:FRC>FRC2</openlr:FRC>
                        <openlr:FOW>MULTIPLE_CARRIAGEWAY</openlr:FOW>
                        <openlr:BEAR>42</openlr:BEAR>
                    </openlr:LineAttributes>
                </openlr:LastLocationReferencePoint>
                <openlr:Offsets>
                    <openlr:PosOff>28</openlr:PosOff>
                    <openlr:NegOff>0</openlr:NegOff>
                </openlr:Offsets>
                <openlr:SideOfRoad>LEFT</openlr:SideOfRoad>
                <openlr:Orientation>NO_ORIENTATION_OR_UNKNOWN</openlr:Orientation>
            </openlr:PointAlongLine>
        </openlr:PointLocationReference>
    </openlr:XMLLocationReference>
</openlr:OpenLR>
```

13.2.3.2 Decoding

The decoding process starts with a validation of the physical representation of the location reference. If the input data is not valid the decoder should report an error.

After the validation phase the decoder tries to find network nodes which match the coordinates for the location reference points. The first location reference point should result in node 5 and the second location reference point should result in node 7.

In the next step the decoder resolves for each location reference point a line which is an outgoing line for all location reference points except the last one and an incoming line for the last location reference point whereby these lines match the given LRP attributes. A ranking of these lines will determine a line which matches best. In the example the best matching line for both location reference points is the line between the nodes 5 and 7 (see Figure 44).

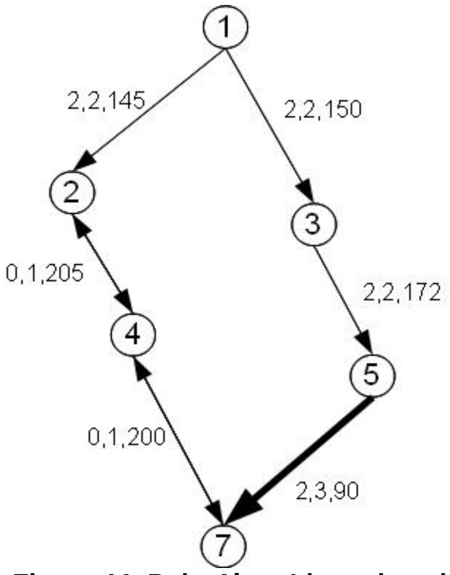

Figure 44: PointAlongLine - decoded reference line

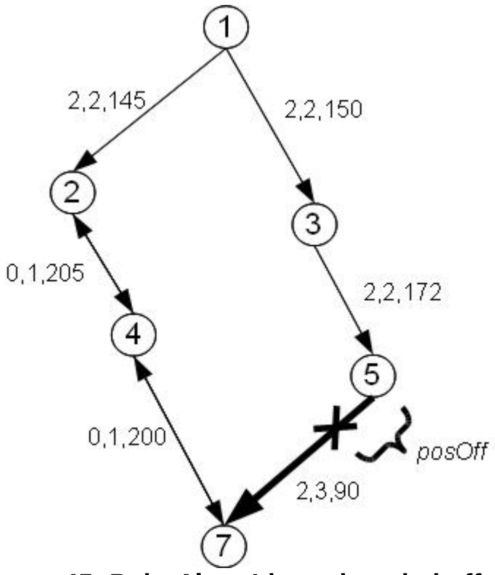

Figure 45: PointAlongLine - decoded offset

If the best matching lines for the first and second location reference point are not the same then the decoder should calculate the shortest path between these two lines. The resulting path shall be used in the next steps.

After having determined the path between the first and second LRP the offset value needs to be calculated. If the binary physical format version 3 is used the decoder would need to determine the length of this path. This length is used to resolve the percentage value into a meter value. The offset percentage value is 30.27% and the resulting offset length is 27.8 meter (see Figure 45).

The values for SideOfRoad and Orientation are read from the physical format and the decoded PointAlongLine location is shown in Figure 46.

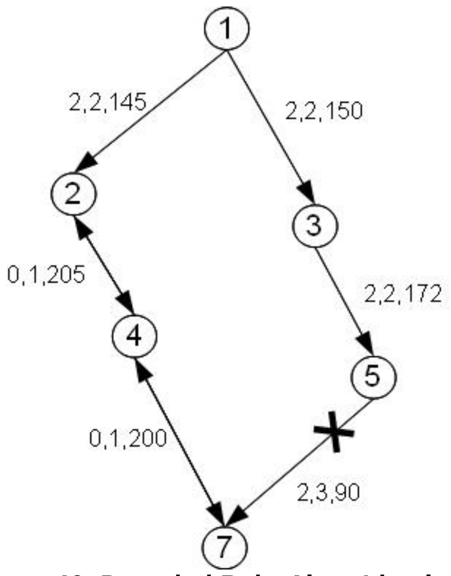

Figure 46: Decoded PointAlongLine location

13.2.4. PoiWithAccessPoint

The PoiWithAccessPoint defines a point of interest as a geo coordinate and an access point to that point directly on a line or near by a line (identified by a network line an offset value). In the example shown in Figure 47 the point of interest is marked with a small black cross and the access point with a large black cross. The distance between the start of the line and the access point shall be 28 meter or 30.43% of the line length. The coordinates of the point of interest are shown in Table 112. The additional information about the side of the road and the orientation are mentioned in Table 113.

	Longitude	Latitude
X	6.12699°	49.60728°

Table 112: Example coordinates

Information type	Value
SideOfRoad	LEFT
Orientation	NO_ORIENTATION_OR_UNKNOWN

Table 113: Additional information for poi with access point

13.2.4.1 Encoding

The encoding of a PoiWithAccessPoint location starts with validating the input data. The access point of the point location shown in Figure 47 needs to be translated into a network line and an offset value. The access point is on the determined line and the offset describes the distance between the start node of that line and the point location (see Figure 48). The point of interest is represented by the geo coordinate.

The geo coordinate and the line including the offset value will be the input for the OpenLR™ encoder and needs to be checked. The offset value shall not exceed the length of the line and the coordinate values shall not exceed the valid boundaries for longitude and latitude values. If the validation of the input data fails then the encoder shall report an error.

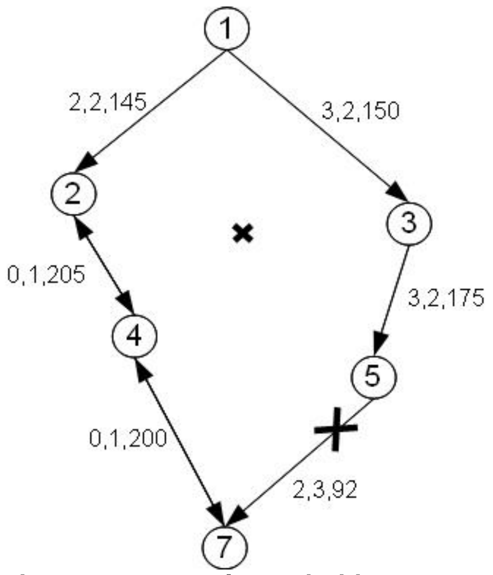

Figure 47: Example - PoiWithAccessPoint

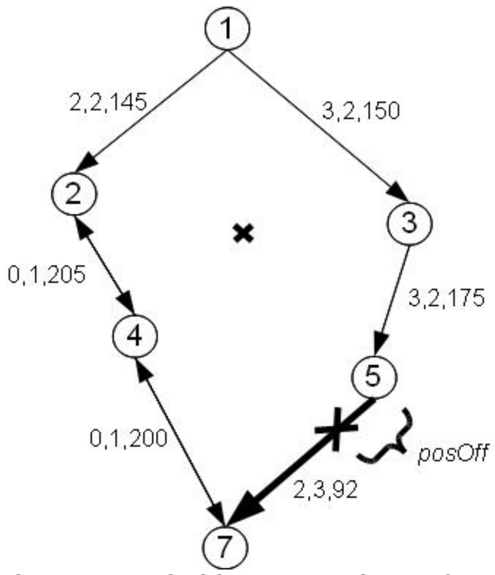

Figure 48: PoiWithAccessPoint - reference line and offset

If the input data is valid then the encoder would try to expand the referenced line in order to find valid nodes as start and end of the location. As all nodes are defined as valid the encoder can directly create two location reference points describing the reference line. The first LRP points to the start of the line (node 5 in Figure 43) and the second LRP (last LRP) points to the end of the reference line (node 7 in Figure 43). The line will therefore be referenced twice but one time in the direction form node 5 to node 7 and the second time against that direction.

