

Cognitive OpenLS Specification

Stefan Hansen, Alexander Klippel, Kai-Florian Richter



SFB/TR 8 Report No. 012-10/2006

Report Series of the Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition
Universität Bremen / Universität Freiburg

Contact Address:

Dr. Thomas Barkowsky
SFB/TR 8
Universität Bremen
P.O.Box 330 440
28334 Bremen, Germany

Tel +49-421-218-64233
Fax +49-421-218-98-64233
barkowsky@sfbtr8.uni-bremen.de
www.sfbtr8.uni-bremen.de

COGNITIVE OPENLS SPECIFICATION

October 18, 2006

Stefan Hansen,

CRC-Spatial Information, University of Melbourne / LISAsoft, Melbourne

Alexander Klippel,

CRC-Spatial Information, Department of Geomatics, University of Melbourne

Kai-Florian Richter,

Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition,
Universität Bremen

Acknowledgements

This work results from a diploma thesis at Universität Bremen that has been jointly supervised by the Universities Bremen and Melbourne and is based on ongoing research at both sites. Cognitive OpenLS is currently being implemented at Spatial Information Systems Ltd. / LISASoft, Melbourne.

This work has been supported by the Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition, which is funded by the Deutsche Forschungsgemeinschaft (DFG), and by the Cooperative Research Centre for Spatial Information, whose activities are funded by the Australian Commonwealth's Cooperative Research Centres Programme. OpenLS is a trademark of the Open Geospatial Consortium.

Contents

1	Introduction	7
2	OpenLS	9
2.1	The OpenLS framework	9
2.1.1	XML for location services	10
2.1.2	The OpenLS services	10
2.2	Route directions with OpenLS	11
2.2.1	Route directions in the route service	11
2.2.2	Route directions in the Navigation Service	12
2.3	Data structure of the Navigation Service	12
2.3.1	NavigationResponse	12
2.3.2	Structure of the route directions	13
2.3.3	Data type of a single instruction	14
2.3.4	ManeuverType: Extending the AbstractManeuverType	18
3	Cognitive OpenLS	23
3.1	Approach	23
3.1.1	Cognitively ergonomic route directions	23
3.1.2	Additional Features	24
3.1.3	Technical realisation	24
3.2	Extending RouteManeuverList	24
3.2.1	XStartingManeuverType	25
3.2.2	XEndManeuverType	26
3.3	XManeuverType	28
3.3.1	Deriving from AbstractManeuverType	28
3.3.2	Replacing NextSegment by CurrentSegment	29
3.4	Making route directions more precise	30
3.4.1	Direction model	30
3.4.2	Naming the structure	31
3.4.3	Types representing intersections	31
3.5	Integration of landmarks in OpenLS	34
3.5.1	AbstractLandmarkType	34
3.5.2	Landmarks for orientation at Start and End Point	34
3.5.3	AbstractNElementLMType	36
3.5.4	Abstract1ElementLMType	37
3.5.5	Description of Landmarks	40
3.6	Chunking Route Directions	41
3.6.1	ChunkType	42

3.6.2	AbstractChunkElement	44
4	Examples of Cognitive OpenLS	47
4.1	Example 1: A complete route	47
4.2	Example 2: Spatial chunking	52
4.3	Example 3: Competing branches	55
5	Schema	57

List of Figures

2.1	Data structure of a list of route maneuvers in OpenLS.	14
2.2	Data structure of an abstract route maneuver in OpenLS.	15
2.3	List of possible actions described in a route direction of OpenLS.	16
2.4	Categories of intersections in OpenLS.	17
2.5	Possible directional changes in OpenLS.	18
2.6	Data structure of an extended route maneuver in OpenLS.	19
3.1	Diagram of the extended data structure of a route maneuver list.	25
3.2	This figure depicts the assumed position of the wayfinder at the start of a route. To follow the route the wayfinder has to turn right.	26
3.3	Data structure of a start maneuver and an end maneuver.	27
3.4	The extended data structure of a route maneuver.	29
3.5	List of possible types of intersections.	31
3.6	Data structure of the different types of intersections.	32
3.7	Data structure representing <i>n-Elements</i> landmarks, <i>Starting-PointLMType</i> and <i>EndPointLMType</i>	36
3.8	Data structure representing <i>1-Element</i> landmarks.	39
3.9	Possible spatial relations between a point-landmark and a decision point.	40
3.10	Data Structure supporting Chunking.	42
4.1	Map of the route in example 1 from Ronzellenstrasse 18, 28359 Bremen to Berckstrasse 10, 28359 Bremen. Based on map from <i>www.uk.map24.com</i> from March 2006.	48
4.2	Map of the route in example 2. It is based on a map provided by <i>www.de.map24.com</i> in March 2006.	53
4.3	Map of the route in example 3. It is based on a map provided by <i>www.de.map24.com</i> in March 2006.	56

1 Introduction

Research on wayfinding and route directions in psychology, linguistics, and cognitive science provides ample evidence for principles of good route directions (e.g., [4, 15]). By analyzing human route directions those principles can be extracted that are most successful, in opposition to simply mimic human route direction. In this report we combine existing approaches and our own research. We identified the following three principles that we consider essential for cognitively ergonomic route directions (cf. [7]; see also [13, 19]):

- **Making direction concepts more precise:** depending on the spatial structure of a decision point, describing the action to be performed may differ [10, 11, 14]. For example, a change in direction usually described as “veer right” might turn to “fork right” given that the road ahead forks to the left and right (i.e. there are two possible branches to take, one heading off to the left, one to the right at approximately 45 degrees).
- **References to landmarks:** landmarks are crucial for good route directions [4, 5, 17]. They are used, among others, to identify decision points, to link actions to be performed to decision points, and to provide confirmation information that the correct route is still followed.
- **Subsuming route directions:** subsuming several directions for individual decision points into one higher order direction covering a sequence of decision points with a single instruction is the third important principle of cognitive ergonomic route directions [3, 12]. Examples of instructions employing this *spatial chunking* [12] are “turn left at the third intersection” and “follow the river until the gas station.”

These principles need to be transformed into an adequate data structure to be able to represent the required information for automatically generating cognitively ergonomic route directions. With the Navigation Service the OpenLS specification [16, 1] developed by the Open Geospatial Consortium [18] offers exactly the functionality required for this purpose. This service provides a client with the data necessary to generate route directions.

In this document, we present the specification of Cognitive OpenLS, a data structure that captures these principles, which is based on the OpenLS specification. More details on the motivation, the empirical and computational justification, and the general approach of this line of research can be found elsewhere in our publications, for example, in [7, 10, 14, 19]. In this document, we focus on the data structure itself, i.e. on the XML-specification of Cognitive OpenLS.

The document is structured as follows: in the next chapter we introduce the organization of OpenLS as defined by the OGC. Chapter 3 details how

the cognitive principles required for cognitive ergonomic route directions are integrated in the OpenLS specification, i.e. presents our extension to OpenLS; Chapter 4 shows some examples of specifying wayfinding situations using Cognitive OpenLS. Chapter 5 then lists the XML schema that defines Cognitive OpenLS.

2 OpenLS

In this chapter, we detail the OpenLS specification as provided by the OGC and its functionality. Generally, our work aims at integrating cognitive ergonomics and automatically generated route directions. The Open Location Services specification [16, 1] offers an open interface to a location based server. Extending this approach with the aspects of spatial cognition as discussed in Chapter 1, allows for specifying a system capable of providing automatically generated route directions that adhere to principles of cognitive ergonomics.

2.1 The OpenLS framework

Location based services are usually provided over a network, which offer users information depending on their current location. The functionalities available are primarily designed to meet the requirements of mobile devices, which are assumed to be used for accessing the services. Examples for these services are providing the address of a user's current location (reverse geocode services), points of interest in the closer environment (directory services), or navigation and routing information (route or navigation services).

Location based services are often regarded as one of the main applications in mobile networks. Therefore, the Open Geospatial Consortium proposed an implementation specification, which describes the Open Location Services (OpenLS), also known as the GeoMobility Server (GMS). This standard specifies an open platform for location based services. A Server implementing this platform according to the specification can be accessed by any client that supports OpenLS.

The OpenLS specification describes the basic services of a GMS (directory service, gateway service, location utility service, presentation service, route service), their basic functionality, the communication between client and server, the abstract data types (ADTs) and the relationship of OpenLS to other specifications and standards. The provided services are described in Section 2.1.2.

The OpenLS standard consists of a set of specifications of interfaces (defined as XML schemas), which determine access to the basic services of such a server and the abstract data types used in the documents that are exchanged between server and client. OpenLS defines primarily the interaction between client and server (request and response schemas) and the format in which the transferred data is encoded. This way, the functionality a server implementing OpenLS has to offer are already implicitly defined.

Since May 2005 OpenLS is available in version 1.1 with status *final*. Recently the OGC has formed a group, which will develop the next version of the OpenLS specification.

2.1.1 XML for location services

The main part of the OpenLS specification comprises the definition of the XML-based markup language *XML for Location Services (XLS)*. The documents defined as request and response schemas and exchanged in communication between client and a GMS are encoded in XLS.

XLS is a markup language defined in a XML-schema. It defines the structure and application of the XML-documents, which are transferred as requests and responses for the client/server interaction, and it specifies the abstract data types (ADTs) (e.g., there is an ADT for an address, a location, a route) used by the OpenLS services for encoding information. The ADTs representing route directions are discussed later in more detail. These are extended with the cognitive ergonomic aspects to become Cognitive OpenLS.

2.1.2 The OpenLS services

The basic service framework of a GeoMobility Server comprises the five so-called core services. These services are described in [16]. In an additional document [1], a sixth service, the Navigation Service, is specified. This Navigation Service does not belong to the core services. In the following section, the functionality of these service is briefly described.

Core services

- **Part 1: Directory Service** offers access to an online directory (such as the Yellow Pages). It allows for finding a specific or a nearest place, a requested product or service.
- **Part 2: Gateway Service** is an interface between a GMS and a Location Server for requesting the location of one or more mobile devices.
- **Part 3: Location Utility Service** (Geocoder/Reverse Geocoder) returns to a given position the complete normalized description of a feature location (place name, street address, postal code) or given a feature location (place, street or postal code) it provides the position and the complete normalized description.
- **Part 4: Presentation Service** The Presentation Service portrays a map based on any geographic information for display on a mobile terminal.
- **Part 5: Route Service** offers routes and additional navigation information.

Navigation service

The Navigation Service is an enhanced version of the Route Service. It is not part of the core services of the OpenLS specification and is described in a separate document [1].

The Navigation Service supports the same parameters as the Route Service and thus, offers the same functionality. Additionally, it extends the Route Service by adding special parameters for navigation purposes. This allows a client providing a user with more detailed and elaborate route directions. It introduces a special data structure for route directions. The information encoded in such ADTs can be used by OpenLS based applications to produce more specific and appropriate route directions.

2.2 Route directions with OpenLS

The OpenLS specification offers two different possibilities for providing route directions. The first is based on verbal instructions encoded as simple strings and supported by both services dealing with routes, namely the Route Service and the Navigation Service. The second possibility, which is only part of the Navigation Service, provides for each instruction a complex object. This can be used by a client to generate directions according to the user's preferences. In the following section, both methods are described in greater detail and their advantages and disadvantages are discussed.

2.2.1 Route directions in the route service

The functionality of the Route Service focuses on calculating a route between two or more points and presenting the requested route as an overlay over a map of the surrounding environment. Therefore, it offers the subscriber several parameters for specifying the characteristics of the route (e.g., different route preferences like fastest, shortest or most scenic route). Additionally to this overlay, the Route Service provides the user with simple route directions. A single direction in the Route Service consists of:

- a string containing the instruction itself,
- the geometry of the route elements, where the described action takes place, encoded in GML,
- a bounding box bordering the next area of interest,
- the travel duration and
- the actual length of the current route segment.

For more user specific route directions, subscribers can indicate their preferred language. Optionally, the text-format of the directions can be set by a parameter. The default value is *text/plain*. Other options for setting this parameter are not specified in the OpenLS standard.

The route directions of the Route Service provide only verbal instructions. Multimodal directions including, for example, visualizations in form of pictograms describing the required action can only be generated by the accessing

application. They are not specifically supported by the transferred data. Therefore, the given route directions are, as far as the verbal instructions are concerned, not flexible at all and a client cannot adapt them according to cognitive aspects.

The verbal instructions are generated on the server and transferred as an (arbitrary) string. Based on this specification, it is not possible to decide whether the provided route directions regard the aspects of cognitive ergonomic route directions discussed in Chapter 1. Furthermore, encoding instructions in simple strings might be sufficient for providing route directions respecting for some of the cognitive ergonomic aspects. However, this way it is impossible to ensure their integration in route directions presented to a user.

2.2.2 Route directions in the Navigation Service

The Navigation Service offers two possibilities of encoding route directions. Since it provides all parameters and all functions which are available in the Route Service, the first possibility is the verbal route directions described in the previous section. The second possibility is the generation of complex objects that describe each route direction in more detail.

The provided objects contain preprocessed information which are necessary for generating route directions. This information needs to be further processed by a client to be presentable to a human user. The application accessing this service has to create its own route directions based on the provided data. The advantage of transferring objects for each route direction rather than complete instructions is that the client application can generate route directions that regard the users' individual preferences and the technical constraints of the client system (e.g., display size or processing speed).

The parameters and information provided by the Navigation Service in an XLS-element representing route directions are described in detail in the next section. The abstract data types and information provided by elements of these types are explained.

2.3 Data structure of the Navigation Service

As already mentioned, in OpenLs communication between client and server is realized as request and response pairs. In this section, we detail the data structure and parameters of a Navigation Response. We focus on the elements required to generate route directions on the client side.

2.3.1 NavigationResponse

For each service exists a request / response pair specified as a XML-schema. In the following, alle elements of a *NavigationResponse* are explained. The focus is placed on the response for a requested list of travel maneuvers.

NavigateResponseType

The document containing a server's answer to a requested navigation service consists of a XLS-document containing the type *NavigationRequestType*. For each requested feature of the Navigation Service it contains an element which provides the appropriate information. A *NavigateResponseType* comprises specific elements containing the requested information and elements that are common for all types of responses (e.g., for error handling). All elements providing the actual information are optional. This, for example, allows for transferring only error codes in the case that an error occurred. The elements which are specific for a Navigation Response are:

RouteHandle This optional element contains a reference to the requested route on the server. It allows the client to request additional information about the route or an alternate route.

RouteSummary In a *RouteSummary* the route's overall characteristics are specified. The contained information consists of the estimated travel time, the whole distance the traveller has to cover for following the route and a bounding box bordering the area in which the route is located. The use of this element is optional.

RouteGeometry The geometry of the route is provided by this optional element. It contains the coordinates of the relevant parts of the roads belonging to the route encoded in *GML*.

RouteInstructionsList The *RouteInstructionsList* provides a list of turn-by-turn route instructions. It encodes the instructions according to the route directions used in the Route Service (cf. Section 2.2.1). The use of this element is optional.

RouteMap Each of these optional elements contain a map of a requested area. The map is transferred as a binary image encoded as base64 [8]. The number of provided maps is unbound.

RouteManeuverList The *RouteManeuverList* contains the route directions generated according to the more complex method used in the Navigation Service. The use of this element is optional.

2.3.2 Structure of the route directions

The more complex route directions offered by the Navigation Service are provided in the element *RouteManeuverList* of the complex type *RouteManeuverListType*, which is derived from the abstract type *AbstractRouteManeuverListType*. Their structure (cf. also Fig. 2.1) is described in the following:

AbstractRouteManeuverListType

This abstract type is the parent type of *RouteManeuverListType*, which is used for storing route directions. It represents a simple list of so-called travel

maneuvers. A travel maneuver comprises the description of a decision point including the required action and the following route segment. Therefore, such a list consists a sequence of an unbounded number of *Maneuver* elements, with every element representing a single maneuver. The *Maneuver* elements in the list are of the type *AbstractManeuverType* that specifies a single travel maneuver.

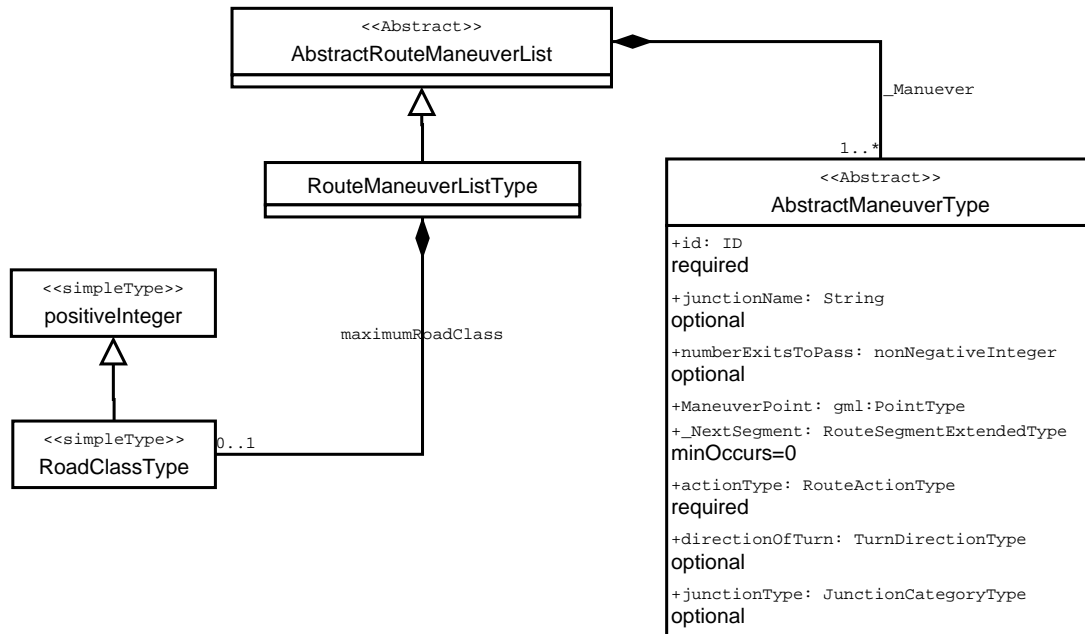


Figure 2.1: Data structure of a list of route maneuvers in OpenLS.