Using two LRPs also provides the opportunity to use an artificial location reference point (not to be bound to a network node) in order to reduce the distance between the two location reference points and therefore to increase the accuracy of the offset.

In the last step the two location reference points, the offset value and the geo coordinate needs to be physically encoded.

Binary location reference

The following tables (Table 114 to Table 126) show the details of the binary physical format (version 3). The data size in this example is 21 bytes. The positive offset value of 30.43% falls into the bucket number 77. The geo coordinate (point of interest) will be referenced relatively to the coordinates of the first location reference point.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	0	1	0	1	0	1	1

Table 114: Binary example: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	0	0

Table 115: Binary example: LRP 1 – absolute longitude

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0	1	1	1	1	1	1	0

Table 116: Binary example: LRP1 – absolute latitude

Bit	7	6	5	4	3	2	1	0
Description	Orientation		FRC			FOW		
Value	0	0	0	1	0	0	1	0

Table 117: Binary example: LRP1 – attribute 5

Bit	7	6	5	4	3	2	1	0
Description	LFRCNP			Bearing				
Value	0	1	0	1	0	0	0	1

Table 118: Binary example: LRP1 – attribute 2

Bit	7	6	5	4	3	2	1	0
Description	DNP							
Value	0	0	0	0	0	0	0	1

Table 119: Binary example: LRP1 – attribute 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	0

Table 120: Binary example: LRP2 – relative longitude

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	0

Table 121: Binary example: LRP2 – relative latitude

Bit	7	6	5	4	3	2	1	0
Description	SideOfRoad		FRC			FOW		
Value	1	0	0	1	0	0	1	0

Table 122: Binary example: LRP2 – attribute 6

Bit	7	6	5	4	3	2	1	0
Description	RFU	PoffF	NoffF	Bearing				
Value	0	1	0	1	0	1	0	0

Table 123: Binary example: LRP3 – attribute 4

Bit	7	6	5	4	3	2	1	0
Description	POFF							
Value	0	1	0	0	1	1	0	1

Table 124: Binary example: positive Offset

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1

Table 125: Binary example: POI – relative longitude

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1

Table 126: Binary example: POI – relative latitude

A Base64 encoded representation of the binary data is the following string:

KwRboCNGfhJRAf/O/7SSQ03/fgCD

XML location reference

The following XML location reference shows a PoiWithAccessPoint location consisting of exactly two location reference points, one (positive) offset and a geo coordinate.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<openlr:OpenLR xmlns="http://www.openlr.org/openlr">
    <openlr:LocationID>TestID03</openlr:LocationID>
    <openlr:XMLLocationReference>
        <openlr:PointLocationReference>
            <openlr:PoiWithAccessPoint>
                <openlr:LocationReferencePoint>
                    <openlr:Coordinates>
                        <openlr:Longitude>6.12829</openlr:Longitude>
                        <openlr:Latitude>49.60597</openlr:Latitude>
                    </openlr:Coordinates>
                    <openlr:LineAttributes>
                        <openlr:FRC>FRC2</openlr:FRC>
                        <openlr:FOW>MULTIPLE_CARRIAGEWAY</openlr:FOW>
                        <openlr:BEAR>202</openlr:BEAR>
                    </openlr:LineAttributes>
                    <openlr:PathAttributes>
                        <openlr:LFRCNP>FRC2</openlr:LFRCNP>
                        <openlr:DNP>92</openlr:DNP>
                    </openlr:PathAttributes>
                </openlr:LocationReferencePoint>
                <openlr:LastLocationReferencePoint>
                    <openlr:Coordinates>
                        <openlr:Longitude>6.12779</openlr:Longitude>
                        <openlr:Latitude>49.60521</openlr:Latitude>
                    </openlr:Coordinates>
                    <openlr:LineAttributes>
                        <openlr:FRC>FRC2</openlr:FRC>
                        <openlr:FOW>MULTIPLE_CARRIAGEWAY</openlr:FOW>
                        <openlr:BEAR>42</openlr:BEAR>
                    </openlr:LineAttributes>
                </openlr:LastLocationReferencePoint>
                <openlr:Offsets>
                    <openlr:PosOff>28</openlr:PosOff>
                    <openlr:NegOff>0</openlr:NegOff>
                </openlr:Offsets>
                <openlr:Coordinates>
                    <openlr:Longitude>6.12699</openlr:Longitude>
                    <openlr:Latitude>49.60728</openlr:Latitude>
                </openlr:Coordinates>
                <openlr:SideOfRoad>LEFT</openlr:SideOfRoad>
                <openlr:Orientation>NO_ORIENTATION_OR_UNKNOWN</openlr:Orientation>
            </openlr:PoiWithAccessPoint>
        </openlr:PointLocationReference>
    </openlr:XMLLocationReference>
</openlr:OpenLR>
```

13.2.4.2 Decoding

The decoding process starts with a validation of the physical representation of the location reference. If the input data is not valid the decoder should report an error.

After the validation phase the decoder tries to find network nodes which match the coordinates for the location reference points. The first location reference point should result in node 5 and the second location reference point should result in node 7.

In the next step the decoder resolves for each location reference point a line which is an outgoing line for all location reference points except the last one and an incoming line for the last location reference point whereby these lines matches the given LRP attributes. A ranking of these lines will determine a line which matches best. In the example the best matching line for both location reference points is the line between the nodes 5 and 7 (see Figure 49).

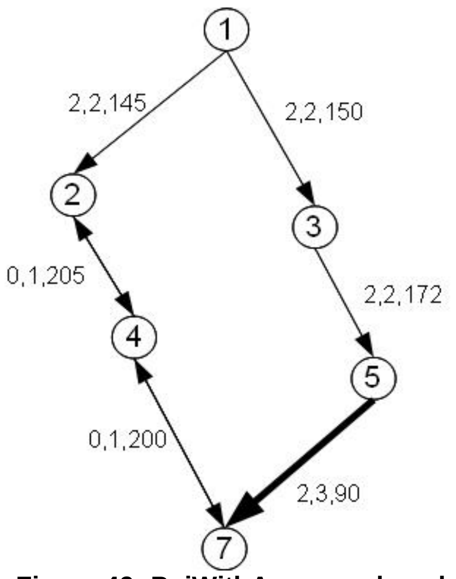

Figure 49: PoiWithAccess - decoded reference line

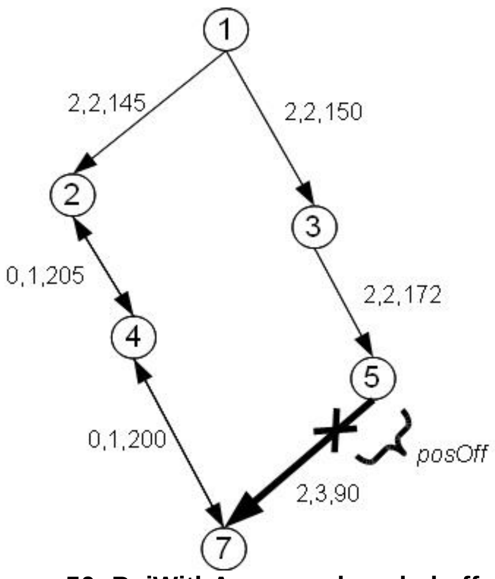

Figure 50: PoiWithAccess - decoded offset

If the best matching lines for the first and second location reference point are not the same then the decoder should calculate the shortest path between these two lines. The resulting path shall be used in the next steps.

After having determined the path between the first and second LRP the offset value needs to be calculated. If the binary physical format version 3 is used the decoder would need to determine the length of this path. This length is used to resolve the percentage value into a meter value. The offset percentage value is 30.27% and the resulting offset length is 27.8 meter (see Figure 50).

The geo coordinates of the point of interest are shown in Table 127. The values for SideOfRoad and Orientation are read from the physical format and the decoded PoiWithAccessPoint location is shown in Figure 51.

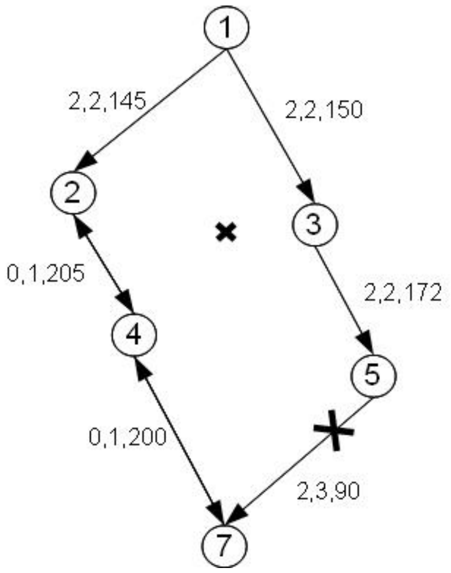

Figure 51: Decoded PoiWithaccessPoint location

	Longitude	Latitude
X	6.12701°	49.60727°

Table 127: Decoded GeoCoordinate values

13.3 Area location

13.3.1. Circle

Figure 52 illustrates a circle location inside the network. In principle circle location references are not bound to the road network. Anyhow it’s laid behind to give a demonstrative impression of the practical use case. The center coordinate is marked with the black cross, the radius drawn by a dashed circle.
Table 128 lists the parameters of the location reference.

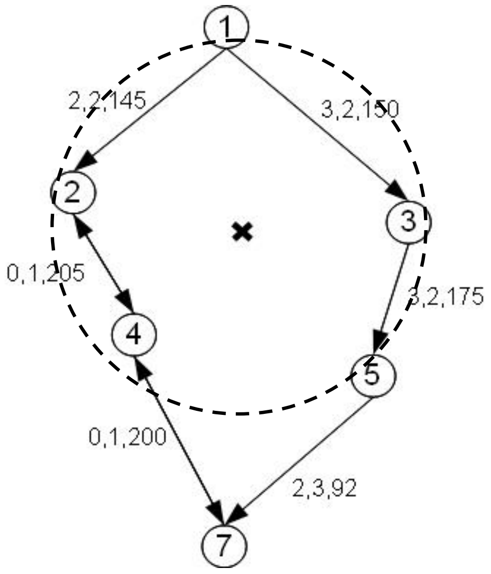

Figure 52: Example – Circle (encoder)

Information type	Value
Center longitude	6.12699°
Center latitude	49.60728°
Radius	170 meters

Table 128: Example parameters

13.3.1.1 Encoding

The encoding of circle location references starts with a validation of the coordinates of the center point. These must fit into the valid ranges for longitude and latitude coordinates. If the geo coordinates are invalid the encoder shall report an error. If the longitude and latitude values are correct the encoder can immediately create the physical format.

Binary location reference

The size of the binary location reference varies from 8 to 11 bytes depending on the size of the radius value. The here described example contains a small radius which fits into one byte. The location reference therefore takes 8 bytes. The following tables (Table 129 to Table 132) explain the layout for each single byte.

The status byte indicates that the location reference describes no point location. Area bits 0 and 1 are set to zero. Furthermore version 3 of the binary format is specified. The geo coordinates (longitude and latitude values) of the center point are transformed into integer values according to Equation 1 and then translated into the binary format. The single radius byte finishes off the location reference.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	0	0	0	0	0	1	1

Table 129: Binary example - Circle: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	1	1	0	0	0	1	1

Table 130: Binary example - Circle: Absolute longitude center point

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	1	0	1	1	1	0	1	1

Table 131: Binary example - Circle: Absolute latitude center point

Bit	7	6	5	4	3	2	1	0
Value	1	0	1	0	1	0	1	0

Table 132: Binary example - Circle: radius

A Base64 encoded representation of the binary data is the following string:

AwRbYyNGu6o=

XML location reference
The XML location reference shows an area location reference that consists of a Circle.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OpenLR xmlns="http://www.openlr.org/openlr">
    <LocationID>circle</LocationID>
    <XMLLocationReference>
        <AreaLocationReference>
            <CircleLocationReference>
                <GeoCoordinate>
                    <Coordinates>
                        <Longitude>6.12699</Longitude>
                        <Latitude>49.60728</Latitude>
                    </Coordinates>
                </GeoCoordinate>
                <Radius>100</Radius>
            </CircleLocationReference>
        </AreaLocationReference>
    </XMLLocationReference>
</OpenLR>
```

13.3.1.2 Decoding

The decoding starts with the validation of the physical format. If the format is invalid and cannot be read then the decoder should fail.

If the physical format is valid the decoder needs to extract the geo coordinates and the radius. For the binary physical format this will result in the values shown in Table 133. The decoded Circle is shown in Figure 53.

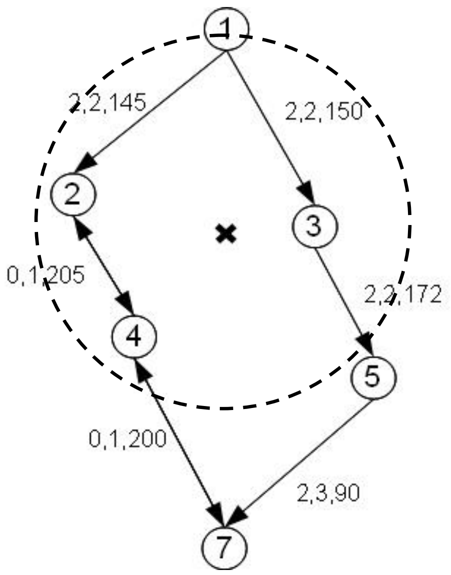

Figure 53: Example - Decoded Circle

Information type	Value
Center longitude	6.12699°
Center latitude	49.60727°
Radius	170 meters

Table 133: Decoded attributes

13.3.2. Rectangle

Figure 54 illustrates a rectangle location inside the network. In principle rectangle location references are not bound to the road network. Anyhow it’s laid behind to give a demonstrative impression of the practical use case. The two coordinates defining the rectangles lower left and upper right corners are marked with a black dot.

Coordinate	Longitude	Latitude
Lower Left	6.12555°	49.60586°
Upper Right	6.12875°c	49.60711°

Table 134Table 134 lists the parameters of the location reference.

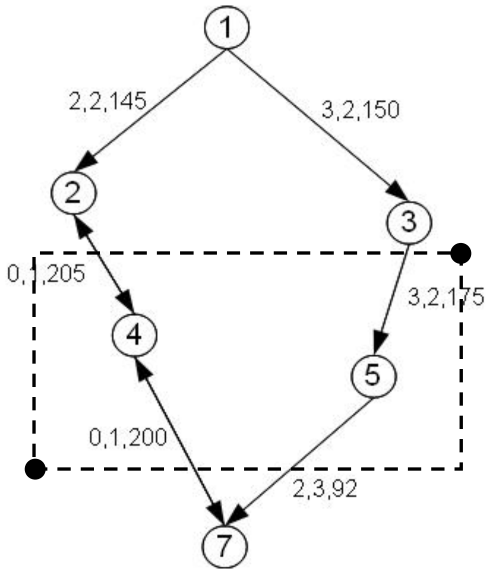

Figure 54: Example - Rectangle (encoder)

Coordinate	Longitude	Latitude
Lower Left	6.12555°	49.60586°
Upper Right	6.12875°c	49.60711°

Table 134: Example parameters

13.3.2.1 Encoding

The encoding of rectangle location references starts with a validation of the coordinates of the corner points. These must fit into the valid ranges for longitude and latitude coordinates. If the geo coordinates are invalid the encoder shall report an error. If the longitude and latitude values are correct the encoder can immediately create the physical format.

Binary location reference

The size of the binary location reference is 11 bytes for small or 13 for large rectangles. The here described example possesses a distance between lower left and upper right that fits into a small binary location reference container. The following tables (Table 135 to Table 139) explain the layout for each single byte.

The status byte indicates that the location reference describes an area location with area bits 1 and 0 set to $(10)_2$. Furthermore version 3 of the binary format is specified. The geo coordinates of the lower left point are transformed into integer values according to Equation 1 and then translated into the binary format. The upper right corner coordinates are calculated relative from the lower left using Equation 3 and then transformed to the binary format.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	1	0	0	0	0	1	1

Table 135: Binary example - Rectangle: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	0	1	0	0	0	0	0

Table 136: Binary example - Rectangle: Absolute longitude lower left

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0	1	1	1	1	0	0	1

Table 137: Binary example - Rectangle: Absolute latitude lower left

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0

Table 138: Binary example: Rectangle – relative longitude upper right

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1

Table 139: Binary example: Rectangle – relative latitude upper right

A Base64 encoded representation of the binary data is the following string:

QwRbICNGeQFAAH0=

XML location reference

The XML location reference shows an area location reference that consists of a Rectangle.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OpenLR xmlns="http://www.openlr.org/openlr">
    <LocationID>rectangle</LocationID>
    <XMLLocationReference>
        <AreaLocationReference>
            <RectangleLocationReference>
                <Rectangle>
                    <LowerLeft>
                        <Longitude>6.12555</Longitude>
                        <Latitude>49.60586c</Latitude>
                    </LowerLeft>
                    <UpperRight>
                        <Longitude>6.12875c</Longitude>
                        <Latitude>49.60711</Latitude>
                    </UpperRight>
                </Rectangle>
            </RectangleLocationReference>
        </AreaLocationReference>
    </XMLLocationReference>
</OpenLR>
```

13.3.2.2 Decoding

The decoding starts with the validation of the physical format. If the format is invalid and cannot be read then the decoder should fail.

If the physical format is valid the decoder needs to extract the geo coordinates of the corner points. For the binary physical format this will result in the values shown in Table 140. The decoded rectangle is shown in Figure 55.

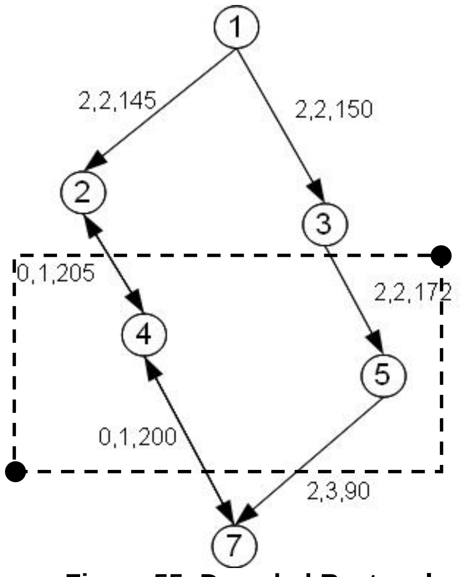

Figure 55: Decoded Rectangle

Coordinate	Longitude	Latitude
Lower Left	6.12555°	49.60586°
Upper Right	6.12875°c	49.60711°

Table 140: Decoded attributes

13.3.3. Grid

Figure 56 illustrates a grid location inside the network. In principle grid location references are not bound to the road network. Anyhow it's laid behind to give a demonstrative impression of the practical use case. The two coordinates defining the lower left and upper right corners of the grids base rectangle are marked with a black dot.

Table 141 lists the parameters of the location reference.

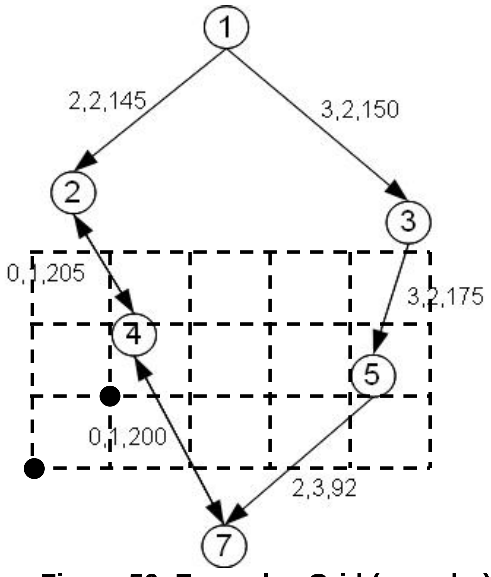

Figure 56: Example - Grid (encoder)

Information type	Value
Lower left longitude	6.12555°
Lower left latitude	49.60586°
Upper right longitude	6.12875°
Upper right latitude	49.60711°
Number columns	5
Number rows	3

Table 141: Example parameters

13.3.3.1 Encoding

The encoding of grid location references starts with a validation of the coordinates of the corner points. These must fit into the valid ranges for longitude and latitude coordinates. If the geo coordinates are invalid the encoder shall report an error. If the longitude and latitude values are correct the encoder can immediately create the physical format.