RouteManeuverList

The *NavigationResponse* for a requested navigation service provides a *RouteManeuverList*. For each decision point along the route there is one instruction describing the action the traveller has to perform. A *RouteManeuverList* contains all these travel maneuvers stored in a list.

A *RouteManeuverList* element is of the complex type *RouteManeuverListType*, which is an extension of *AbstractRouteManeuverListType* and provides all parameters of its parent. The extension adds the optional attribute *maximumRoadClass* (*RoadClassType*), which provides the number of levels in the road-class hierarchy. This hierarchy ranks the roads according to their relative size or importance in the route.

2.3.3 Data type of a single instruction

In this section, the data type representing a single route direction is described, including an description of the simple and complex types of the used attributes and elements.

AbstractManeuverType

The complex type *AbstractManeuverType* (cf. Fig. 2.2) is used for storing information about a single route instruction describing the action which has to be performed at a decision point and the decision itself.

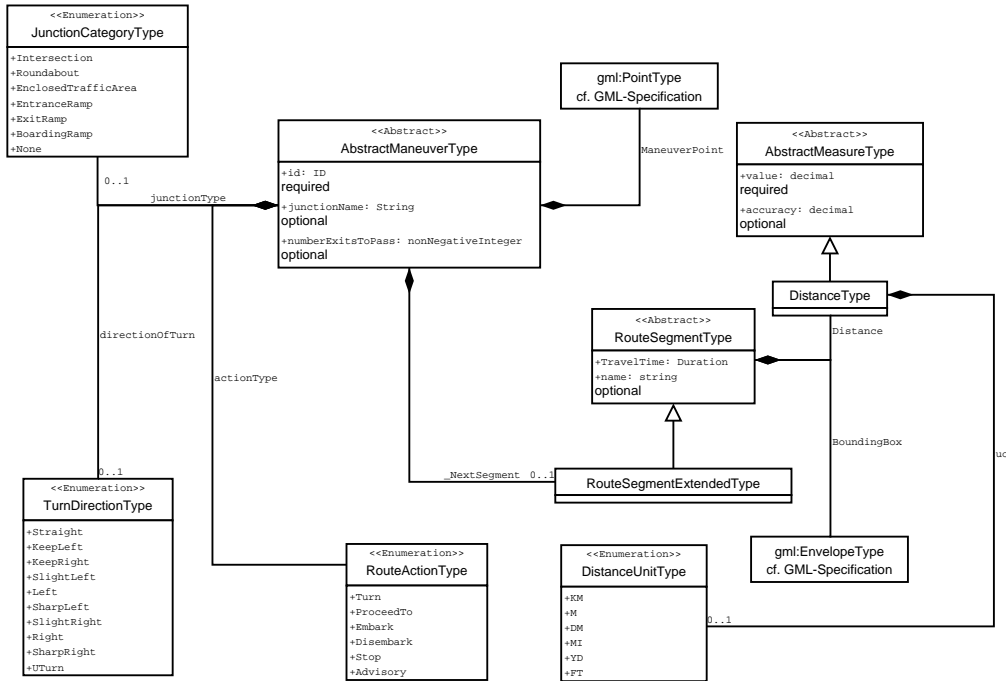


Figure 2.2: Data structure of an abstract route maneuver in OpenLS.

For this purpose an *AbstractManeuverType* contains the following elements and attributes:

Elements:

ManeuverPoint The coordinates of the location where the described action takes place is stored in this attribute which contains an element of the complex type *PointType*. This type is an element of the *GML*-specification [2]. The encoding of coordinates is not discussed in this work.

_NextSegment Information about the next segment of the route is contained in this optional element of the type *AbstractRouteSegmentType*.

Attributes:

id Each instruction in OpenLS has a unique identifier (id). It is of type *ID*.

junctionName Encoded as a simple *string*, the name of the current intersection is provided by this optional attribute.

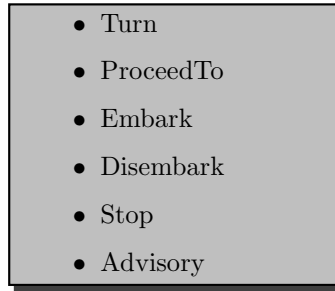
- 
- Turn
 - ProceedTo
 - Embark
 - Disembark
 - Stop
 - Advisory

Figure 2.3: List of possible actions described in a route direction of OpenLS.

junctionType The type of the current intersection is given by this optional attribute. It is of the simple type *JunctionCategoryType* (cf. Section 2.3.3).

numberExitsToPass This optional attribute of the type *NonNegativeInteger* is only used in conjunction with two of the categories of intersections, which are roundabout and complex intersections, and gives the number of exits the traveller has to pass before leaving a roundabout or a complex intersection. However, only the category *Roundabout* can be found in the enumeration of the simple type *JunctionCategoryType*. A complex intersection is not further mentioned or described in the OpenLS specification.

actionType Each instruction describes an action stored in this attribute of the simple type *RouteActionType* (cf. Section 2.3.3 and Fig. 2.3).

directionOfTurn If *actionType* takes the value *Turn* this attribute of the simple type *TurnDirectionType* is used to specify the turn direction. If *actionType* has a different value *directionOfTurn* should not be used.

AbstractRouteSegment

A route direction in OpenLS always includes data about the next route segment the traveller needs to follow after passing the current decision point.

This data is provided by elements of the type *AbstractRouteSegment*. They contain elements that specify the distance of the segment, the estimated travel time, and a bounding box enclosing the segment. Optionally such an element can also provide the name of the described segment.

RouteActionType

For specifying the action the traveller has to perform at a decision point, OpenLS introduces several categories of describing the action. Hence, every route direction provides an attribute of the simple type *RouteActionType*. It enumerates the possible action categories. The values an attribute of this type can take and their meaning according to the documentation of OpenLS are:

- **Turn:** Instructing the traveller to enter the next route segment
- **ProceedTo:** Directing a wayfinder to enter a segment without specifying the action (e.g., used for the first route direction)
- **Embark:** Instructing a wayfinder to enter, for example, public transport
- **Disembark:** Directing a wayfinder to get off, for example, public transport
- **Stop:** Notifying a wayfinder that she arrived at a stopping point
- **Advisory:** Used for instructing a wayfinder to keep on following the current route segment

A special case in this enumeration is the action *Turn*. Only if the attribute of the type *RouteActionType* takes this value, the attribute *TurnDirectionType*, which is described later in this chapter, can be used. For all other values specifying a turn direction is not allowed. However, this constraint is only set by the documentation. A valid *XLS*-document still can combine a turn direction with an attribute of the type *RouteActionType* that has taken another value than *Turn* as it is not restricted in the schema itself.

JunctionCategoryType

Each route direction in OpenLS is categorized according to the type of intersection at which the described action has to be performed. Eight categories of intersections are introduced (cf. Fig. 2.4). Each route maneuver contains an attribute of the simple type *JunctionCategoryType*, which enumerates the possible types of intersections.

- | | |
|-----------------------|----------------|
| • Intersection | • ExitRamp |
| • Roundabout | • ChangeOver |
| • EnclosedTrafficArea | • BoardingRamp |
| • EntranceRamp | • None |

Figure 2.4: Categories of intersections in OpenLS.

These categories distinguish between general intersections, roundabouts, and enclosed traffic areas, which are described in the specification as “an area in which unstructured traffic movements are allowed” [1, p. 22], entrance ramps to highways/motorways, the according exit ramps, change-overs between highways/motorways, boarding ramps on public transport (e.g., a ferry), and, if the route direction does not describe an action at an intersection (e.g., if there is just a road name change or the instruction contains only a travel advisory) the attribute can take the value *none*.

TurnDirectionType

If a route instruction describes a turn the traveller has to perform the attribute *directionOfTurn* of the type *TurnDirectionType* is used, which specifies the directional change at the decision point (cf. Fig. 2.5). It is not to be used if the attribute *actionType* takes another value than *Turn* (cf. Section 2.3.3).



Figure 2.5: Possible directional changes in OpenLS.

The simple type *TurnDirectionType* defines an enumeration of all possible descriptions of directional change. The directions are encoded as verbal expressions. It covers an eight-sector direction model (straight, left, right, slight left, slight right, sharp left, sharp right, u-turn) and additionally the directions *keep left* and *keep right*. OpenLS does not specify the actual angular change for each of these direction changes. Therefore, it is not possible to decide on which concrete direction model the specification is based.

2.3.4 ManeuverType: Extending the AbstractManeuverType

Since the complex type *AbstractManeuverType* used in the *AbstractRouteManeuverListType* to describe a single route instruction is declared as abstract, elements which are directly of this type, cannot be created. Therefore, another complex type has to be defined inheriting the characteristics of an *AbstractManeuverType*.

The OpenLS specification offers one non-abstract complex type, which is derived from *AbstractManeuverType* (cf. Fig. 2.3.3). The *ManeuverType* inherits not only the attributes of the *AbstractManeuverType*, but also extends its functionality by adding new attributes and elements. The additional parameters are described in the following sections.

AdvisoryType

One of the main extensions added in the complex type *ManeuverType* is the introduction of the element *Advisory* of the complex type *AdvisoryType*. It allows the server to generate a much more elaborate instruction.

Every advisory is categorised by its attribute of the simple type *AdvisoryCategoryType*. The actual category depends on what the advisory is informing about. The possible categories are:

- **StartLocation:** The start point of the route

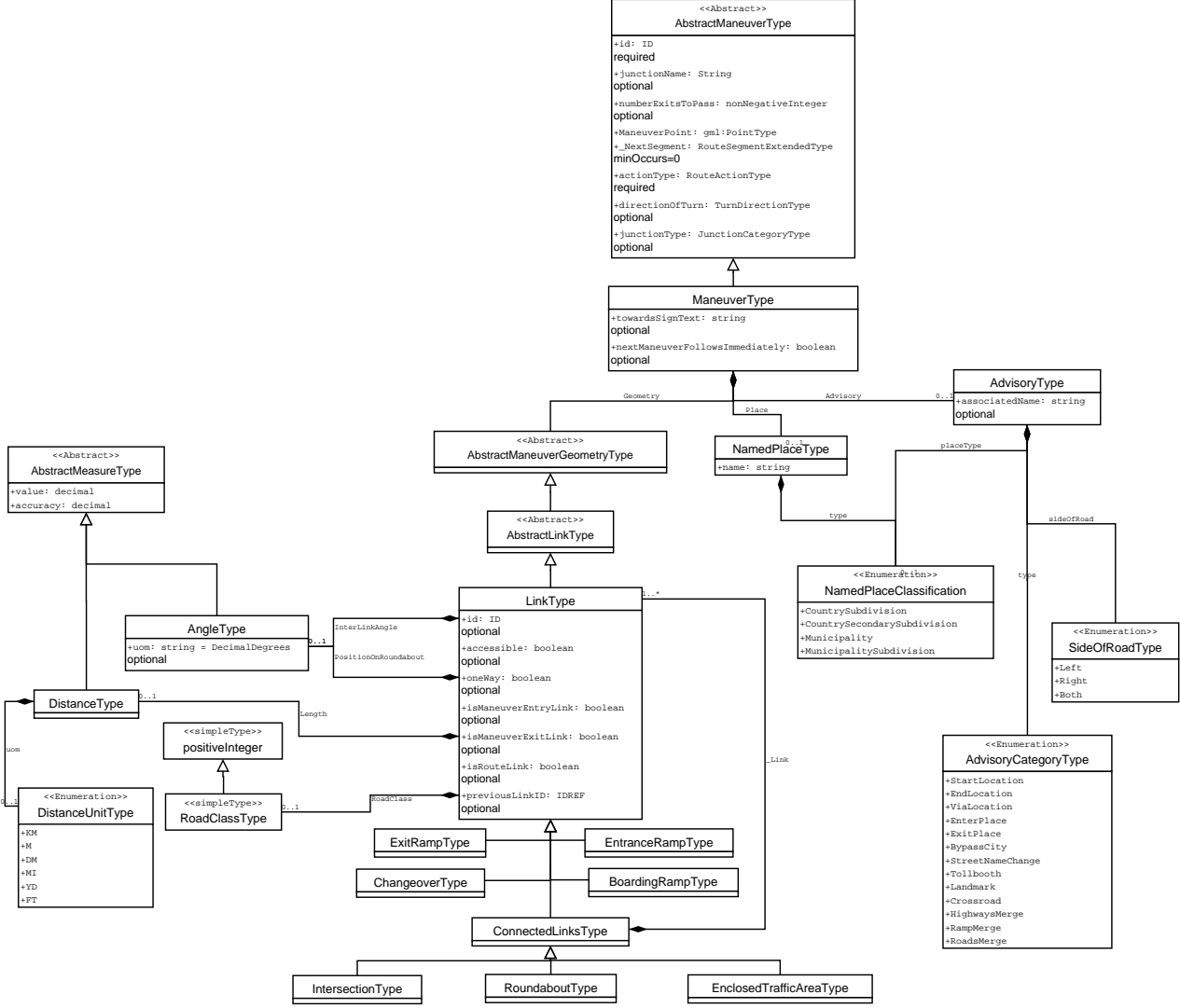


Figure 2.6: Data structure of an extended route maneuver in OpenLS.

- **EndLocation:** The end point of the route
- **ViaLocation:** A special location the route leads through
- **EnterPlace:** Informs a wayfinder that she enters a named place (e.g., a country, state, or city)
- **ExitPlace:** Leaving a named place
- **BypassCity:** Turning on a branch to bypass a city
- **StreetNameChange:** At this location , a change of the street name occurs
- **Tollbooth:** Passing a tollbooth
- **Landmark:** Passing or turning at a landmark
- **Crossroad:** Passing or turning at a crossroad
- **HighwaysMerge:** The highway a wayfinder is currently on merges with another highway
- **RampMerge:** A ramp merges with a current road
- **RoadsMerge:** The road a wayfinder is currently on merges with another road

Furthermore, an advisory is associated with a name (*associatedName*) and contains an optional attribute of the type *NamedPlaceClassification*. This provides the level in a hierarchy defined by comprising the different types: *CountrySubDivision*, *CountrySecondarySubdivision*, *Municipality*, and *MunicipalitySubdivision*. A further explanation of this hierarchy is not given. Additionally, an advisory includes optionally the attribute *sideOfRoad* of the type *sideOfRoad*, which associates the advisory with either one side of the road or both.

namedPlaceType

Similar to an advisory the extended route instructions have an attribute of the type *NamedPlaceClassification*, which stores the level of the hierarchy explained in the previous section.

Geometry

The extended version of the route directions also provides a more elaborate description of the geometry of the current maneuver. This information is given by the attribute *Geometry* of the type *AbstractManeuverGeometryType*. Derived from *AbstractManeuverGeometryType* and *AbstractLinkType*, the complex type *LinkType* consists of several attributes and elements. The general geometry of the link is described by the element *InterLinkAngle* of the type *AngleType*. It states the angle of the outgoing branch relative to the branch

on which the traveller arrives. If the current direction describes an action at a roundabout, the position of the link on the roundabout is given (*PositionOnRoundabout* of the type *AngleType*) and the actual length of the (next) link is provided (*Length*).

Apart from the actual geometric information, the type of the road is provided. It contains the attribute *roadClass* of the type *RoadClassType*, providing the class of the road described by a simple positive integer, a boolean attribute providing the accessibility of the road (*Accessible*) and a boolean attribute storing, whether the road is a one way street or not (*oneway*). All these attributes are optional.

The next group of information provided by this type regards the role of the link in the current route. A link can be an entrance link (*boolean* attribute *isManeuverEntranceLink*), an exit link (*boolean* attribute *isManeuverExitLink*), or a route link (*boolean* attribute *isRouteLink*). The exact meaning of these links is not specified [2, p. 38]. Again, the use of these attributes is optional.

Finally, *LinkType* contains some internal information. Every link has an optional ID (attribute *id* of the type *ID*) and it also provides optionally the ID of the previous link (attribute *previousLinkID* of the type *ID*).

towardsSignText

A *ManeuverType* can contain an optional attribute of the type *towardsSignText*. This type provides a simple string in which the name of the place (usually a city) the route is heading for is given.

nextManeuverFollowsImmediately

The meaning of this boolean attribute is not specified in the OpenLS specification. It is probably used to specify situations where the next action to take follows very shortly after the current action, i.e. a wayfinder has only very short time to prepare for the next action. Its usage is optional.

3 Cognitive OpenLS

The OpenLS standard as defined by the OGC does not support cognitively ergonomic route directions. In this chapter, we propose an extension that integrates precise direction concepts, landmarks and spatial chunking of route directions in the OpenLS specification. The abstract data types of the OpenLS Navigation Service which provide data for the generation of route directions (*AbstractRouteManeuverList*, *AbstractManeuverType* and all related types) are restructured. The changes and their implementation are described in detail in this chapter. The XML specification of our extension is listed in Chapter 5.

3.1 Approach

Since the current version of the OpenLS specification does not support all necessary features for generating cognitively ergonomic route directions, the standard, or more precisely the underlying data model, has to be extended. To this end it is sufficient to adapt some types in the data structure of the navigation service.

The extension replaces the complex types *AbstractRouteManeuverList*, *AbstractManeuverType* and all related types. The proposed data types are integrated in the data structure in the same way the official extension of the *AbstractManeuverType*, the *ManeuverType* are implemented. All new types are derived from the original type.