Binary location reference

The size of the binary location reference is 15 bytes. The following tables (Table 142 to Table 148) explain the layout for each single byte.

The status byte indicates that the location reference describes an area location with area bits 1 and 0 set to $(10)_2$. Furthermore version 3 of the binary format is specified. The geo coordinates of the lower left point are transformed into integer values according to Equation 1 and then translated into the binary format. The upper right corner coordinates are calculated relative from the lower left using Equation 3 and then transformed to the binary format. The coordinates are followed by the grid parameters. Each of both, the number of rows and the columns, consists of two bytes.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	1	0	0	0	0	1	1

Table 142: Binary example - Grid: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	0	1	0	0	0	0	0

Table 143: Binary example - Grid: Absolute longitude lower left

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0	1	1	1	1	0	0	1

Table 144: Binary example - Grid: Absolute latitude lower left

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0

Table 145: Binary example: Grid – relative longitude upper right

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Table 146: Binary example: Grid – relative latitude upper right

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table 147: Binary example: Grid – number of columns

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table 148: Binary example: Grid – number of rows

A Base64 encoded representation of the binary data is the following string:

QwRbICNGeQBKAB8ABQAD

XML location reference

The XML location reference shows an area location reference that consists of a grid.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OpenLR xmlns="http://www.openlr.org/openlr">
    <LocationID>grid</LocationID>
    <XMLLocationReference>
        <AreaLocationReference>
            <GridLocationReference>
                <Rectangle>
                    <LowerLeft>
                        <Longitude>6.12555</Longitude>
                        <Latitude>49.60586</Latitude>
                    </LowerLeft>
                    <UpperRight>
                        <Longitude>6.126291</Longitude>
                        <Latitude>49.606170</Latitude>
                    </UpperRight>
                </Rectangle>
                <NumColumns>5</NumColumns>
                <NumRows>3</NumRows>
            </GridLocationReference>
        </AreaLocationReference>
    </XMLLocationReference>
</OpenLR>
```

13.3.3.2 Decoding

The decoding starts with the validation of the physical format. If the format is invalid and cannot be read then the decoder should fail.

If the physical format is valid the decoder needs to extract the geo coordinates of the corner points of the base rectangle as well as the value for columns and rows. For the binary physical format this will result in the values shown in Table 149. The decoded grid is shown in Figure 57.

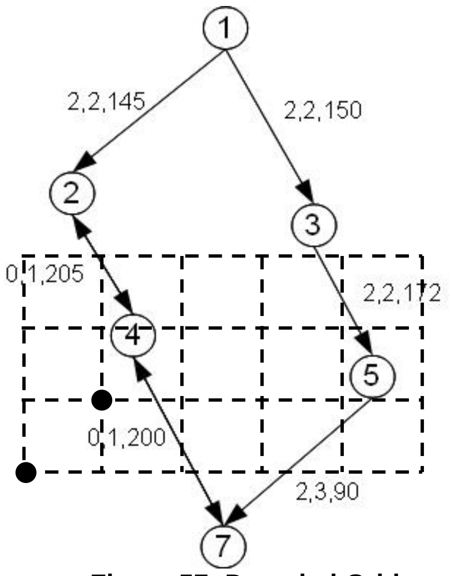

Figure 57: Decoded Grid

Information type	Value
Lower left longitude	6.12555°
Lower left latitude	49.60586°
Upper right longitude	6.12875°c
Upper right latitude	49.60711°
Number columns	5
Number rows	3

Table 149: Decoded attributes

13.3.4. Polygon

Figure 58 illustrates a polygon location inside the network. In principle polygon location references are not bound to the road network. Anyhow it's laid behind to give a demonstrative impression of the practical use case. The coordinates are marked with black dots in which the start coordinate at the lower left is highlighted.

Table 150 lists the coordinate sequence which defines all necessary parameters of the polygon location.

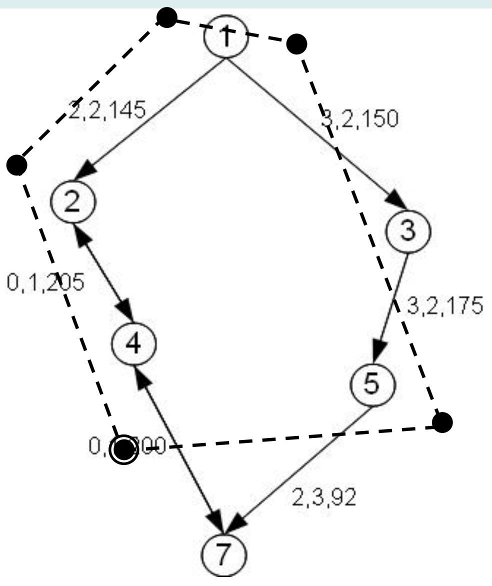

Figure 58: Example - Polygon (encoder)

Coordinate	Longitude	Latitude
Start	6.12549°	49.60576°
Point 2	6.12903°	49.60591°
Point 3	6.12739°	49.60834°
Point 4	6.12657°	49.60870°
Point 5	6.12492°	49.60795°

Table 150: Polygon coordinates

13.3.4.1 Encoding

The encoding of polygon location references starts with a validation of the coordinates. These must fit into the valid ranges for longitude and latitude coordinates. If the geo coordinates are invalid the encoder should report an error. If the longitude and latitude values are correct the encoder can immediately create the physical format from the input data.

Binary location reference

The size of the binary location reference for this polygon is 23 bytes. The following tables (Table 151 to Table 161) explain the layout for each single byte.

The status byte indicates that the location reference describes an area location with area bits 1 and 0 set to $(01)_2$. Furthermore version 3 of the binary format is specified. The geo coordinates of the starting point are transformed into integer values according to Equation 1 and then translated into the binary format. The other coordinates are calculated relative from the starting point using Equation 3 and then transformed to the binary format.

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	0	0	1	0	0	1	1

Table 151: Binary example - Polygon: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	0	0	1	1	1	0	1

Table 152: Binary example - Polygon: Absolute longitude start point

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0	1	1	1	0	1	0	1

Table 153: Binary example - Polygon: Absolute latitude start point

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0

Table 154: Binary example: Polygon – relative longitude point 2

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Table 155: Binary example: Polygon – relative latitude point 2

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	0	1	0	1	1	1	0	0

Table 156: Binary example: Polygon – relative longitude point 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1	1

Table 157: Binary example: Polygon – relative latitude point 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	0

Table 158: Binary example: Polygon – relative longitude point 4

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

Table 159: Binary example: Polygon – relative latitude point 4

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	0	1	0	1	1	0	1	1

Table 160: Binary example: Polygon – relative longitude point 5

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	1

Table 161: Binary example: Polygon – relative latitude point 5

A Base64 encoded representation of the binary data is the following string:

EwRbHSNGdQFiAA//XADz/64AJP9b/7U=

XML location reference

The XML location reference shows an area location reference that consists of a polygon.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OpenLR xmlns="http://www.openlr.org/openlr">
    <LocationID>pentagon</LocationID>
    <XMLLocationReference>
        <AreaLocationReference>
            <PolygonLocationReference>
                <PolygonCorners>
                    <Coordinates>
                        <Longitude>6.12549</Longitude>
                        <Latitude>49.60576</Latitude>
                    </Coordinates>
                    <Coordinates>
                        <Longitude>6.12903</Longitude>
                        <Latitude>49.60591</Latitude>
                    </Coordinates>
                    <Coordinates>
                        <Longitude>6.12739</Longitude>
                        <Latitude>49.60834</Latitude>
                    </Coordinates>
                    <Coordinates>
                        <Longitude>6.12657</Longitude>
                        <Latitude>49.6087</Latitude>
                    </Coordinates>
                    <Coordinates>
                        <Longitude>6.12492</Longitude>
                        <Latitude>49.60795</Latitude>
                    </Coordinates>
                </PolygonCorners>
            </PolygonLocationReference>
        </AreaLocationReference>
    </XMLLocationReference>
</OpenLR>
```