In the following, the aspects regarded in the extension are explained. It is distinguished between aspects of human ergonomic route directions and additional aspects which are integrated for technical reasons. Furthermore, the approach for the technical realization is briefly discussed.

3.1.1 Cognitively ergonomic route directions

The main concern of the proposed extension of the OpenLS specification is to enable the automatic generation of cognitively ergonomic route directions based on the OpenLS data model. Thus, all aspects discussed in Chapter 1 are regarded:

- **Direction model:** The direction model the encoding of the provided turn direction is based on is adapted from the direction model presented by [9] in combination with naming the structure of the intersection. This is further detailed in Section 3.4.1.
- **Structure of intersections:** Including references to the structure of an intersection is explained in Section 3.4.2.

- **Landmarks:** Landmarks play an important role in giving route directions. We integrate an elaborated model for their description in OpenLS (cf. Section 3.5).
- **Chunking:** The extended version of OpenLS offers the possibility of spatial chunking. The implementation of the combination of Klippel's [12] and Dale's [3] approach is described in Section 3.6.

3.1.2 Additional Features

Besides of the aspects of cognitively ergonomic route directions some additional features are integrated in the OpenLS specification. They make the usage of the data structure easier, clearer and less error-prone.

- **Start point:** The start of a route is a crucial point in a description of a route. Accordingly, it is treated separately in the extended version of the OpenLS specification (cf. Section 3.2.1).
- **End point:** Similar to the start point the end point of route requires special attributes. A special *complex type* providing the necessary attributes is introduced (cf. Section 3.2.2).

3.1.3 Technical realisation

Data about a single instruction in a *Navigation Response* is significantly increased in the new version of the OpenLS specification. Especially the *complex type ManeuverType* has been radically restructured and extended. Therefore, the compatibility to the original version of the *AbstractManeuverType* is not given. A server or client that supports one of the two versions cannot deal with the abstract data types of the other.

3.2 Extending RouteManeuverList

In a list of single route directions, the start point and the end point of the route play a special role. Instructions on the start point helps the wayfinder to orientate and to commence the route in the correct direction. The end point unambiguously identifies the actual destination.

Since start and end point have to be treated separately, they should also be indicated explicitly in the list of the route maneuver. Thus, a complex type *XRouteManeuverListType* is derived from *AbstractRouteManeuverList*, which adds two elements containing information about the start and the end point of a route. The already existing elements of a *AbstractRouteManeuverList*, the unbounded number of elements of the type *AbstractManeuverType*, remain and thus, assure the compatibility of the next extensions to the original abstract version of the OpenLS Navigation Service. Additionally, like in *RouteManeuverList* the attribute *maximumRoadClass* is used.

An UML-diagram of the relationship between the single types, their attributes and elements is shown in figure 3.1. The exact structure of a start and an end point is explained in the following two sections.

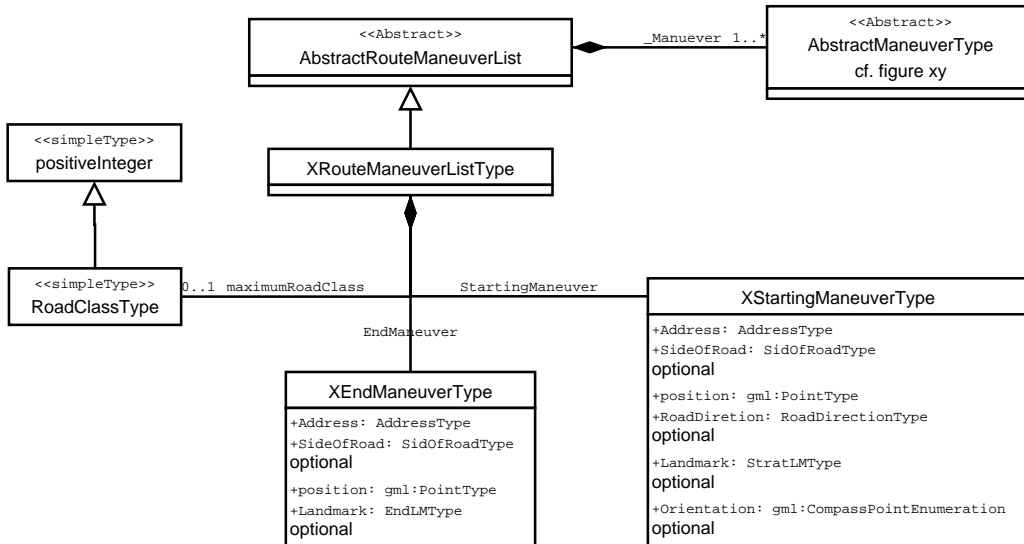


Figure 3.1: Diagram of the extended data structure of a route maneuver list.

3.2.1 XStartingManeuverType

Orientating the wayfinder at the beginning of the route is a crucial part in giving route directions. It has to be assured that the wayfinder follows the first route segment in the right directions. Therefore, the complex type *XStartingManeuver* is introduced providing a set of attributes and elements containing the necessary and available information for the first orientation (cf. Fig. 3.2.1).

Position This element gives the exact geographic position of the start point of the route encoded as an *gml:PointType*.

Address The element *Address* provides the address of the start point of the route. For encoding this information, the complex type *AddressType* of the OpenLS specification is used. It can contain either an unstructured *free form address* (a simple string), a street address or an intersection address and some additional elements (e.g., a post code).

RoadDirection Starting at the address/position, a wayfinder has to follow the first segment of the route. At the start point, facing the road the wayfinder can either turn left or right or he can proceed (in some special cases) straight. Therefore, the attribute *RoadDirection* can be one element of the enumeration of the simple type *RoadDirection*: *left*, *right* or *straight*.

Orientation The complex type *XStartingManeuver* additionally offers an attribute providing the cardinal direction in which the wayfinder has to go. For this purpose the simple type from the GML specification *CompassPointEnumeration* is used. It splits the possible directions into 16 homogeneous sectors. Each sector is represented by a cardinal direction (e.g., south).

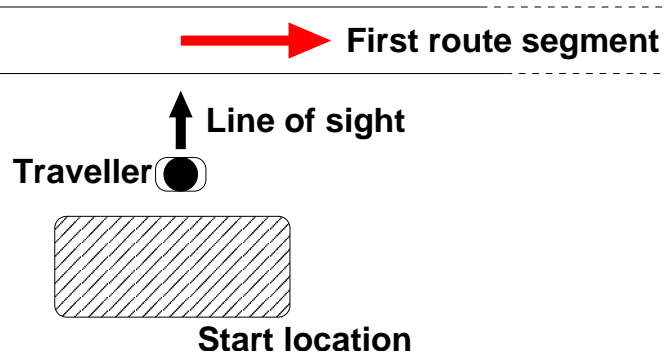


Figure 3.2: This figure depicts the assumed position of the wayfinder at the start of a route. To follow the route the wayfinder has to turn right.

Landmark The third possibility of orientating the wayfinder at the origin of the route is to describe the direction relatively to a landmark. The landmark for this purpose is provided by the element *Landmark*. This element is of the complex type *StartPointLMLMType*. A detailed description of this type can be found in Section 3.5.2).

id A start maneuver has an ID that identifies it unambiguously.

3.2.2 XEndManeuverType

Like the start point of a route, the end point is encoded separately. Its main function is to provide all information in order to enable the wayfinder to identify the destination of her route. Therefore, elements of this type provide the following information (cf. Fig. 3.2.1).

Position The exact geographic position encoded as an *gml:PointType* of the end point of the route is provided by the element *Position*.

Address This element contains the address of the end point of the route. For encoding this information, the XLS complex type *AddressType* is used (cf. section 5.3.1).

SideOfRoad The end point of the route can be located either on the left side, the right side or on both sides of the road to the current direction the user is travelling in. The simple type *SideOfRoad* offers the three values: *left*, *right* and *both*. The attribute *SideOfRoad* of a *EndManeuverType* can take one of them as value.

Landmark For describing the position of the end of the route, a landmark may be used. The spatial relation between the end point and the landmark has to be described, as well as the landmark itself. The necessary features for this purpose are provided by the complex type *EndPointLMType*. An element of the type *EndManeuverType* contains the element *Landmark*, which is of this type.

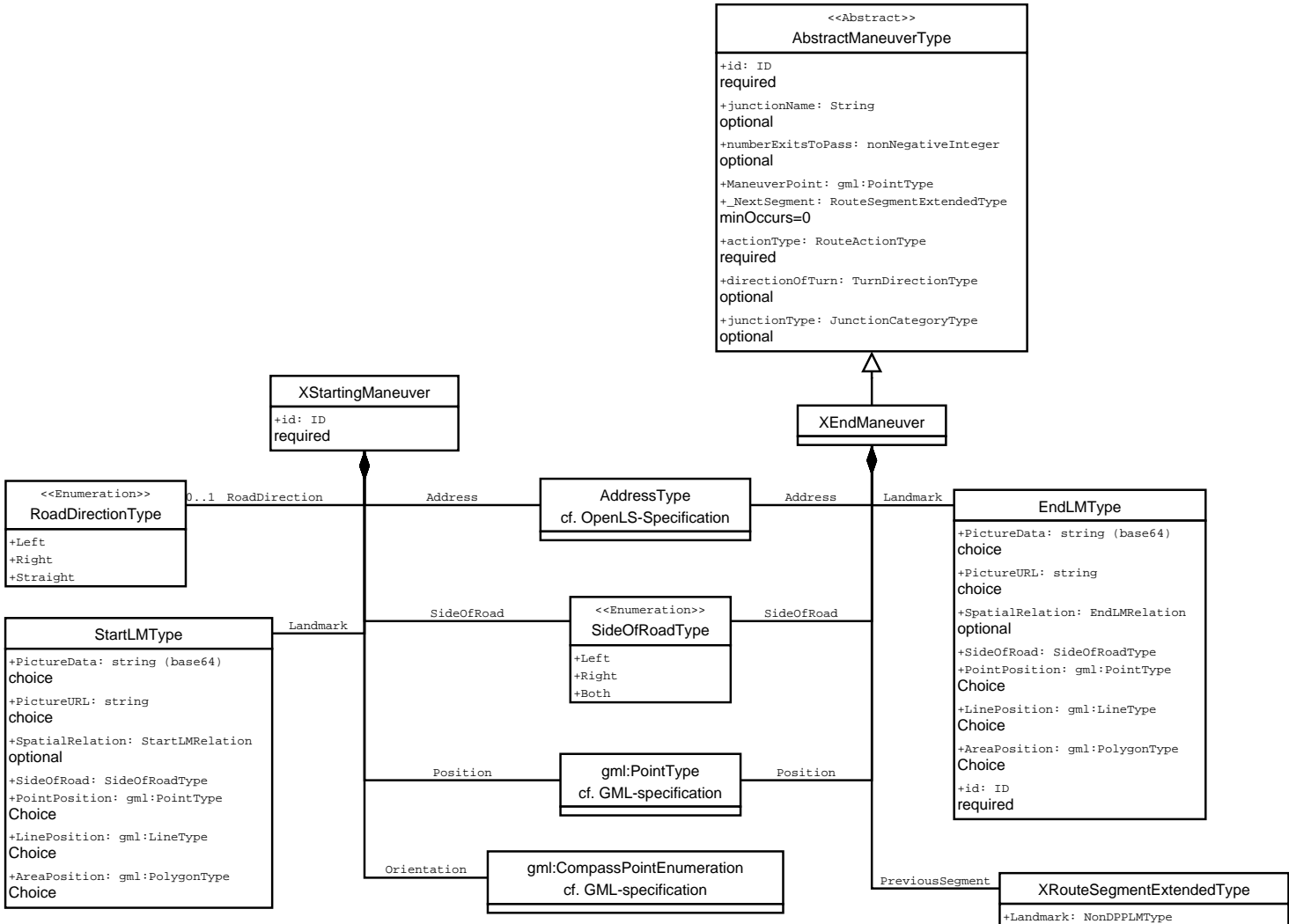


Figure 3.3: Data structure of a start maneuver and an end maneuver.

previousSegment In the data structure, a decision point of a route and the route segment leading to this decision point are stored together. Hence, the last route segment leading to the end point of the route has to be stored together with the end point. The element *previousSegment* of the complex type *XRouteSegmentExtendedType* contains this segment.

id A start maneuver has an ID that identifies it unambiguously.

3.3 XManeuverType

Route directions based on the complex type *AbstractManeuverType* do not support cognitively ergonomic route directions. Also the type *ManeuverType* derived from *AbstractManeuverType* does not offer the necessary features to encode all information for generating the aspects of cognitively ergonomic route instructions. Hence, the extension of the OpenLS Navigation Service introduces a new complex type *XManeuverType* for encoding single route directions, which provides the required elements and attributes (cf. Fig. 3.3.1).

3.3.1 Deriving from AbstractManeuverType

Like the original *ManeuverType* *XManeuverType* is derived from *AbstractManeuverType*. Even though it replaces all of the elements and attributes of the super type except of one, the derivation is necessary to assure the compatibility to the original, abstract version of the Navigation Service. The reused attribute is:

id This attribute encodes an identification number in order to identify the direction unambiguously as an attribute of the simple type *ID*.

The new attributes are:

previousSegment In our extension a decision point and the route segment which leads to it form a unit (e.g., “*Follow street XY and then turn left at the church.*”). This schema is commonly used in human generated route directions and current navigation services. Therefore, the element *previousSegment* of the complex type *XRouteSegmentExtendedType* is introduced.

nextStreet This attribute contains the name of the street the wayfinder is turning in when she performs the described maneuver. The information is also provided by other elements (e.g., the next segment, which is still contained because of the derivation, or the following maneuver), but these should not be used in Cognitive OpenLS.

JunctionCategory Since the element *JunctionType* of the complex type *AbstractManeuverType* does not offer the features required for making route directions more precise, the element *JunctionCategory* is introduced. It is of the complex type *AbstractJunctionType*, which is discussed in Section 3.4.3.

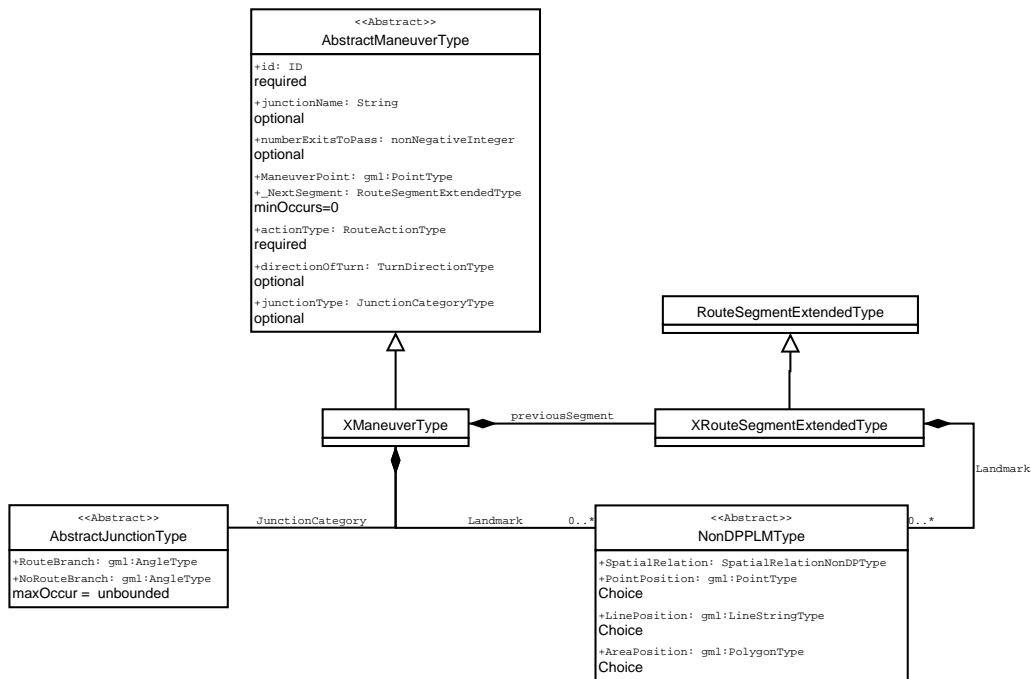


Figure 3.4: The extended data structure of a route maneuver.

Landmarks play an important role in the understanding of route directions by humans. Hence, elements providing information about landmarks are introduced. Decision points can be related to *1-Element* landmarks. *n-Elements* landmarks can only be associated with chunks. Since a decision point can have more than one landmark, the amount of elements representing a landmark is unbounded.

Landmark This element of the abstract complex type *Abstract1ElementLMType* provides all required information about a landmark with a point-like function. In principle, the number of landmarks associated with a decision point is unbounded. The attributes and elements of *AbstractPoint-LikeLMType* and its sub-types are described in Section 3.5.

3.3.2 Replacing NextSegment by CurrentSegment

In an elementary route direction always (apart from the start point) a decision point and the previous route segment are combined. The route directions generated on this basis generally follow the schema:

“Follow street XY and turn right after the landmark.”

This schema is also quite commonly used by humans giving route instructions, as well as in instructions generated by navigation services.

In an element based on the complex type *AbstractManeuverType* of the original version of the Navigation Service, a decision point is combined with the

following route segment, rather than with the previous segment. However, it is possible to restructure the instructions and combine a decision point with the following decision point. Since it has to be possible to relate a route segment to a *1-Element* landmark, a new type providing the necessary element for storing the information about a landmark is introduced.

The complex type *XRouteSegmentExtendedType* is derived from the original XLS complex type *RouteSegmentExtendedType* and therefore, contains the same attributes and elements. In order to enable the use of landmarks, new elements are introduced:

Landmark The information about a landmark with a point-like function is stored in this element of the type *NonDPPLMType*. A segment can be related to an unbounded number of landmarks of this type.

RoadNameChange This attribute of the type *RoadNameChangeType* is also introduced in this extension and provides the necessary information if the street name changes at a segment and not at a decision point. The complex type *RoadNameChangeType* contains the new street name encoded as string and the position of the change as *gml:PointType*.

Streetname This attribute stores the name of the street to which the segment belongs. In contrast to the name provided by *RouteSegmentExtendedType*, this attribute is required.

3.4 Making route directions more precise

According to the aspects of cognitively ergonomic route directions our OpenLS extension introduces data types, which describe the spatial situation at a decision point as exactly as necessary and combines this with a direction concept specifying the turn direction.

3.4.1 Direction model

In the current version of the Navigation Service, a direction is described by a verbal expression. OpenLS offers a collection of ten verbal expressions in a simple data type. How the underlying direction model structures the space is not further explained.

Similar to the original verbal expressions, the proposed extension of the specification offers simple types for different kinds of intersections providing an appropriate direction concept and its respective direction. Relating structure of the intersection and appropriate directions constrains insensible combinations.

The directions for each type of intersection are encoded as enumerations of verbal expressions. However, these verbal expressions are not necessarily to be used in the route directions finally presented to a user. The verbal expressions are only used in order to make the encoding of the data more readable for humans.

The directions given in the instructions are all constrained by the type of the intersection. These types require only the directions *straight*, *left* and

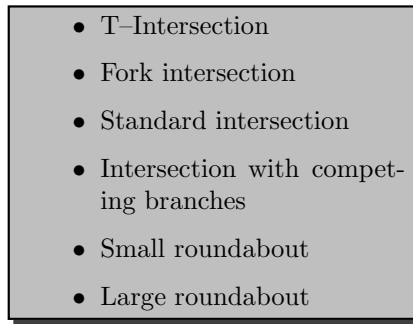
- 
- T-Intersection
 - Fork intersection
 - Standard intersection
 - Intersection with competing branches
 - Small roundabout
 - Large roundabout

Figure 3.5: List of possible types of intersections.

right. The definition of *left* and *right* is intelligible; the directions are divided by the straight axis. Directions in the left half are represented by *left*, the directions on the right side by *right*. However, it is not exactly specified which directions should be represented by *straight* and subject to ongoing research.

3.4.2 Naming the structure

Describing the spatial situation at an intersection helps the wayfinder to orientate herself and allows to simplify the given turn directions. For example, specifying an intersection as T-intersection reduces the possible directions to the two options *left* and *right*.

An element representing an intersection provides the spatial situation by listing all branches. This allows for producing, for example, a pictorial representation of the intersection. Furthermore, all intersections are classified by standard types. The offered types of intersections are listed in figure 3.5.

3.4.3 Types representing intersections

The elements of the current OpenLS version do not provide all features required to generate precise route directions and its data structure does not constrain the generation in a way that minimizes the possibility of creating unambiguous instructions. Therefore, several new types for supporting this purpose are introduced (cf. Fig. 3.4.3).

AbstractJunctionType

The proposed extension classifies each intersection into a category. Each category is represented by a complex type derived from *AbstractJunctionType*. *AbstractJunctionType* provides the attribute elements which are necessary for all types of intersections.

name If an intersection has a special name identifying it, the name can be stored in this optional attribute encoded as a simple string.

Encoding the spatial structure For each intersection its spatial structure has to be provided. This allows giving the wayfinder an exact description of the spatial situation and helps her to orientate. Every branch of the intersection has to be listed. The branches are encoded as complex type *Branch*. Introduced in the proposed extension, this type provides information on the angle between branch and the route segment leading to the intersection as *gml:AngleType* and the street name of the branch.

RouteBranchOut The branch on which the wayfinder leaves the intersection must be stored in a separate element, since it has to be possible to distinguish it from the other branches due to its special role.

NoRouteBranch All other branches are stored in an element with the name *NoRouteBranch*. The occurrence of this element is unbounded.

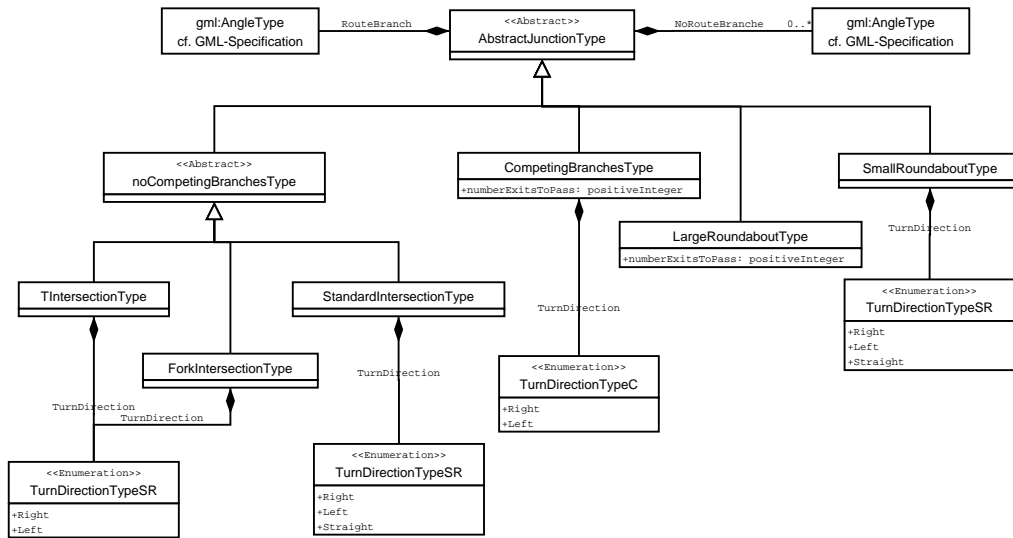


Figure 3.6: Data structure of the different types of intersections.

AbstractNoCompetingBranchesType

Junctions of the type *AbstractNoCompetingBranchesType* can be classified in one of the categories T–intersection, Fork–intersection and standard intersection. For these classes of intersections it is sufficient to know the direction of the turn and the category of the intersection. From this abstract type all three sub–categories are derived.

TIntersectionType T–intersections are intersections where the road on which the wayfinder arrives ends at the intersection. If the wayfinder approaches the same intersection on a different branch, it is not classified as a T–intersection but as a standard intersection. The only attribute of this type contains the

direction of the turn. It is of the simple type *TFTurnDirectionType* and can only take the value *left* or *right*.

ForkIntersection Similar to the T–intersection, the incoming branch of the intersection sets whether it is categorised as fork intersection or not. The wayfinder has to arrive on the branch that splits into the two other branches. Also similar to the T–intersection, it provides only one attribute of the simple type *TFTurnDirectionType*.

StandardIntersectionType All other intersections that have no branch competing with the outgoing branch belong to the category standard intersection. The complex type of this group has only one attribute containing the turn direction. In this case it is of the simple type *SITurnDirectionType* and can therefore take the value *left*, *right* or *straight*.

CompetingBranchesType

The set of possible turn directions is divided by the *straight-axis* in a left and a right part. Two branches of an intersection compete with each other, if they are both either in the right or the left part. A branch going straight cannot compete with another branch.

If at an intersection the outgoing branch competes with another branch, this intersection is represented by the complex type *CompetingBranchesType*. For this type of intersection, a direction concept is used that combines a coarse turn direction (*left* or *right*) with an ordering concept: the competing branches are counted and the appropriate number is chosen for the branch to take. An element of the type *CompetingBranchesType* has two attributes. The *Branch-Number* contains the number in the ordering concept encoded as a positive integer value. *TurnDirection* is of simple type *TurnDirectionTypeC*, which can take either the value *left* or *right*, whereas these values stand for one of the two parts of the space of turn directions.

SmallRoundAboutType

Two different classes exist for roundabouts. Small roundabouts are encoded as elements of the type *SmallRoundaboutType*. These roundabouts are so small that an instruction like “*Turn left*” is comprehensible without problems. The only attribute of this complex type representing an intersection is of the simple type *TurnDirectionType*. An attribute of this type can take a value according to the direction model discussed previously.

LargeRoundaboutType

The second class of roundabouts is for large roundabouts. For large roundabouts the use of turn directions is ambiguous. Hence, the instruction for performing a travel maneuver is to give the number of exits the wayfinder must pass in the roundabout. The only attribute that an element the complex type

LargeRoundaboutType has is a simple integer value, which gives the number of exits to pass at the roundabout.

3.5 Integration of landmarks in OpenLS

Landmarks play an important role in route directions. They can have different purposes and functions, for example, they can be used for identifying segments and decision points, they can help a wayfinder to orientate or they can specify a required action.

According to their different functions, we set up a taxonomy for the classification of landmarks (cf. [7]). This taxonomy is used as a basis for the abstract data types representing landmarks in the proposed extension of the OpenLS specification.

The usage and function of these newly introduced types depends on the context of other types, which are contained as elements. This is discussed in the following in the presentation of the different complex landmark types.

3.5.1 AbstractLandmarkType

All complex types representing a particular type of landmark are derived from the abstract type *AbstractLandmarkType*. This type contains all elements and attributes common to all landmarks.

The elements and attributes of *AbstractLandmarkType* are the name (*Name*) and a description (*description*) of the landmark. The attribute *Name* should contain the official name of the landmark. The *description* should provide all information that is necessary to enable the wayfinder to identify the landmark in the environment.

The use of the attribute *Name* is optional and its content is encoded as a simple string. The element description is of the complex type *AbstractLMDescription*. More information about this type is given in Section 3.5.5.

3.5.2 Landmarks for orientation at Start and End Point

The types *StartingPointLMType* and *EndPointLMType* provide all information about the landmarks used for orientating the wayfinder at start and end of a route.

StartingPointLMType

By choosing the information that is provided by an element representing a start point the situation of a wayfinder has to be regarded. It is to assume that the wayfinder is not familiar with the environment. Therefore, she do not know which direction she has to head to in order to follow the route. She also does not know where the landmark is located and how it looks like. Thus, the complex type *StartingPointLMType* is introduced (cf. Fig. 3.5.2). Additionally to the attribute *Name* and the element *Description*, which are already contained by

the abstract complex type *AbstractLandmarkType*, the following elements are provided:

PointPosition, LinePosition, AreaPosition For the exact geographic position of the landmark, it can be chosen between these three elements. Which one is selected depends on the geometry of the landmark. The position is encoded as *gml:PointType*, *gml:LineStringType* or *gml:PolygonType*.

Orientation Since this type of landmark is used to help the wayfinder to orientate in general without relation to any other elements of the route directions (for example, roads or current travel direction), providing the general orientation of the landmark with respect to the wayfinder's current position is in some cases helpful. Thus, the attribute *Orientation* contains this information encoded as an element of the simple type *gml:Compass-PointEnumeration*.

SpatialRelation In this optional attribute the information whether a wayfinder has to follow the first segment *towards* or *away* from the landmark is stored.

EndPointLMType

In contrast to the start of a route, the wayfinder is already correctly oriented on the last route segment. The task of the last instruction is to support the wayfinder in identifying her final destination. The end point of a route is in most cases a street address, which is usually identified by its house number. Since this number is quite often not visible or not easy to spot for the wayfinder, it is helpful to describe the position of the destination of the route in relation to a much more salient landmark. Therefore, the complex type *EndPointLMType* is introduced (cf. Fig 3.5.2). It provides, additionally to the name and a description of the landmark, the following elements in order to fulfill this task:

PointPosition, LinePosition, AreaPosition For the exact geographic position of the landmark it can be chosen between these three elements. Which one is selected depends on the geometry of the landmark. The position is encoded as *gml:PointType*, *gml:LineStringType* or *gml:PolygonType*.

SpatialRelation The attribute *SpatialRelation* stores the spatial relation between the described landmark and the destination of the route. The relation is encoded as a verbal expression. The possible relations are defined by the simple type *EndLMRelationType*. According to this a landmark used to identify the destination of a route can be located next to the destination (*left* or *right*) or opposite to it.

SideOfRoad Since the wayfinder is following a road, the landmark can be located either on the right side of the road, the left side or on both. This information is stored in this optional attribute.

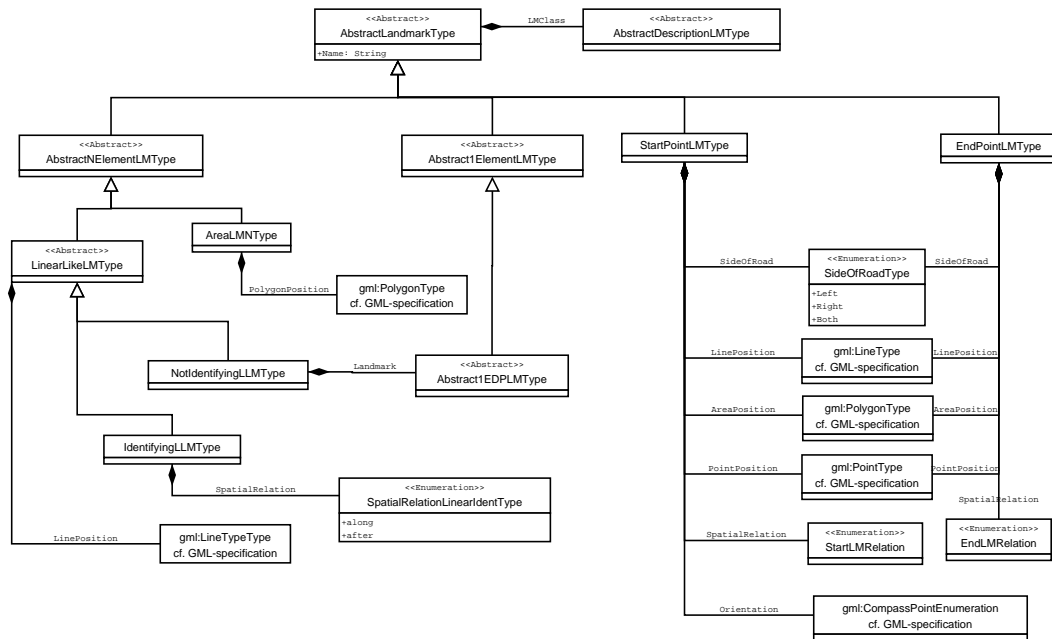


Figure 3.7: Data structure representing n -Elements landmarks, *StartingPointLMType* and *EndPointLMType*.

3.5.3 AbstractNElementLMType

Landmarks related to more than one route element are represented by the abstract class *AbstractNElementLMType* (cf. Fig 3.5.2). Two child-classes representing the two required sub-categories are introduced. *AbstractLineLMType* stores information on landmarks conceptualised as linear and *AreaLMNType* those conceptualised as area-like.

AreaLMNType

Area-like landmarks identifying a chunk of route elements are represented by the complex type *AreaLMNType*. Since there are no types derived from this type, it is not abstract. Landmarks of this type have one additional element:

PolygonPosition For giving the position of the landmark a polygon is provided in this element. The polygon is encoded according to the GML specification as a *gml:PolygonType*.

The spatial relation of a landmark to the related decision point is always *around*. Hence, it is implicitly encoded in the complex type *AreaLMNType*.

AbstractLineLMType

Landmarks with a linear function are located along the route and are therefore related to more than one element of the route. *AbstractLinearLikeLMType* is introduced, which covers with its attributes and elements the characteristics common to all types of landmarks with a linear function.

AreaPosition, LinePosition For the exact position of the landmark it can be chosen between these two elements. Depending on the geometry of the landmark, the position is encoded as *gml:PolygonType* or *gml:LineStringType*.

IdentifyingLLType The crucial part of describing the course of a route with a landmark with linear characteristics is the point where the route stops following the course of the landmark. The kind of landmarks represented by *IdentifyingLLType* do not require reference to an additional landmark to identify this point. It is sufficient to identify the last decision point at which this relation ends by providing the spatial relation to each route element.

SpatialRelation This spatial relation is described by the attribute *SpatialRelation*, which is of the simple type *SpatialRelationLinearIdentType*. This type enumerates the possible relations between the landmark and the route, which are *along* or *after*.

NotIdentifyingLLType Linear objects along the route, which are able to describe the course of the route due to their geometry but are not sufficient to specify where the route stops following this course are described by the complex type *NotIdentifyingLLLType*. Since the landmark does not identify the end of its course with the route, additional information has to be provided describing this end point. For this purpose these line-landmarks contain an additional point-landmark.

Landmark The point where the course of the route stops following the course of the linear landmark is described by an additional landmark with a point-like function. Therefore, the element *Landmark* provides the required additional information encoded as the complex type *AbstractPointLMType*. This type is described later in this chapter.

3.5.4 Abstract1ElementLMType

The category of landmarks related to a single element of the route is divided into two sub-categories: point-landmarks and area-landmarks. The abstract complex type *Abstract1ElementLMType* represents with its two child-classes these two categories (cf. Fig. 3.5.4).

Abstract1EDPLMType

The abstract complex type *Abstract1EDPLLMTType* represents the supertype for all landmarks identifying only one decision point. Using this type allows for introducing in the type *XManeuverType* only one element (of an unbounded occurrence) for all possible types of landmarks. However, since these sub-categories have no common attributes or elements apart from those common for all landmarks, *Abstract1EDPLMType* has no additional attributes or elements.

AreaLM1Type

Landmarks with an area-like function are represented by the complex type *AreaLM1Type*. It is derived from the type *Abstract1EDPLLMTType*. The elements needed to identify a single decision point located within an area-like landmark requires the following attributes:

PolygonPosition For giving the position of the landmark a polygon is provided. The polygon is encoded according to the GML specification as a *gml:PolygonType*.

The spatial relation of such a landmark to a decision point is always *in*. Thus, it is implicitly encoded in the complex type *AreaLM1Type*.