13.3.4.2 Decoding

The decoding starts with the validation of the physical format. If the format is invalid and cannot be read then the decoder should fail.

If the physical format is valid the decoder needs to extract the coordinates. For the binary physical format this will result in the values shown in Table 162. The decoded polygon is shown in Figure 59.

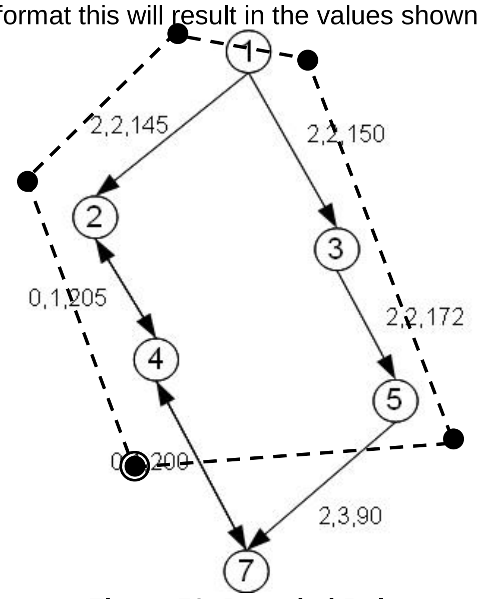

Figure 59: Decoded Polygon

Coordinate	Longitude	Latitude
Start	6.12549°	49.60577°
Point 2	6.12903°	49.60592°
Point 3	6.12739°	49.60835°
Point 4	6.12658°	49.60871°
Point 5	6.12493s°	49.60796°

Table 162: Decoded coordinates

13.3.5. ClosedLine

The encoder map is shown in Figure 60 and consists of 12 nodes and 22 lines (two-way lines are counted twice). The necessary line attributes are shown beside every line using the format: <FRC>, <FOW>, <Length in meters>. The arrows indicate the driving direction for each line. The coordinates are chosen exemplary and do not correspond to any real world objects!

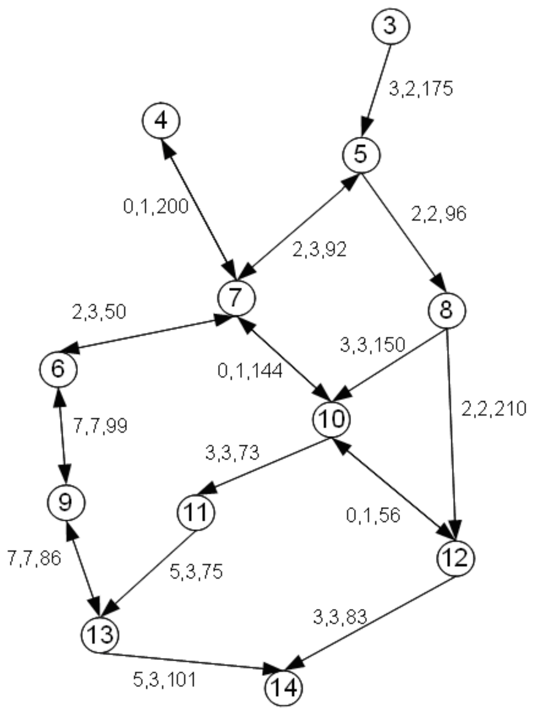

Figure 60: Example network

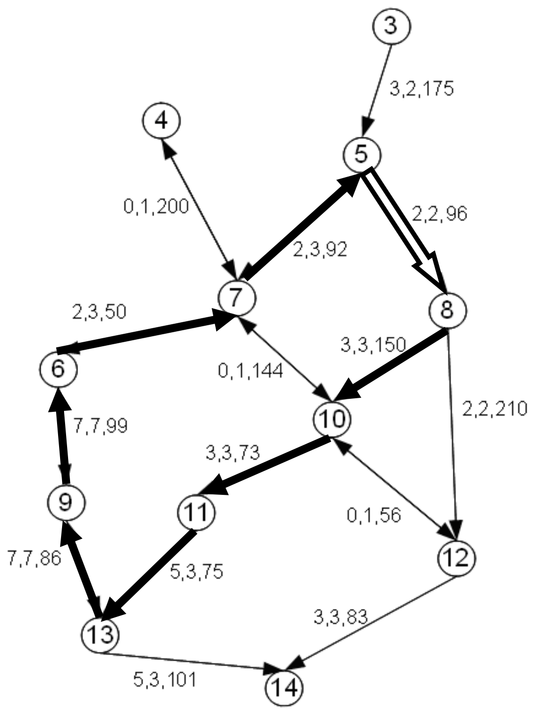

Figure 61: Location

The location to be encoded is shown in Figure 61 using bold lines. The starting line is highlighted by a block arrow. The location ranges from node 5 over nodes 8, 10, 11, 13, 9, 6, 7 back to node 5. Its total length in the encoder map is 770 meters. The ordered list of lines and the map to be used during encoding serves as input for the OpenLR™ encoder.

13.3.5.1 Encoding

In the first step of encoding the location will be checked for validity. Since the location is connected, drivable and closed it is valid.