StructureLMType

The spatial configuration of an intersection can be salient enough to function as a point-like landmark. The complex type *StructureLMType* is introduced providing the information necessary for the identification of an intersection functioning as a landmark.

Intersection The information about the intersection itself is already contained in the element *JunctionCategory* of the element of the type *XManeuverType* representing an instruction at a decision point.

In order to keep the data structure easily usable and to point out that this information must be used here again, the same element of the type *JunctionCategory* is contained again in the element *Intersection*, even though this results in providing redundant data.

StreetNameLMType

Streets identified by their name can function as point-like landmarks. They are represented by the complex type *StreetNameLMType*, which provides the required information the wayfinder needs in order to identify the particular street at the decision point. Since the name of the landmark is already contained in the supertype *AbstractLandmarkType*, this type adds only the following attribute:

PositionAtIntersection This attribute of the type *gml:AngleType* gives the angle of the branch of the intersection functioning as a landmark with respect to the branch on which the wayfinder arrives at the intersection.

GSOLMType

All other landmarks with a point-like function which do not belong to the categories described above are represented by the complex type *GSOLMType*. It describes a general salient object in the environment and provides attributes and elements containing general information about the appearance of such an object.

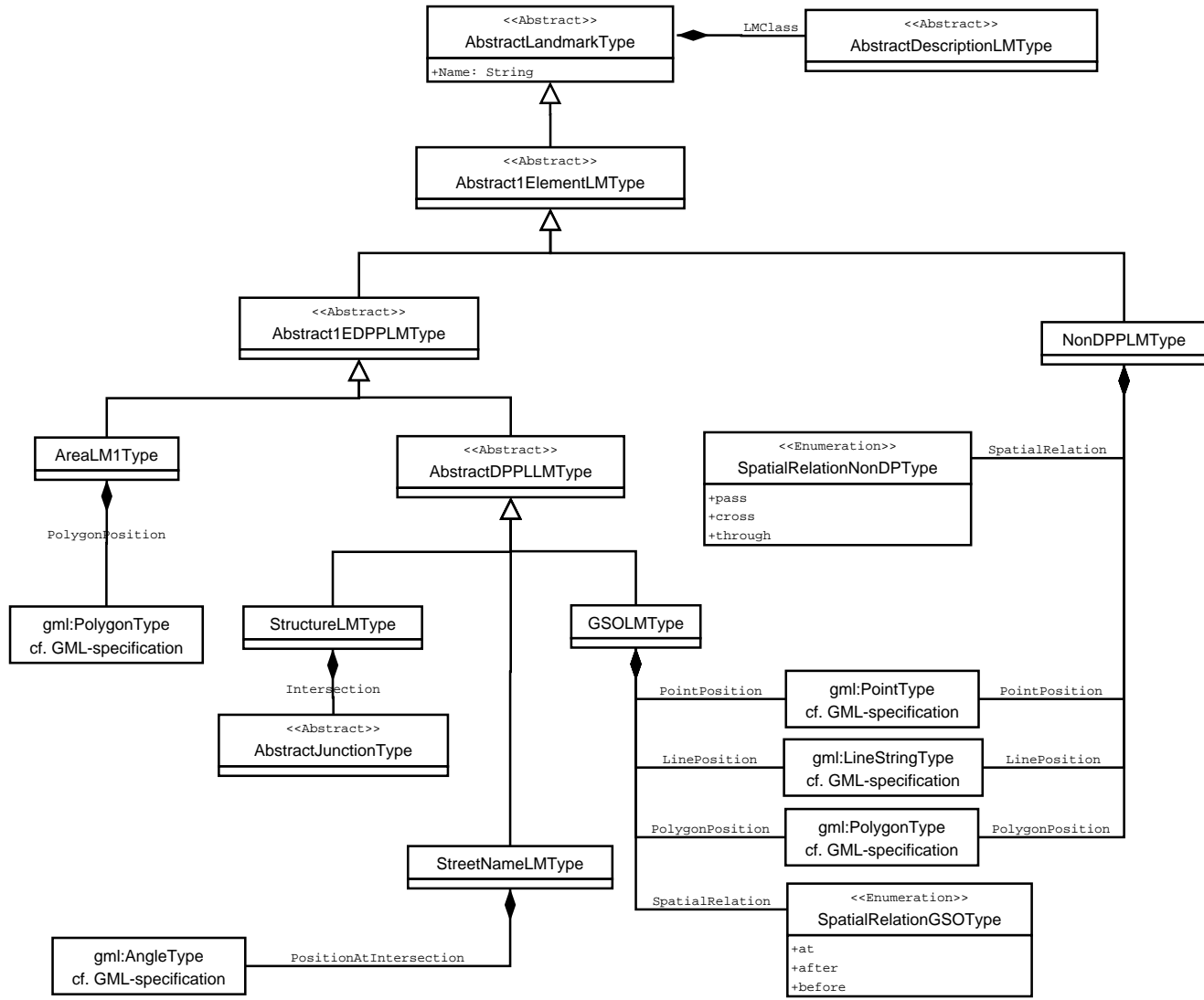


Figure 3.8: Data structure representing 1-Element landmarks.

PointPosition, LinePosition, AreaPosition A GSO can have an area-like, linear or point-like geometry. Accordingly, there is a choice between different ways to encode the geographic location. The position is encoded either as *gml:PointType*, as *gml:LineStringType*, or as *gml:PolygonType*.

SpatialRelation This attribute gives the spatial relation of the described object to the route segment. It is of the simple type *SpatialRelationGSOType*, which lists all possible spatial relations identified by a verbal label (cf. figure 3.9).

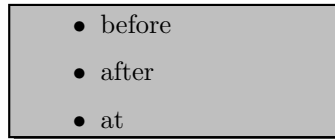
- 
- before
 - after
 - at

Figure 3.9: Possible spatial relations between a point-landmark and a decision point.

NonDPPLMType

For point-landmarks located at route segments the type *NonDPPLMType* derived from *Abstract1ElementLMType* is introduced. It provides the following attribute and element:

PointPosition, LinePosition, AreaPosition As in the other types representing landmarks which can have different geometries, there is a choice between different ways to encode the geographic location. The position is encoded either as *gml:PointType*, as *gml:LineStringType*, or as *gml:PolygonType*.

SpatialRelation The possible spatial relations are provided by this attribute encoded as *SpatialRelationNonDPTType*. Depending on the geometry of the landmark, the relation can be *pass*, *cross* or *through*.

3.5.5 Description of Landmarks

For using landmarks properly in route directions, they need to be described in order to be identifiable. A wayfinder has to be informed what kind of salient object is used as the landmark and its appearance has to be described as far as it is necessary to identify the object unambiguously.

All this necessary information has to be provided in the landmark elements. Since this part of the OpenLS navigation service tries to deliver the requested data independent from the finally generated instruction, the information for describing a landmark has to be encoded independent from constraints imposed, for example, by concrete verbal description.

Presently, this problem has not yet been sufficiently solved. Here, only a short sketch of a possible solution is outlined to define a non-abstract type for generating example documents based on the proposed extension.

PictureData In order to enable the wayfinder to identify the used landmark a picture showing the landmark can be helpful. Thus, the optional element *PictureData* is provided. Thereby the picture is encoded as a string according to *base64*, a binary to text encoding scheme defined in the *IETF* (Internet Engineering Task Force) standard for *Multipurpose Internet Mail Extensions* [6]. This scheme for encoding binary data in a string is also used in other parts of the OpenLS specification.

PictureURL Apart from the picture itself, also an Internet address can be provided linking to the Internet location of a picture showing the landmark. The *URL* (Uniform Resource Locator) is encoded as a simple string. This element can be used as an alternative to *PictureData* or additionally for making more pictures available.

InfoURL In order to allow the service to provide more information about a landmark, an *URL* can be provided in the attribute *InfoURL*. The Internet address is encoded as a simple string and links to an Internet page containing additional information about the landmark.

3.6 Chunking Route Directions

Spatial chunking allows reducing the number of route directions, and by building up a hierarchy subsumes instructions to focus on the essential information. Since chunking is not implemented in the original version of the OpenLS Navigation Service, the proposed extension adds the necessary types and adapts the data structure accordingly.

To enable the usage of chunking in OpenLS according to the work of presented in [12] and [3], several changes in the design of the data structure described in *XLS* have to be made. The possibility of subsuming a sequence of directions in one single instruction has to be introduced to allow for spatial chunking. To build up a hierarchy it is also necessary to offer attributes and elements which relate the route directions to each other.

A travel maneuver that summarises more than one decision point has to be recognisable for the users as a higher order route segment. This way they know that there is more detailed information accessible if the summarised direction is not sufficient for them. Such an instruction has to contain all the information about each subsumed decision point, so it is not necessary to contact the server again for requesting more specific instructions.

For the internal data structure, it is helpful if higher order route segments can be used in the same way as any other elementary instruction. This allows the combination of chunks of different levels in one segment. For example, a list of instructions may consist of chunks and elementary directions on the same level of the built up hierarchy. Also, an elementary route direction can be chunked together with an instruction that already subsumes several other elementary route directions.

Besides segmenting directions on a higher level to build up a hierarchy also chunking of elementary route direction elements has to be implemented. For

each possible chunking type (e.g., based on the different types of landmarks) the necessary attributes and elements which allow for indicating the end of a chunk and to describe the required actions have to be introduced.

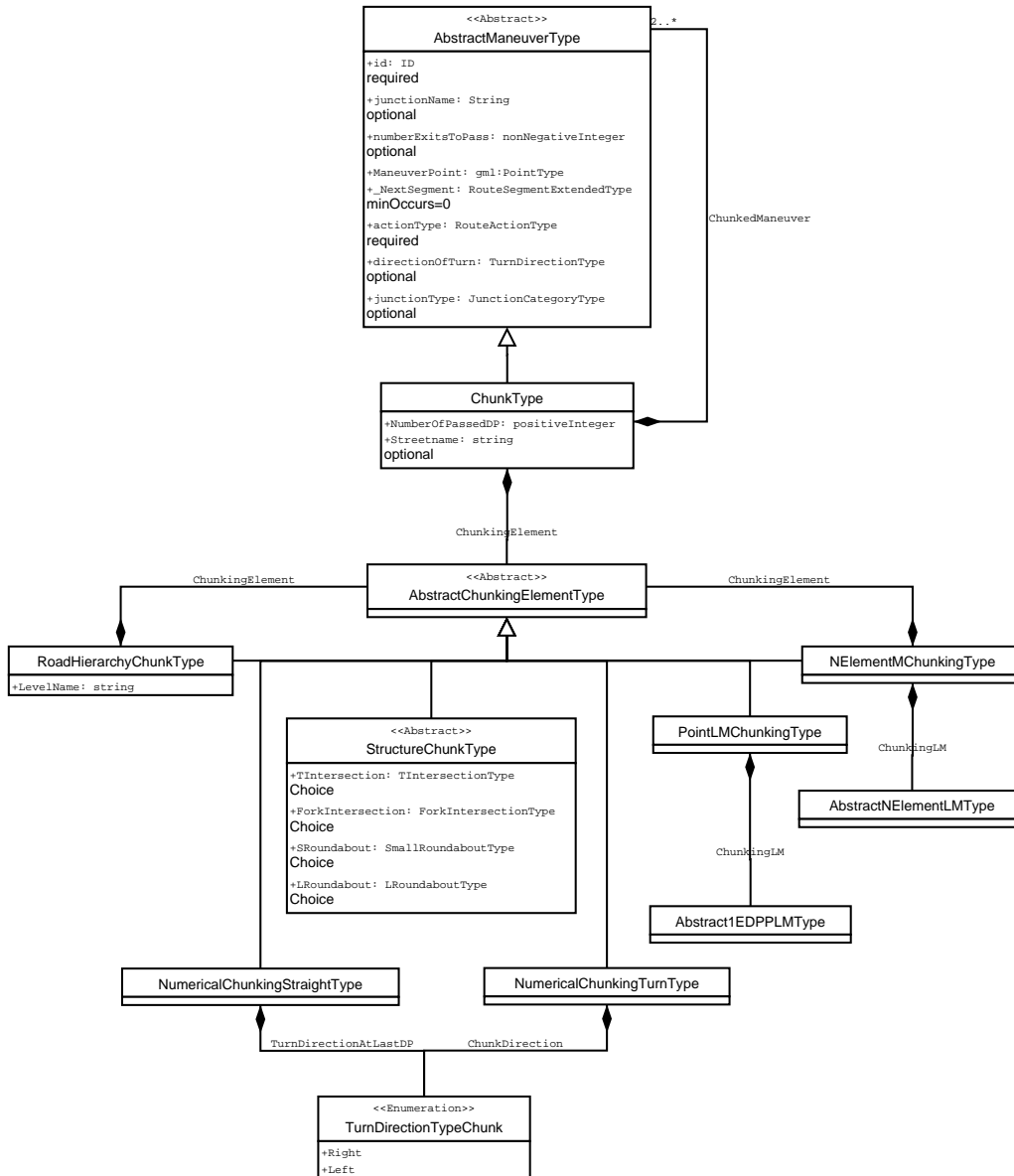


Figure 3.10: Data Structure supporting Chunking.

3.6.1 ChunkType

To enable chunking, the complex type *ChunkType* is introduced which is derived from *AbstractManeuverType*. An element of this type provides all information for describing a summarising route segment and additionally all information about the summarised directions.

Being derived from *AbstractManeuverType* has the advantage that it can be used as a normal direction and contains the same elements and attributes as an usual direction, but also provides additional attributes and elements. The type *ChunkType* extends *AbstractManeuverType* by the number of combined directions, a list of these directions, an optional street name and the element *ChunkingElement*.

NumberOfPassedDP This attribute states for how many decision points the wayfinder has to perform the described action. It is essential for numerical chunking where this is the basic information. But it can also be used for other types of chunking as additional information for supporting the identification of the end of the segment.

ChunkedManeuver This element provides a list of the subsumed instructions as elements of the type *AbstractManeuverType*. Each element of this list represents one of the summarised directions and the corresponding information.

Since the type *ChunkType* is derived from *AbstractManeuverType*, the list can also contain an object of the type *ChunkType*. This allows building a recursive hierarchy. The structure of the hierarchy is not constrained.

Since each single subsumed instruction is still available in this list including all information necessary for its description, it is possible to generate more detailed and less chunked route directions from this data set without contacting the sever again, if this is requested by the user.

Streetname The additional information provided by this attribute is simply the name of the street which is used to subsume a sequence of decision points along the road.

Street name chunking always requires an additional element to indicate the end of the chunk. For specifying the end of such a chunk, the other element *ChunkingElement*, which is always part of a *ChunkType* element, can be used.

ChunkingElement To indicate unambiguously the last decision point of a sequence, an element of the type *AbstractChunkElement* is part of each chunk. Therefore, each object contains an element of the type *AbstractChunkElement*.

LastIntersection Even though the last instruction of a chunk provides the category of the last intersection and the required turn, a *ChunkType* contains an element of the type *AbstractJunctionCategoryType*. Since the same element is part of the chunking element *StructureChunkType* this element is optional to avoid further redundancy.

These elements define the type of chunking, which is used to create this subsuming instruction. For each single type of chunking, one type derived

from *AbstractChunkElement* exists, which contains the necessary information and indicates the type of chunking. The structure of an element of the type *AbstractChunkElement* is explained in the next section.

3.6.2 AbstractChunkElement

For each possibility to indicate the last decision point of a sequence of chunked route directions, an element of a different type is used.

Chunking based on the name of the street is integrated in the other chunking methods, since its usage always needs a special element for indicating the last decision point, similar to the other types of chunking.

StructureChunkType

If the structure of an intersection is used to indicate the end of a chunk, the chunking element *StructureChunkType* is used. It contains either an element of the type *TIntersectionType*, *ForkIntersectionType*, *LargeRoundaboutType* or *SmallRoundAboutType*. The other categories of intersections are not salient enough.

RoadHierarchyChunkType

For the usage of road hierarchy chunking, the level of the current road segment must be indicated. Since it differs depending, for example, on the country (the administrative road hierarchy can vary in different countries) and the actual type of hierarchy (e.g., an official hierarchy or just a street where you have always the right of way), the used hierarchy cannot be predetermined. Thus, it is practically impossible to regard all possibilities for the hierarchy of the current street network. Therefore, an element of the type *RoadHierarchy* contains a String *LevelName*, which is supposed to provide the official name of the current road hierarchy level. An element of the type *RoadHierarchy* contains again an *AbstractChunkElement* (*ChunkingElement*) to indicate the end of the segment.

NumericalChunkingTurnType

Elements of this type are used to indicate the usage of numerical chunking. The necessary information is mainly already provided by the corresponding *AbstractChunkElement*, because it can also be used for other types of chunking as supporting information. Therefore, this type contains only additional information of the turn (*TurnDirection*) since the category of directional change has to be identical at each of the subsumed decision points.

NumericalChunkingStraightType

Apart from the missing *TurnDirection* this type is identical with *NumericalChunkingTurnType*. It indicates the chunking of several decision points without directional change.

PointLMChunkingType

For using a point-landmark as chunking element and, thus, landmark-based chunking, a chunking element of this type is used. The only element it contains is the landmark object of the type *Abstract1EDPLMType*.

NElementLMChunkingType

For the chunking method based on *n-Elements* landmarks, a special type is derived from *AbstractChunkElement*. Like the *PointLMChunkingType* it contains a landmark, but of the type *AbstractNElementLMType*. If the landmark is not sufficient to identify the end of the chunk, additionally another chunking element can be provided. Therefore, chunking with a *n-Elements* landmark can, for example, be combined with *PointLMChunkingType* or numerical chunking.

4 Examples of Cognitive OpenLS

In this chapter we provide some handcrafted examples on how to specify route directions using Cognitive OpenLS. All examples are based on real situations, but the used data is not based on an existing data set. For creating the examples, a route was calculated using the navigation service *www.uk.map24.com* and turned manually into an encoding in Cognitive OpenLS.

Due to this method of generating the required data, there is no geospatial data available and thus, no exact angular information; all elements storing geographic coordinates are set on default values. Only the first example covers the description of a complete route. All others focus on a certain aspect and, thus, contain only selected elements of the XLS-code.

For each example we provide possible verbal externalizations that, again, are only used to illustrate the data-structure. They are not generated by any automatic system, nor are they meant to be the only possible solution.

All pictures used in this chapter are based on maps generated with *www.uk.-map24.com* in March 2006.

4.1 Example 1: A complete route

The first example is the only one that covers a complete route from start point to end point. The route starts at *Ronzelenstrasse 18, 28359 Bremen* and ends at the *Ortsamt Horn-Lehe (Berckstrasse 10, 28359 Bremen)*. It comprises instructions for the beginning and the end of the route, a chunk using numerical chunking, four point-landmarks and the basic features of a route (e.g., a simple maneuver).

Each XLS file starts with a header defining the name space and the other included XML-schemas. The extended version of XLS needs to include the schema-file *ADT_NavXtension.xsd*.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xls:XManeuverList xmlns:xls="http://www.opengis.net/xls" xmlns:sch="
  http://www.ascc.net/xml/schematron" xmlns:gml="http://www.opengis.net
  /gml" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://
  www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.
  opengis.net/xls
3 D:\xml\ols1_1\05-016\ADT_NavXtension.xsd">
```

The description of the route starts with a chunk subsuming the two decision points of the route. The start and the end maneuver can be found at the end of the maneuver list. The *xls:ManeuverPoint* gives the location of the last decision point of the chunk.

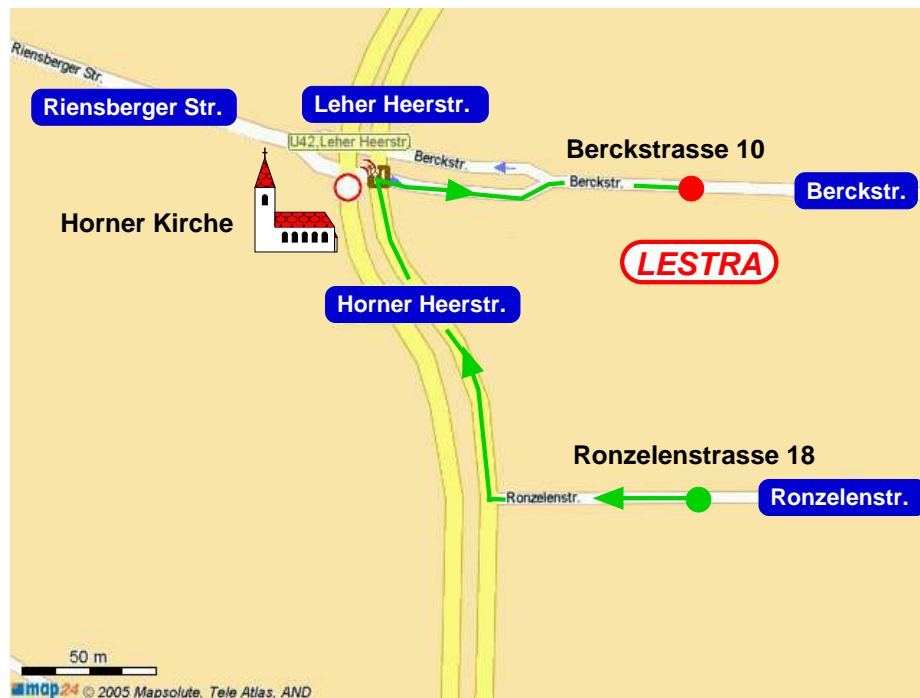


Figure 4.1: Map of the route in example 1 from Ronzelenstrasse 18, 28359 Bremen to Berckstrasse 10, 28359 Bremen. Based on map from www.uk.map24.com from March 2006.

```

4 <xls:ChunkManeuver id="chunk1" actionType=" Advisory" NumberOfPassedDP="
5   2">
6   <xls:ManeuverPoint>
7     <gml:pos>122.452372436 37.7725892713</gml:pos>
8   </xls:ManeuverPoint>

```

The first of the chunked maneuvers describes the action a wayfinder has to perform at a T–intersection. This intersection is also used as a structural landmark; therefore the information about the intersection is encoded twice.

```

9   <xls:ChunkedManeuver xsi:type="xls:XManeuverType" actionType="Turn"
10     id="dp1" directionOfTurn=" Right" junctionType=" Intersection" >
11     <xls:ManeuverPoint>
12       <gml:pos>122.452372436 37.7725892713</gml:pos>
13     </xls:ManeuverPoint>
14     <xls:JunctionCategory xsi:type="xls:TIntersectionType"
15       TurnDirection=" right">
16       <xls:RouteBranch Streetname=" Horner_Heerstrasse">
17         <xls:Angle uom=" degree">90</xls:Angle>
18       </xls:RouteBranch>
19       <xls:NoRouteBranch Streetname=" Horner_Heerstrasse">
20         <xls:Angle uom=" degree">270</xls:Angle>
21       </xls:NoRouteBranch>
22     </xls:JunctionCategory>
23     <xls:Landmark xsi:type="xls:StructureLMType">
24       <xls:Description xsi:type="xls:LMDescriptionExampleType"></
25         xls:Description>
26       <xls:Intersection xsi:type="xls:TIntersectionType" TurnDirection="
27         right">
28         <xls:RouteBranch Streetname=" Horner_Heerstrasse">
29           <xls:Angle uom=" degree">90</xls:Angle>
30         </xls:RouteBranch>
31         <xls:NoRouteBranch Streetname=" Horner_Heerstrasse">
32           <xls:Angle uom=" degree">270</xls:Angle>
33         </xls:NoRouteBranch>
34       </xls:Intersection>
35     </xls:Landmark>
36     <xls:PreviousSegment Streetname=" Ronzelenstrasse">
37       <xls:Distance value=" 10"></xls:Distance>
38       <xls:TravelTime>P1Y2M3DT10H30M12.3S</xls:TravelTime>
39       <xls:BoundingBox>
40         <gml:pos>122.452372436 37.7725892713</gml:pos>
41         <gml:pos>122.452372436 37.7725892713</gml:pos>
42       </xls:BoundingBox>
43     </xls:PreviousSegment>
44   </xls:ChunkedManeuver>
45
46   <xls:ChunkedManeuver xsi:type="xls:XManeuverType" actionType="Turn"
47     id="dp2" directionOfTurn=" Right" junctionType=" Intersection" >
48     <xls:ManeuverPoint>
49       <gml:pos>122.452372436 37.7725892713</gml:pos>
50     </xls:ManeuverPoint>
51     <xls:JunctionCategory xsi:type="xls:StandardIntersectionType"
52       TurnDirection=" right">
53       <xls:RouteBranch Streetname=" Berckstrasse">
54         <xls:Angle uom=" degree">90</xls:Angle>
55       </xls:RouteBranch>
56       <xls:NoRouteBranch Streetname=" Riensberger_Strasse">
57         <xls:Angle uom=" degree">270</xls:Angle>
58       </xls:NoRouteBranch>
59       <xls:NoRouteBranch Streetname=" Leher_Heerstrasse">
60         <xls:Angle uom=" ">0</xls:Angle>

```

```

55     </xls:NoRouteBranch>
56 </xls:JunctionCategory>

```

The second of the chunked decision points is identified by church, which functions as a GSO/point-landmark. The landmark is also integrated in the description of the required turn, since a wayfinder has to perform a directional change after she passed the church.

```

61     <xls:Landmark xsi:type="xls:GSOLMType" Name="Horner_Kirche"
62       SpatialRelation="after" >
63     <xls:Description xsi:type="xls:LMDescriptionExampleType"></
64       xls:Description>
65     <xls:PointPosition>
66     <gml:pos>122.452372436 37.7725892713</gml:pos>
67     </xls:PointPosition>
68 </xls:Landmark>
69 <xls:PreviousSegment Streetname="Horner_Heerstrasse">
70   <xls:Distance value="10"></xls:Distance>
71   <xls:TravelTime>P1Y2M3DT10H30M12.3S</xls:TravelTime>
72   <xls:BoundingBox>
73     <gml:pos>122.452372436 37.7725892713</gml:pos>
74     <gml:pos>122.452372436 37.7725892713</gml:pos>
75   </xls:BoundingBox>
76   </xls:PreviousSegment>
77 </xls:ChunkedManeuver>

```

The chunking element for this chunk has to provide the type of chunking (this is implicitly done by its type) and the directional change at the chunked turns. The number of subsumed elements is given at the beginning as an attribute of the chunk itself. The turn at the last decision point of the chunk is described by the element *xls>LastIntersection*. The element *xls:ChunkedSegments* contains the estimated travel time and distance for all chunked segments together.

```

76     <xls>LastIntersection xsi:type="xls:StandardIntersectionType"
77       TurnDirection="left">
78     <xls:RouteBranch Streetname="Wilhelm-Kaisen-Bruecke">
79       <xls:Angle uom="degree">270</xls:Angle>
80     </xls:RouteBranch>
81     <xls:NoRouteBranch Streetname="Balgebrueckstrasse">
82       <xls:Angle uom="degree">90</xls:Angle>
83     </xls:NoRouteBranch>
84     <xls:NoRouteBranch Streetname="Martinistrasse">
85       <xls:Angle uom="">0</xls:Angle>
86     </xls:NoRouteBranch>
87   </xls>LastIntersection>
88
89   <xls:ChunkingElement
90     xsi:type="xls:NumericalChunkingTurnType"
91     TurnDirection="right"></xls:ChunkingElement>
92
93   <xls:ChunkedSegments>
94     <xls:Distance value="10"></xls:Distance>
95     <xls:TravelTime>P1Y2M3DT10H30M12.3S</xls:TravelTime>
96     <xls:BoundingBox>
97       <gml:pos>22.452372436 37.772589271</gml:pos>
98       <gml:pos>22.452372436 37.772589271</gml:pos>
99     </xls:BoundingBox>

```

```

100     </xls:ChunkedSegments>
101
102     </xls:ChunkManeuver>

```

The start maneuver orients the wayfinder at the beginning of the route. In this case the absolute direction (*West*), the direction according to the address and the first route segment (*right*), and a landmark are provided. Since there is no better object available that could function as landmark just the next road the wayfinder is heading towards is used.

```

76     <xls:StartingManeuver id="Start" Orientation="W" RoadDirection="right">
77
78         <xls:Address countryCode="DE">
79             <xls:freeFormAddress>
80                 Ronzelenstrasse 10, 28359 Bremen (Horn-Lehe)
81             </xls:freeFormAddress>
82         </xls:Address>
83
84         <xls:Position>
85             <gml:pos>22.452372436 37.7725892713</gml:pos>
86         </xls:Position>
87
88         <xls:Landmark Orientation="W" SpatialRelation="towards" Name="Horner_
89             Heerstrasse">
90             <xls:Description xsi:type="xls:LMDescriptionExampleType"></
91                 xls:Description>
92             <xls:PointPosition>
93                 <gml:pos>22.452372436 37.7725892713</gml:pos>
94             </xls:PointPosition>
95         </xls:Landmark>
96     </xls:StartingManeuver>

```

For describing the destination of the route a point-landmark is given in the end maneuver. To inform the wayfinder about its location and relation to its destination, the absolute orientation and the side of the road the object is located on is given, both according to the wayfinder's current position. Furthermore, the relationship between destination and landmark is provided (*opposite*).

```

76     <xls:EndManeuver id="end" SideOfRoad="Left">
77
78         <xls:Address countryCode="DE">
79             <xls:freeFormAddress>Berckstrasse 10, 28359 Bremen</
80                 xls:freeFormAddress>
81         </xls:Address>
82
83         <xls:Position>
84             <gml:pos>22.452372436 37.7725892713</gml:pos>
85         </xls:Position>
86
87         <xls:Landmark Orientation="N" Name="Lestra" SideOfRoad="Right"
88             SpatialRelation="opposite">
89             <xls:Description xsi:type="xls:LMDescriptionExampleType"></
90                 xls:Description>
91             <xls:PointPosition>
92                 <gml:pos>22.452372436 37.7725892713</gml:pos>
93             </xls:PointPosition>
94         </xls:Landmark>

```

```
92
93 <xls:PreviousSegment Streetname=" Berckstrasse">
94 <xls:Distance value=" 12"></xls:Distance>
95 <xls:TravelTime>P1Y2M3DT10H30M12.3S</xls:TravelTime>
96 <xls:BoundingBox>
97 <gml:pos>22.452372436 37.772589271</gml:pos>
98 <gml:pos>22.452372436 37.772589271</gml:pos>
99 </xls:BoundingBox>
100 </xls:PreviousSegment>
101
102 </xls:EndManeuver>
103
104
105 </xls:XManeuverList>
```

The example illustrates the basic usage of the data model and some more advanced features. This example demonstrates as many of the features as possible; some of them would not be used in the actual generated route directions. For example, the church describing the turn at the second decision point might not be mentioned since this turn is already described by the numerical chunk. Possible route directions based on this specification are:

1. *Starting at Ronzelenstarsse 18, turn right into Ronzelenstrasse towards Horner Heerstrasse.*
2. *Turn twice right.*
3. *Following Berckstrasse, you arrive at Berckstrasse 10 located on your left-hand side opposite Lestra.)*

4.2 Example 2: Spatial chunking

The second example demonstrates chunking using a linear landmark and chunking using road hierarchy. The chunk employing the landmark is subsumed in the other chunk together with a third chunk.

After turning on the road *Osterdeich*, which is also called *B75*, the route leads along the river *Weser* until the bridge *Wilhelm-Kaisen-Bruecke*, which functions as a point-like landmark. The road is still part of the *B75*, even though the street name had changed while following the river to *Tiefer*. After the bridge the street is called *Friedrich-Ebert-Strasse*, but the wayfinder is still on the *B75*. She leaves it when she turns into *Kornstrasse* at the end of the chunk. This second part of the chunk combining the roads belonging to the *B75* can also be subsumed using numerical chunking.

The example starts with the first chunk, which subsumes the two other chunks. Parts of the specification that do not illustrate features not already contained in the first example are replaced by [...] to shorten the listings. The first chunked maneuver is already the other chunk, which subsumes also 9 maneuvers.

```
1 <xls:ChunkManeuver id="C1" actionType=" Advisory" NumberOfPassedDP=" 2">
2
```



Figure 4.2: Map of the route in example 2. It is based on a map provided by www.de.map24.com in March 2006.

```
3   [...]
4
5   <xls:ChunkedManeuver xsi:type="xls:ChunkType" id="C2" actionType="
6     Advisory" NumberOfPassedDP="9">
7     [...]
8
9     <xls:ChunkedManeuver xsi:type="xls:XManeuverType" actionType="Turn"
10      id="M2" directionOfTurn="Straight" junctionType="Intersection" >
11       [...]
12     </xls:ChunkedManeuver>
13     [...]
14
15     <xls:ChunkedManeuver xsi:type="xls:XManeuverType" actionType="Turn"
16      id="M10" directionOfTurn="Left" junctionType="Intersection">
17       [...]
18     </xls:ChunkedManeuver>
```

The river *Weser* is used for landmark chunking. Since this landmark is not sufficient to identify the end of the chunk, a point-landmark (a bridge) is used for this purpose. The second of the two chunks subsuming eight intersections uses again numerical chunking. Since this is already demonstrated in the first example, it is not listed in detail.

```
17   <xls:ChunkingElement xsi:type="xls:NElementLMChunkType" >
18     <xls:ChunkingLM xsi:type="xls:NotIdentifyingLLType" Name="Weser">
19       <xls:Description xsi:type="xls:LMDescriptionExampleType"></
20         xls:Description>
21       <xls:LinePosition>
22         <gml:pos>22.452372436 37.7725892713</gml:pos>
23         <gml:pos>22.452372436 37.7725892713</gml:pos>
24       </xls:LinePosition>
25       <xls:Landmark xsi:type="xls:GSOLMType" SpatialRelation="at" Name=
26         "Wilhelm-Kaisen-Bruecke">
27         <xls:Description xsi:type="xls:LMDescriptionExampleType"></
28           xls:Description>
29         <xls:PointPosition>
30           <gml:pos>122.452372436 37.7725892713</gml:pos>
31         </xls:PointPosition>
32       </xls:Landmark>
33     </xls:ChunkingLM>
34   </xls:ChunkingElement>
35
36   <xls>LastIntersection xsi:type="xls:StandardIntersectionType"
37     TurnDirection="left">
38     [...]
39   </xls>LastIntersection>
40
41   <xls:ChunkedSegments>
42     [...]
43   </xls:ChunkedSegments>
44 </xls:ChunkedManeuver>
45
46   <xls:ChunkedManeuver xsi:type="xls:ChunkType" id="C3" actionType="
47     Advisory" NumberOfPassedDP="8">
48     [...]
49   </xls:ChunkedManeuver>
```


The chunking element of the type *xls:RoadHierarchyChunkType* requires an additional chunking element which identifies the end of the chunk. In this case, simply the name of the next street has been chosen. However, if appropriate, numerical chunking could also provide the same information.