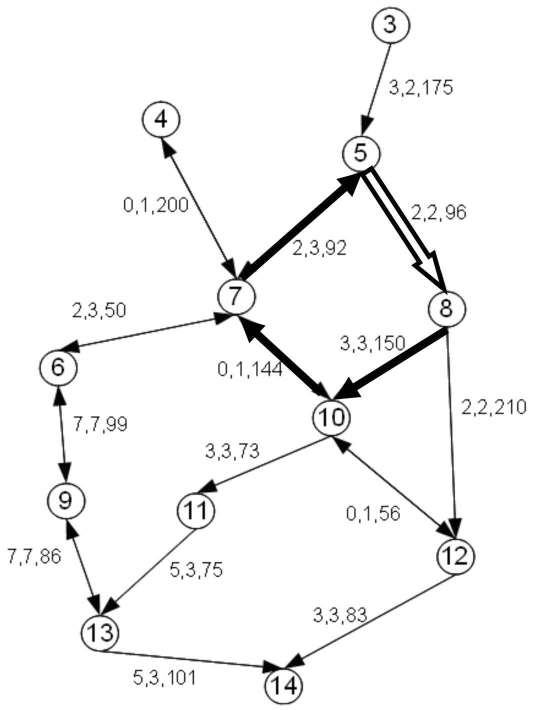

Figure 62: Shortest path

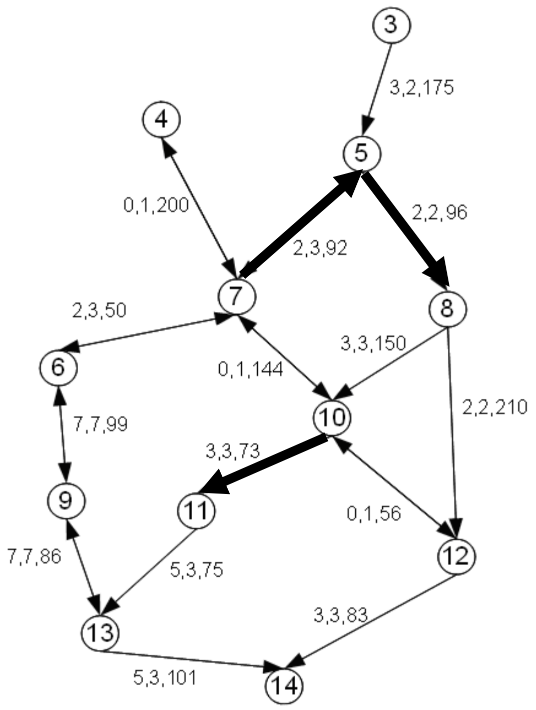

Figure 63: Location reference points

The encoder next calculates a shortest-path between the start line (between nodes 5 and 8) and the end line (between nodes 7 and 5) of the location. The resulting shortest path is outlined in Figure 62 using bold lines. It has a length of 531 meters.

In the next step the encoder process checks whether the location is covered by the calculated shortest-path. It will detect that this is not the case and there is a deviation after node 10. Consequently the encoder will determine the line from node 10 to 11 to become a new intermediate location reference point. The shortest-path up to this line covers its corresponding part of the location completely.

The remaining part of the location matches the shortest path between the two relevant lines from node 10 to 11 and 7 to 5. The location is therefore completely covered by the two calculated shortest paths.

The two identified sub paths that cover the location are used in the next step to define the location reference points. Figure 63 shows the lines being selected for the location reference points. The first location reference point points to the line from node 5 to 8 and indicates the start of the location reference path, the second one points to the line from node 10 to 11, this line was necessary to avoid the deviation from the location path. The last location reference point points to the line from node 7 to 5 and indicates the end of the location reference path.

The last step of the encoding process will generate the binary and XML representation of the location reference.

Binary location reference

The location reference consists of three location reference points. Table 163 shows the coordinates for nodes 5 and 10. These nodes are the corresponding nodes to the location reference points. In preparation of the binary format this table also shows the relative coordinates. Node 5 corresponds to the location reference point 1 and will have coordinates in absolute format. Node 10 corresponding to location reference point 2 will have relative coordinates to the location reference point 1.

Node ID	LRP index	Longitude	Latitude	Relative longitude	Relative latitude
5	1 and 3	6.12829°	49.60597°	--	--
10	2	6.12838°	49.60398°	9	-199

Table 163: Example coordinates

Table 164 collects the relevant data for each location reference point. This includes the functional road class, the form of way and the bearing of the corresponding line. The required information about the path between two subsequent location reference points is also shown (lowest functional road class and distance to the next location reference point).

LRP index	FRC	FOW	BEAR	LFRCNP	DNP
1	FRC2	MULTIPLE_CARRIAGEWAY	134°	FRC3	246
2	FRC3	SINGLE_CARRIAGEWAY	227°	FRC7	475
3	FRC2	SINGLE_CARRIAGEWAY	239°	--	--

Table 164: Location reference points determined during encoding

These tables above hold all relevant information for creating the binary data. The following tables outline the binary data according to section 8:

Bit	7	6	5	4	3	2	1	0
Description	RFU	ArF1	PF	ArF0	AF	Version		
Value	0	1	0	1	1	0	1	1

Table 165: Binary example: status byte

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	1	0	1	0	0	0	0	0

Table 166: Binary example: LRP 1 – absolute longitude

Byte	First								Second								Third							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	1	0	0	0	1	1	0	1	0	0	0	1	1	0	0	1	1	1	1	1	1	0

Table 167: Binary example: LRP1 – absolute latitude

Bit	7	6	5	4	3	2	1	0
Description	RFU	RFU	FRC			FOW		
Value	0	0	0	1	0	0	1	0

Table 168: Binary example: LRP1 – attribute 1

Bit	7	6	5	4	3	2	1	0
Description	LFRCNP			Bearing				
Value	0	1	1	0	1	0	1	1

Table 169: Binary example: LRP1 – attribute 2

Bit	7	6	5	4	3	2	1	0
Description	DNP							
Value	0	0	0	0	1	0	0	0

Table 170: Binary example: LRP1 – attribute 3

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Table 171: Binary example: LRP2 – relative longitude

Byte	First								Second							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Value	1	1	1	1	1	1	1	1	0	0	1	1	1	0	0	1

Table 172: Binary example: LRP2 – relative latitude

Bit	7	6	5	4	3	2	1	0
Description	RFU	RFU	FRC			FOW		
Value	0	0	0	1	1	0	1	1

Table 173: Binary example: LRP2 – attribute 1

Bit	7	6	5	4	3	2	1	0
Description	LFRCNP			Bearing				
Value	1	1	1	1	0	1	0	0

Table 174: Binary example: LRP2 – attribute 2

Bit	7	6	5	4	3	2	1	0
Description	DNP							
Value	0	0	0	0	0	1	0	0

Table 175: Binary example: LRP2 – attribute 3

The encoded data for the last logical LRP is reduced to only those attributes that differ from LRP 1, since this is mainly the same point but only a different referenced line.

Bit	7	6	5	4	3	2	1	0
Description	RFU	RFU	FRC			FOW		
Value	0	0	0	1	0	0	1	1

Table 176: Binary example: LRP3 – attribute 1

Bit	7	6	5	4	3	2	1	0
Description	RFU			Bearing				
Value	0	0	0	1	0	1	0	1

Table 177: Binary example: LRP3 – attribute 7

A Base64 encoded representation of the binary data is the following string:

WwRboCNGfhJrBAAJ/zkb9AgTFQ==

XML location reference

The location reference consists of the first two location reference points and those attributes of the last logical LRP that differ from LRP 1 since this is mainly the same point but only a different reference line.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<OpenLR xmlns="http://www.openlr.org/openlr">
    <LocationID>closedLine</LocationID>
    <XMLLocationReference>
        <AreaLocationReference>
            <ClosedLineLocationReference>
                <LocationReferencePoint>
                    <Coordinates>
                        <Longitude>6.12829</Longitude>
                        <Latitude>49.60597</Latitude>
                    </Coordinates>
                    <LineAttributes>
                        <FRC>FRC2</FRC>
                        <FOW>MULTIPLE_CARRIAGEWAY</FOW>
                        <BEAR>134</BEAR>
                    </LineAttributes>
                    <PathAttributes>
                        <LFRCNP>FRC3</LFRCNP>
                        <DNP>246</DNP>
                    </PathAttributes>
                </LocationReferencePoint>
                <LocationReferencePoint>
                    <Coordinates>
                        <Longitude>6.12838</Longitude>
                        <Latitude>49.60398</Latitude>
                    </Coordinates>
                    <LineAttributes>
                        <FRC>FRC3</FRC>
                        <FOW>SINGLE_CARRIAGEWAY</FOW>
                        <BEAR>227</BEAR>
                    </LineAttributes>
                    <PathAttributes>
                        <LFRCNP>FRC7</LFRCNP>
                        <DNP>246</DNP>
                    </PathAttributes>
                </LocationReferencePoint>
                <LastLine>
                    <FRC>FRC2</FRC>
                    <FOW>SINGLE_CARRIAGEWAY</FOW>
                    <BEAR>239</BEAR>
                </LastLine>
            </ClosedLineLocationReference>
        </AreaLocationReference>
    </XMLLocationReference>
</OpenLR>
```

13.3.5.2 Decoding

Step 1 of the decoding process extracts the physical data (binary format) and checks the validity of the location reference.
The information extracted from the binary data example is shown in Table 178 and Table 179.

LRP index	Longitude	Latitude
1	6.1283°	49.60596°
2	6.12839°	49.60397°
3	6.1283°	49.60596°

Table 178: Decoded coordinates

LRP index	FRC	FOW	Bearing	LFRCNP	DNP
1	FRC2	MULTIPLE_CARRIAGEWAY	123.75° - 135.00°	FRC3	234.4m – 293.0m
2	FRC3	SINGLE_CARRIAGEWAY	225.00° - 236.25°	FRC7	468.8m – 527.4m
3	FRC2	SINGLE_CARRIAGEWAY	236.25° - 247.50°	--	0m

Table 179: Decoded LRP information

This information is sufficient to resolve the location on the decoder map shown in Figure 64. The map consists of 13 nodes and 24 lines (two-way lines are counted twice).

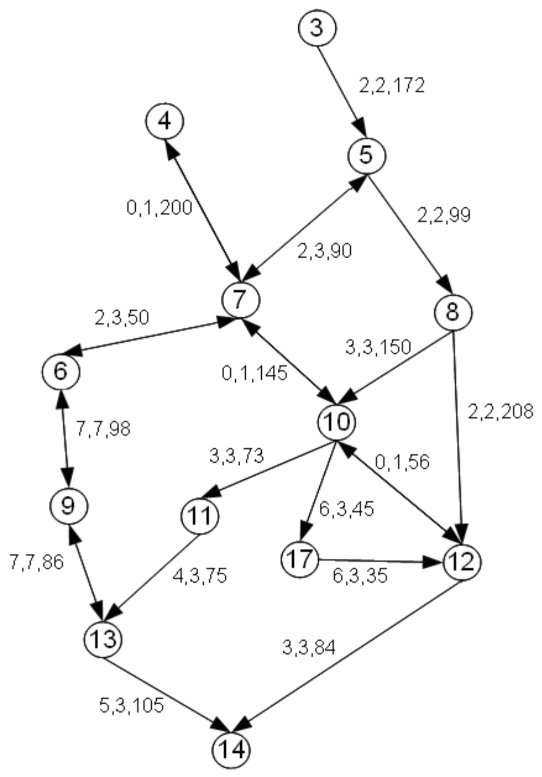

Figure 64: Decoder map

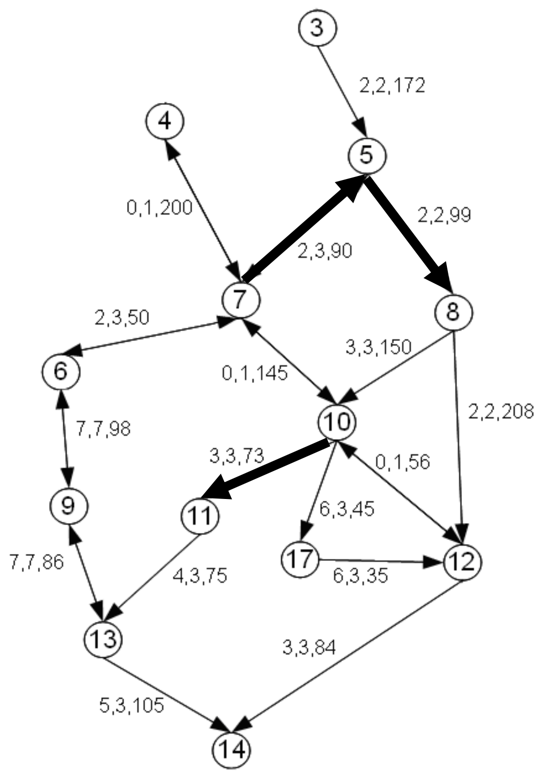

Figure 65: Location reference points

The bold lines in Figure 65 will be determined by the decoder as best fitting to the attributes of the location reference points.

The shortest-path calculation completes the identification of the location in the decoder map. The first shortest path (from node 5 to node 10) is valid with its length of 249 meters compared to the *distance to next point* value of the first location reference point (234.4 – 293.0 meters). The length of the second part of the location defined by the shortest-path from node 10 back to node 5 is 472 meters. This value also fits to the *distance to next point* interval of the location reference point 2 (468.8 meters – 572.4 meters).

Figure 66 shows the detected location in the decoder map after connecting the shortest paths between the location reference points.

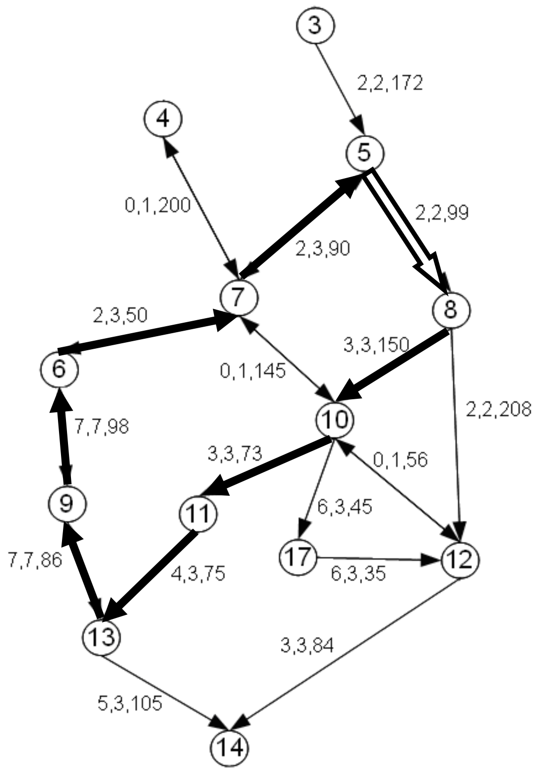

Figure 66: Decoded location

Chapter I

Appendix

List of figures

List of tables

List of equations

An open standard for encoding, transmitting and decoding location references in digital maps

Internet: http://www.openlr.org

Contact:

TomTom (Legal Department)
Email: legal@tomtom.com

TomTom (Technical contact)
Email: openlr@tomtom.com

Address: TomTom International B.V.
Oosterdoksstraat 114
1011DK Amsterdam
the Netherlands