```

52 <xls:ChunkingElement xsi:type="xls:RoadHierarchyChunkType" LevelName="
    B75">
53   <xls:ChunkingElement xsi:type="xls:PointLMChunkType">
54     <xls:ChunkingLM xsi:type="xls:StreetnameLMType" Name="Kornstrasse">
55       <xls:Description xsi:type="xls:LMDescriptionExampleType"></
          xls:Description>
56       <xls:PositionAtIntersection uom="deg">90</
          xls:PositionAtIntersection>
57     </xls:ChunkingLM>
58   </xls:ChunkingElement>
59 </xls:ChunkingElement>
60
61 <xls:ChunkedSegments>
62   [...]
63 </xls:ChunkedSegments>
64
65 </xls:ChunkManeuver>

```

Route directions comprising the segments and decision points along the *B75* would contain on the highest level only one instruction describing the chunk along the *B75*. This instruction generated on the basis of this XLS-code could, for example, be:

- *Follow B75, till you reach Kornstrasse and turn left into Kornstrasse.*
 - *Follow the river and turn left at Wilhelm-Kaisen-Bruecke.*
 - *Turn left into Kornstrasse at the eighth intersection.*

On the intermediate level the route consists of two chunks which are represented by two instructions, even though an instruction requiring to count eight intersections is of questionable adequacy.

4.3 Example 3: Competing branches

The last example contains a single maneuver describing the action at a complex intersection with competing branches. The out-going branch of the route competes with another road. Therefore, an ordering concept has to be introduced. In the element *xls:JunctionCategory*, an ordering concept is used that describes the turn unambiguously. A wayfinder is told to take the second exit (passing one exit) on her right. Additionally the exact configuration of the route is provided to give the wayfinder an exact picture of the spatial configuration.

```

1 <xls:XManeuver id="x" actionType="Turn" directionOfTurn="Right"
    junctionType="Intersection" >
2
3   <xls:ManeuverPoint>
4     <gml:pos>122.452372436 37.7725892713</gml:pos>
5   </xls:ManeuverPoint>

```

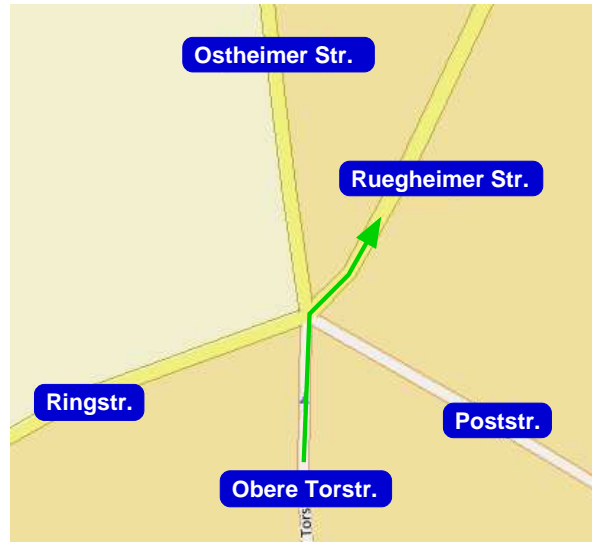


Figure 4.3: Map of the route in example 3. It is based on a map provided by www.de.map24.com in March 2006.

```

6
7 <xls:JunctionCategory xsi:type="xls:CompetingBranchesType"
8   numberExitsToPass="1" TurnDirection="right">
9 <xls:RouteBranch Streetname="Ruegheimer_Strasse">
10 <xls:Angle uom="degree">130</xls:Angle>
11 </xls:RouteBranch>
12 <xls:NoRouteBranch Streetname="Poststrasse">
13 <xls:Angle uom="degree">45</xls:Angle>
14 </xls:NoRouteBranch>
15 <xls:NoRouteBranch Streetname="Ostheimer_Strasse">
16 <xls:Angle uom="degree">190</xls:Angle>
17 </xls:NoRouteBranch>
18 <xls:NoRouteBranch Streetname="Ringstrasse">
19 <xls:Angle uom="degree">280</xls:Angle>
20 </xls:NoRouteBranch>
21 </xls:JunctionCategory>
22
23
24 <xls:PreviousSegment Streetname="Obere_Torstrasse">
25 <xls:Distance value="10"></xls:Distance>
26 <xls:TravelTime>P1Y2M3DT10H30M12.3S</xls:TravelTime>
27 <xls:BoundingBox>
28 <gml:pos>122.452372436 37.7725892713</gml:pos>
29 <gml:pos>122.452372436 37.7725892713</gml:pos>
30 </xls:BoundingBox>
31 </xls:PreviousSegment>
32
33 </xls:XManeuver>

```

An instruction based on this data could, for example, be:

“Take the second branch on your right.”

5 Schema

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <schema xmlns:gml="http://www.opengis.net/gml" xmlns:xls="http://www.
   opengis.net/xls" xmlns="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.opengis.net/xls" elementFormDefault="
   qualified">
3 <import namespace="http://www.opengis.net/gml" schemaLocation="gml4xls.
   xsd"/>
4 <include schemaLocation="ADT.xsd"/>
5 <include schemaLocation="ADT_Navigation.xsd"/>
6 <!-- Basic element -->
7 <element name="XManeuverList" type="xls:XRouteManeuverListType"/>
8 <!-- General Types -->
9 <complexType name="XRouteManeuverListType">
10 <annotation>
11 <documentation>Extended version of the
12 AbstractManeuverListType. Defines a list of
13 travel maneuvers (chapter 5.3)
14 </documentation>
15 </annotation>
16 <complexContent>
17 <extension base="xls:RouteManeuverListType">
18 <sequence>
19 <element name="StartingManeuver" type="
   xls:XStartingManeuverType">
20 <annotation>
21 <documentation>
22 Element describing the starting maneuver
23 (chapter 5.3.1)
24 </documentation>
25 </annotation>
26 </element>
27 <element name="EndManeuver" type="xls:XEndManeuverType">
28 <annotation>
29 <documentation>
30 Element describing the starting maneuver
31 (chapter 5.3.1)
32 </documentation>
33 </annotation>
34 </element>
35 </sequence>
36 </extension>
37 </complexContent>
38 </complexType>
39 <complexType name="XStartingManeuverType">
40 <annotation>
41 <documentation>
42 All information about the first instruction of a route
43 (chapter 5.3.1)
44 </documentation>
45 </annotation>
46 <sequence>
47 <element name="Address" type="xls:AddressType">
48 <annotation>
49 <documentation>
50 Element providing the street address of
51 the starting point. AddressType is explained
52 in the OpenLS specification.
```

```
53         </documentation>
54     </annotation>
55 </element>
56 <element name="Position" type="gml:PointType">
57     <annotation>
58         <documentation>
59             Geographic location of the starting point encoded
60             as gml:PointType
61         </documentation>
62     </annotation>
63 </element>
64 <element name="Landmark" type="xls:StartingPointLMType">
65     <annotation>
66         <documentation>
67             Landmark used for orientating the traveller at
68             the starting point
69         </documentation>
70     </annotation>
71 </element>
72 </sequence>
73 <attribute name="id" type="ID" use="required" />
74 <attribute name="Orientation" type="gml:CompassPointEnumeration" use="
75     "required">
76     <annotation>
77         <documentation>
78             Specifying the travel direction as an orientation
79             encoded as gml:CompassPointEnumeration.
80         </documentation>
81     </annotation>
82 </attribute>
83 <attribute name="RoadDirection" type="xls:RoadDirectionType" use="
84     optional">
85     <annotation>
86         <documentation>
87             The direction to travel along the first route segment
88             from the starting point.
89         </documentation>
90     </annotation>
91 </attribute>
92 </complexType>
93 <complexType name="XEndManeuverType">
94     <annotation>
95         <documentation>All information about the last instruction of a
96             route
97             (chapter 5.3.2)
98         </documentation>
99     </annotation>
100 <sequence>
101     <element name="Address" type="xls:AddressType">
102     <annotation>
103         <documentation>
104             Element providing the street address of
105             the end point. AddressType is explained in the OpenLS
106             specification.
107         </documentation>
108     </annotation>
109 </element>
110 <element name="Position" type="gml:PointType">
111     <annotation>
112         <documentation>
113             Geographic location of the end point encoded as
114             gml:PointType
115         </documentation>
116     </annotation>
117 </element>
118 <element name="Landmark" type="xls:EndPointLMType">
```

```

116     <annotation>
117         <documentation>
118             Landmark used for orientating the traveller at
119             the starting point
120         </documentation>
121     </annotation>
122 </element>
123 <element name="PreviousSegment" type="xls:XRouteSegment">
124     <annotation>
125         <documentation>
126             Segment leading to the end point
127         </documentation>
128     </annotation>
129 </element>
130 </sequence>
131 <attribute name="id" type="ID" use="required"/>
132 <attribute name="SideOfRoad" type="xls:SideOfRoadType" use="optional"
133     >
134     <annotation>
135         <documentation>
136             Specifying on which side of the road the destination is
137             located. Using an OpenLS simple type
138         </documentation>
139     </annotation>
140 </attribute>
141 </complexType>
142 <complexType name="XManeuverType">
143     <annotation>
144         <documentation>
145             Extended version of the
146             AbstractManeuverListType. Defines a travel maneuver (chapter
147             5.)
148         </documentation>
149     </annotation>
150 <complexContent>
151     <extension base="xls:AbstractManeuverType">
152         <sequence>
153             <element name="JunctionCategory" type="xls:AbstractJunctionType
154                 ">
155                 <annotation>
156                     <documentation>
157                         This element contains the category of the intersection
158                     </documentation>
159                 </annotation>
160             </element>
161             <element name="Landmark" type="xls:Abstract1EDPLMType"
162                 minOccurs="0" maxOccurs="unbounded">
163                 <annotation>
164                     <documentation>
165                         List of 1-element-DP landmarks at this decision point.
166                     </documentation>
167                 </annotation>
168             </element>
169             <element name="PreviousSegment" type="xls:XRouteSegment">
170                 <annotation>
171                     <documentation>
172                         This element contains the previous route segment
173                     </documentation>
174                 </annotation>
175             </element>
176         </sequence>
177     </extension>
178 </complexContent>
179 </complexType>
180 <element name="XManeuver" type="xls:XManeuverType" substitutionGroup="
181     xls:_Maneuver">

```

```
178     <annotation>
179       <documentation>
180         A travel maneuver.
181       </documentation>
182     </annotation>
183   </element>
184   <complexType name="RoadNameChangeType">
185     <annotation>
186       <documentation>
187         Information about a road name change
188       </documentation>
189     </annotation>
190     <sequence>
191       <element name="PointPosition" type="gml:PointType">
192         <annotation>
193           <documentation>
194             Position encoded as a gml:PointType
195           </documentation>
196         </annotation>
197       </element>
198     </sequence>
199     <attribute name="NewName" type="string" use="required">
200       <annotation>
201         <documentation>
202           The new street name
203         </documentation>
204       </annotation>
205     </attribute>
206   </complexType>
207   <complexType name="XRouteSegment">
208     <annotation>
209       <documentation>
210         RouteSegmentExtendedtype extension by landmarks
211       </documentation>
212     </annotation>
213     <complexContent>
214       <extension base="xls:RouteSegmentExtendedType">
215         <sequence>
216           <element name="Landmark" type="xls:NonDPPLMType" minOccurs="0"
217             maxOccurs="unbounded">
218             <annotation>
219               <documentation>
220                 List of point-landmarks at this decision point.
221               </documentation>
222             </annotation>
223           </element>
224           <element name="RoadNameChange" type="xls:RoadNameChangeType"
225             minOccurs="0">
226             <annotation>
227               <documentation>
228                 Information about a street name change
229               </documentation>
230             </annotation>
231           </element>
232         </sequence>
233         <attribute name="Streetname" type="string" use="required">
234           <annotation>
235             <documentation>
236               Streetname of the segment
237             </documentation>
238           </annotation>
239         </attribute>
240       </extension>
241     </complexContent>
242   </complexType>
243   <complexType name="Branch">
```

```

242 <annotation>
243   <documentation>
244     Type representing a branch of an intersection
245   </documentation>
246 </annotation>
247 <sequence>
248   <element name="Angle" type="gml:AngleType">
249     <annotation>
250       <documentation>
251         Angle between branch and incoming branch.
252       </documentation>
253     </annotation>
254   </element>
255 </sequence>
256 <attribute name="Streetname" type="string" use="required">
257   <annotation>
258     <documentation>
259       Name of the street/branch
260     </documentation>
261   </annotation>
262 </attribute>
263 </complexType>
264 <!-- Junctions -->
265 <complexType name="AbstractJunctionType" abstract="true">
266   <annotation>
267     <documentation>
268       Abstract type for a description of an intersection
269     </documentation>
270   </annotation>
271   <sequence>
272     <element name="RouteBranch" type="xls:Branch">
273       <annotation>
274         <documentation>
275           Angle of the outgoing branch in respect with the incoming
276             branch
277           encoded gml:AngleType
278         </documentation>
279       </annotation>
280     </element>
281     <element name="NoRouteBranch" type="xls:Branch" minOccurs="0"
282       maxOccurs="unbounded">
283       <annotation>
284         <documentation>
285           Angle of the other branches in respect with the incoming
286             branch
287           encoded gml:AngleType. One elment for each branch.
288         </documentation>
289       </annotation>
290     </element>
291   </sequence>
292   <attribute name="Name" type="string" use="optional">
293     <annotation>
294       <documentation>
295         If the intersection has a name.
296       </documentation>
297     </annotation>
298   </attribute>
299 </complexType>
300 <complexType name="AbstractNonCompetingType" abstract="true">
301   <annotation>
302     <documentation>
303       Abstract type for a description of an intersection
304     </documentation>
305   </annotation>
306 </complexType>
307 <extension base="xls:AbstractJunctionType"/>

```

```
305     </complexContent>
306 </complexType>
307 <complexType name=" TIntersectionType">
308   <annotation>
309     <documentation>
310       Abstract type for a description of an intersection
311     </documentation>
312   </annotation>
313   <complexContent>
314     <extension base="xls:AbstractNonCompetingType">
315       <attribute name="TurnDirection" type="xls:TurnDirectionTFType"
316         use="required">
317         <annotation>
318           <documentation>
319             Possible directions
320           </documentation>
321         </annotation>
322       </attribute>
323     </extension>
324   </complexContent>
325 </complexType>
326 <complexType name=" ForkIntersectionType">
327   <annotation>
328     <documentation>
329       Abstract type for a description of an intersection
330     </documentation>
331   </annotation>
332   <complexContent>
333     <extension base="xls:AbstractNonCompetingType">
334       <attribute name="TurnDirection" type="xls:TurnDirectionTFType"
335         use="required">
336         <annotation>
337           <documentation>
338             Possible directions
339           </documentation>
340         </annotation>
341       </attribute>
342     </extension>
343   </complexContent>
344 </complexType>
345 <complexType name=" StandardIntersectionType">
346   <annotation>
347     <documentation>
348       Abstract type for a description of an intersection
349     </documentation>
350   </annotation>
351   <complexContent>
352     <extension base="xls:AbstractNonCompetingType">
353       <attribute name="TurnDirection" type="xls:TurnDirectionSIType"
354         use="required">
355         <annotation>
356           <documentation>
357             Possible directions
358           </documentation>
359         </annotation>
360       </attribute>
361     </extension>
362   </complexContent>
363 </complexType>
364 <complexType name=" CompetingBranchesType">
365   <annotation>
366     <documentation>
367       Abstract type for a description of an intersection
368     </documentation>
369   </annotation>
370   <complexContent>
```

```

368     <extension base="xls:AbstractJunctionType">
369         <attribute name="TurnDirection" type="xls:TurnDirectionCType" use
              ="required">
370             <annotation>
371                 <documentation>
372                     Possible directions
373                 </documentation>
374             </annotation>
375         </attribute>
376         <attribute name="numberExitsToPass" type="nonNegativeInteger" use
              ="required">
377             <annotation>
378                 <documentation>
379                     number of exits to pass
380                 </documentation>
381             </annotation>
382         </attribute>
383     </extension>
384 </complexContent>
385 </complexType>
386 <complexType name="LargeRoundaboutType">
387     <annotation>
388         <documentation>
389             Abstract type for a description of an intersection
390         </documentation>
391     </annotation>
392 </complexContent>
393     <extension base="xls:AbstractJunctionType">
394         <attribute name="numberExitsToPass" type="nonNegativeInteger" use
              ="required">
395             <annotation>
396                 <documentation>
397                     number of exits to pass
398                 </documentation>
399             </annotation>
400         </attribute>
401     </extension>
402 </complexContent>
403 </complexType>
404 <complexType name="SmallRoundaboutType">
405     <annotation>
406         <documentation>
407             Abstract type for a description of an intersection
408         </documentation>
409     </annotation>
410 </complexContent>
411     <extension base="xls:AbstractJunctionType">
412         <attribute name="TurnDirection" type="xls:TurnDirectionSRType"
              use="required">
413             <annotation>
414                 <documentation>
415                     Possible directions
416                 </documentation>
417             </annotation>
418         </attribute>
419     </extension>
420 </complexContent>
421 </complexType>
422 <!-- Chunking -->
423 <complexType name="ChunkType">
424     <annotation>
425         <documentation>
426             Type for chunks subsuming other route directions.
427         </documentation>
428     </annotation>
429 </complexContent>

```

```

430     <extension base="xls:AbstractManeuverType">
431         <sequence>
432             <element name="ChunkedManeuver" type="xls:AbstractManeuverType"
433                 minOccurs="2" maxOccurs="unbounded">
434                 <annotation>
435                     <documentation>
436                         List of the chunked maneuvers.
437                     </documentation>
438                 </annotation>
439             </element>
440             <element name="ChunkingElement" type="
441                 xls:AbstractChunkingElementType">
442                 <annotation>
443                     <documentation>
444                         Element specifying the end of the chunk.
445                     </documentation>
446                 </annotation>
447             </element>
448             <element name="LastIntersection" type="xls:AbstractJunctionType
449                 minOccurs="0">
450                 <annotation>
451                     <documentation>
452                         Structure and turn at last intersection
453                     </documentation>
454                 </annotation>
455             </element>
456             <element name="ChunkedSegments" type="
457                 xls:RouteSegmentExtendedType">
458                 <annotation>
459                     <documentation>
460                         Informations about the covered segments
461                     </documentation>
462                 </annotation>
463             </element>
464         </sequence>
465         <attribute name="NumberOfPassedDP" type="positiveInteger" use="
466             required">
467             <annotation>
468                 <documentation>
469                     Attribute for the number of passed decision points.
470                 </documentation>
471             </annotation>
472         </attribute>
473         <attribute name="Streetname" type="string" use="optional">
474         <annotation>
475             <documentation>
476                 Attribute for the streetname, for example
477                 ifstreetname-chunking is used
478             </documentation>
479         </annotation>
480     </attribute>
481 </extension>
482 </complexContent>
483 </complexType>
484 <element name="ChunkManeuver" type="xls:ChunkType" substitutionGroup="
485     xls:_Maneuver">
486     <annotation>
487         <documentation>
488             A chunked travel maneuver.
489         </documentation>
490     </annotation>
491 </element>
492 <complexType name="AbstractChunkingElementType" abstract="true">
493     <annotation>
494         <documentation>
495             Abstract type for a description of a chunking element

```

```

489     </documentation>
490 </annotation>
491 </complexType>
492 <complexType name="RoadHierarchyChunkType">
493   <annotation>
494     <documentation>
495       Abstract type for a description of an intersection
496     </documentation>
497   </annotation>
498   <complexContent>
499     <extension base="xls:AbstractChunkingElementType">
500       <sequence>
501         <element name="ChunkingElement" type="
502           xls:AbstractChunkingElementType">
503           <annotation>
504             <documentation>
505               Identifying the end of the chunk
506             </documentation>
507           </annotation>
508         </element>
509       </sequence>
510       <attribute name="LevelName" type="string" use="required">
511         <annotation>
512           <documentation>
513             Name of the street hierarchy level
514           </documentation>
515         </annotation>
516       </attribute>
517     </extension>
518   </complexContent>
519 </complexType>
520 <complexType name="NElementLMChunkType">
521   <annotation>
522     <documentation>
523       Chunking using a linear landmark
524     </documentation>
525   </annotation>
526   <complexContent>
527     <extension base="xls:AbstractChunkingElementType">
528       <sequence>
529         <element name="ChunkingElement" type="
530           xls:AbstractChunkingElementType" minOccurs="0">
531           <annotation>
532             <documentation>
533               Identifying the end of the chunk
534             </documentation>
535           </annotation>
536         </element>
537         <element name="ChunkingLM" type="xls:AbstractNElementLMType">
538           <annotation>
539             <documentation>
540               Identifying the chunk with a n-elements-landmark
541             </documentation>
542           </annotation>
543         </element>
544       </sequence>
545     </extension>
546   </complexContent>
547 </complexType>
548 <complexType name="PointLMChunkType">
549   <annotation>
550     <documentation>
551       Abstract type for a description of an intersection
552     </documentation>
553   </annotation>
554   <complexContent>

```

```
553     <extension base="xls:AbstractChunkingElementType">
554         <sequence>
555             <element name="ChunkingLM" type="xls:Abstract1EDPLMType">
556                 <annotation>
557                     <documentation>
558                         Identifying the end of the chunk
559                     </documentation>
560                 </annotation>
561             </element>
562         </sequence>
563     </extension>
564 </complexContent>
565 </complexType>
566 <complexType name="StructureChunkType">
567     <annotation>
568         <documentation>
569             Abstract type for a description of an intersection
570         </documentation>
571     </annotation>
572     <complexContent>
573         <extension base="xls:AbstractChunkingElementType">
574             <sequence>
575                 <choice>
576                     <element name="TIntersection" type="xls:TIntersectionType">
577                         <annotation>
578                             <documentation>
579                                 Identifying the end of the chunk
580                             </documentation>
581                         </annotation>
582                     </element>
583                     <element name="ForkIntersection" type="
584                         xls:ForkIntersectionType">
585                         <annotation>
586                             <documentation>
587                                 Identifying the end of the chunk
588                             </documentation>
589                         </annotation>
590                     </element>
591                     <element name="SRoundabout" type="xls:SmallRoundaboutType">
592                         <annotation>
593                             <documentation>
594                                 Identifying the end of the chunk
595                             </documentation>
596                         </annotation>
597                     </element>
598                     <element name="LRoundabout" type="xls:LargeRoundaboutType">
599                         <annotation>
600                             <documentation>
601                                 Identifying the end of the chunk
602                             </documentation>
603                         </annotation>
604                     </element>
605                 </choice>
606             </sequence>
607         </extension>
608     </complexContent>
609 </complexType>
610 <complexType name="NumericalChunkingTurnType">
611     <annotation>
612         <documentation>
613             Abstract type for a description of an intersection
614         </documentation>
615     </annotation>
616     <complexContent>
        <extension base="xls:AbstractChunkingElementType">
```

```

617     <attribute name="TurnDirection" type="xls:TurnDirectionChunkType"
618         use="required">
619         <annotation>
620             <documentation>
621                 Name of the street hierarchy level
622             </documentation>
623         </annotation>
624     </attribute>
625 </extension>
626 </complexContent>
627 </complexType>
628 <complexType name="NumericalChunkingStraightType">
629     <annotation>
630         <documentation>
631             Indicates to chunk straight turns
632         </documentation>
633     </annotation>
634     <complexContent>
635         <extension base="xls:AbstractChunkingElementType">
636         </extension>
637     </complexContent>
638 </complexType>
639 <!-- Landmarks -->
640 <complexType name="AbstractLMDescriptionType" abstract="true">
641     <annotation>
642         <documentation>
643             Abstract type for a description of a landmark
644         </documentation>
645     </annotation>
646 </complexType>
647 <complexType name="LMDescriptionExampleType">
648     <annotation>
649         <documentation>
650             Example type for a description of a landmark
651         </documentation>
652     </annotation>
653     <complexContent>
654         <extension base="xls:AbstractLMDescriptionType">
655         <sequence>
656             <choice>
657                 <element name="PictureData" type="base64Binary" minOccurs="0">
658                     <annotation>
659                         <documentation>
660                             Picture of the landmark encoded with base64
661                         </documentation>
662                     </annotation>
663                 </element>
664                 <element name="PictureUrl" type="string" minOccurs="0">
665                     <annotation>
666                         <documentation>
667                             Url of a picture of the landmark.
668                         </documentation>
669                     </annotation>
670                 </element>
671             </choice>
672         </sequence>
673     </extension>
674 </complexContent>
675 </complexType>
676 <complexType name="AbstractLandmarkType" abstract="true">
677     <annotation>
678         <documentation>
679             The super type of all landmarks
680         </documentation>
681     </annotation>

```

```
681 <sequence>
682   <element name="Description" type="xls:AbstractLMDescriptionType">
683     <annotation>
684       <documentation>
685         Every landmark has to be described.
686       </documentation>
687     </annotation>
688   </element>
689 </sequence>
690 <attribute name="Name" type="string" use="optional" />
691 </complexType>
692 <complexType name="StartingPointLMType">
693   <annotation>
694     <documentation>
695       Landmark for orientation at a starting point.
696     </documentation>
697   </annotation>
698   <complexContent>
699     <extension base="xls:AbstractLandmarkType">
700       <choice>
701         <element name="PointPosition" type="gml:PointType">
702           <annotation>
703             <documentation>
704               A starting point landmark can have a point-like geometry
705               encoded as gml:PointType.
706             </documentation>
707           </annotation>
708         </element>
709         <element name="LinePosition" type="gml:LineStringType">
710           <annotation>
711             <documentation>
712               A starting point landmark can have a linear-like geometry
713               encoded as gml:LineStringType.
714             </documentation>
715           </annotation>
716         </element>
717         <element name="AreaPosition" type="gml:PolygonType">
718           <annotation>
719             <documentation>
720               A starting point landmark can have a area-like geometry
721               encoded as gml:PolygonType.
722             </documentation>
723           </annotation>
724         </element>
725       </choice>
726       <attribute name="Orientation" type="gml:CompassPointEnumeration"
727         use="required">
728         <annotation>
729           <documentation>
730             Orientation of landmark respective the starting point
731             using gml:CompassPointEnumeration.
732           </documentation>
733         </annotation>
734       </attribute>
735       <attribute name="SpatialRelation" type="xls:StartLMRelationType"
736         use="optional">
737       <annotation>
738         <documentation>
739           Spatial relation of landmark and starting point.
740         </documentation>
741       </annotation>
742     </extension>
743   </complexContent>
744 </complexType>
<complexType name="EndPointLMType">
```

```

745 <annotation>
746 <documentation>
747     Landmark for orientation at a end point.
748 </documentation>
749 </annotation>
750 <complexContent>
751 <extension base="xls:AbstractLandmarkType">
752 <choice>
753 <element name=" PointPosition" type="gml:PointType">
754 <annotation>
755 <documentation>
756     A end point landmark can have a point-like geometry
757     encoded as gml:PointType.
758 </documentation>
759 </annotation>
760 </element>
761 <element name=" LinePosition" type="gml:LineStringType">
762 <annotation>
763 <documentation>
764     A end point landmark can have a linear-like geometry
765     encoded as gml:LineStringType.
766 </documentation>
767 </annotation>
768 </element>
769 <element name=" AreaPosition" type="gml:PolygonType">
770 <annotation>
771 <documentation>
772     A end point landmark can have a area-like geometry
773     encoded as gml:PolygonType.
774 </documentation>
775 </annotation>
776 </element>
777 </choice>
778 <attribute name=" Orientation" type="gml:CompassPointEnumeration"
779     use="required">
780 <annotation>
781 <documentation>
782     Orientation of landmark respective the starting point
783     using gml:CompassPointEnumeration.
784 </documentation>
785 </annotation>
786 </attribute>
787 <attribute name=" SpatialRelation" type="xls:EndLMRelationType"
788     use="optional">
789 <annotation>
790 <documentation>
791     Spatial relation of landmark and end point.
792 </documentation>
793 </annotation>
794 </attribute>
795 <attribute name=" SideOfRoad" type="xls:SideOfRoadType" use="
796     optional">
797 <annotation>
798 <documentation>
799     Specifying on which side of the road the end point landmark
800     is
801     located. Using an OpenLS simple type
802 </documentation>
803 </annotation>
804 </attribute>
805 </extension>
806 </complexContent>
807 </complexType>
808 <complexType name=" Abstract1ElementLMType" abstract="true">
809 <annotation>
810 <documentation>

```

```
807         The super type of all landmarks identifying only one route
808         element.
809     </documentation>
810 </annotation>
811 <complexContent>
812     <extension base="xls:AbstractLandmarkType" />
813 </complexContent>
814 </complexType>
815 <complexType name="Abstract1EDPLMType" abstract="true">
816     <annotation>
817         <documentation>
818             The super type of all point-landmarks
819         </documentation>
820     </annotation>
821 <complexContent>
822     <extension base="xls:Abstract1ElementLMType" />
823 </complexContent>
824 <complexType name="AreaLM1Type">
825     <annotation>
826         <documentation>
827             The super type of all 1-element areal landmarks
828         </documentation>
829     </annotation>
830 <complexContent>
831     <extension base="xls:Abstract1EDPLMType">
832         <sequence>
833             <element name="PolygonPosition" type="gml:PolygonType">
834                 <annotation>
835                     <documentation>
836                         Position encoded as a gml:PolygonType
837                     </documentation>
838                 </annotation>
839             </element>
840         </sequence>
841     </extension>
842 </complexContent>
843 </complexType>
844 <complexType name="AbstractNElementLMType" abstract="true">
845     <annotation>
846         <documentation>
847             The super type of all landmarks identifying only one route
848             element.
849         </documentation>
850     </annotation>
851 <complexContent>
852     <extension base="xls:AbstractLandmarkType" />
853 </complexContent>
854 <complexType name="AreaLMNType">
855     <annotation>
856         <documentation>
857             The super type of all n-elements areal landmarks
858         </documentation>
859     </annotation>
860 <complexContent>
861     <extension base="xls:AbstractNElementLMType">
862         <sequence>
863             <element name="PolygonPosition" type="gml:PolygonType">
864                 <annotation>
865                     <documentation>
866                         Position encoded as a gml:PolygonType
867                     </documentation>
868                 </annotation>
869             </element>
870         </sequence>
```

```

871     </extension>
872 </complexContent>
873 </complexType>
874 <complexType name="AbstractLineLMType">
875   <annotation>
876     <documentation>
877       The super type of all type representing a landmark
878       with a linear-like function.
879     </documentation>
880   </annotation>
881   <complexContent>
882     <extension base="xls:AbstractNElementLMType">
883       <sequence>
884         <choice>
885           <element name="PolygonPosition" type="gml:PolygonType">
886             <annotation>
887               <documentation>
888                 Position encoded as a gml:PolygonType
889               </documentation>
890             </annotation>
891           </element>
892           <element name="LinePosition" type="gml:LineStringType">
893             <annotation>
894               <documentation>
895                 Position encoded as a gml:LineStringType
896               </documentation>
897             </annotation>
898           </element>
899         </choice>
900       </sequence>
901     </extension>
902   </complexContent>
903 </complexType>
904 <complexType name="IdentifyingLLType">
905   <annotation>
906     <documentation>
907       Representing a landmark with a linear-like function ,
908       which identifies the last DP.
909     </documentation>
910   </annotation>
911   <complexContent>
912     <extension base="xls:AbstractLineLMType">
913       <attribute name="SpatialRelation" type="
914         xls:SpatialRelationLinearIdentType" use="required">
915         <annotation>
916           <documentation>
917             Spatial relation of the linear landmark to the route
918           </documentation>
919         </annotation>
920       </attribute>
921     </extension>
922   </complexContent>
923 </complexType>
924 <complexType name="NotIdentifyingLLType">
925   <annotation>
926     <documentation>
927       Representing a landmark with a linear-like function ,
928       which identifies the last DP.
929     </documentation>
930   </annotation>
931   <complexContent>
932     <extension base="xls:AbstractLineLMType">
933       <sequence>
934         <element name="Landmark" type="xls:Abstract1ElementLMType">
935           <annotation>
936             <documentation>

```

```
936         Landmark that identifies the end of the relation
937         between route and linear function landmark.
938     </documentation>
939 </annotation>
940 </element>
941 </sequence>
942 </extension>
943 </complexContent>
944 </complexType>
945 <complexType name="NonDPPLMType">
946     <annotation>
947         <documentation>
948             Representing a landmark with a linear-like function ,
949             which identifies the last DP.
950         </documentation>
951     </annotation>
952     <complexContent>
953         <extension base="xls:Abstract1ElementLMType">
954             <sequence>
955                 <choice>
956                     <element name="PointPosition" type="gml:PointType">
957                         <annotation>
958                             <documentation>
959                                 Position encoded as a gml:PointType
960                             </documentation>
961                         </annotation>
962                     </element>
963                     <element name="LinePosition" type="gml:LineStringType">
964                         <annotation>
965                             <documentation>
966                                 Position encoded as a gml:LineStringType
967                             </documentation>
968                         </annotation>
969                     </element>
970                     <element name="PolygonPosition" type="gml:PolygonType">
971                         <annotation>
972                             <documentation>
973                                 Position encoded as a gml:PolygonType
974                             </documentation>
975                         </annotation>
976                     </element>
977                 </choice>
978             </sequence>
979             <attribute name="SpatialRelation" type="
980                 xls:SpatialRelationNonDPTType" use="required">
981                 <annotation>
982                     <documentation>
983                         Spatial relation between landmark and route segment.
984                     </documentation>
985                 </annotation>
986             </attribute>
987         </extension>
988     </complexContent>
989 </complexType>
990 <complexType name="GSOLMType">
991     <annotation>
992         <documentation>
993             Representing a landmark with a point-like function ,
994             which belongs to the category general salient object.
995         </documentation>
996     </annotation>
997     <complexContent>
998         <extension base="xls:Abstract1EDPLMType">
999             <sequence>
1000                 <choice>

```

```

1001         <annotation>
1002             <documentation>
1003                 Position encoded as a gml:PointType
1004             </documentation>
1005         </annotation>
1006     </element>
1007     <element name="LinePosition" type="gml:LineStringType">
1008         <annotation>
1009             <documentation>
1010                 Position encoded as a gml:LineStringType
1011             </documentation>
1012         </annotation>
1013     </element>
1014     <element name="PolygonPosition" type="gml:PolygonType">
1015         <annotation>
1016             <documentation>
1017                 Position encoded as a gml:PolygonType
1018             </documentation>
1019         </annotation>
1020     </element>
1021 </choice>
1022 </sequence>
1023 <attribute name="SpatialRelation" type="
1024     xls:SpatialRelationGSOType" use="required">
1025     <annotation>
1026         <documentation>
1027             Spatial relation between landmark and route segment.
1028         </documentation>
1029     </annotation>
1030 </attribute>
1031 </extension>
1032 </complexContent>
1033 </complexType>
1034 <complexType name="StreetnameLMType">
1035     <annotation>
1036         <documentation>
1037             Representing a landmark with a point-like function ,
1038             which belongs to the category streetname.
1039         </documentation>
1040     </annotation>
1041     <complexContent>
1042         <extension base="xls:Abstract1EDPLMType">
1043             <sequence>
1044                 <element name="PositionAtIntersection" type="gml:AngleType">
1045                     <annotation>
1046                         <documentation>
1047                             Position at intersection encoded as a gml:AngleType
1048                         </documentation>
1049                     </annotation>
1050                 </element>
1051             </sequence>
1052         </extension>
1053     </complexContent>
1054 </complexType>
1055 <complexType name="StructureLMType">
1056     <annotation>
1057         <documentation>
1058             Representing a landmark with a point-like function ,
1059             which belongs to the category structure.
1060         </documentation>
1061     </annotation>
1062     <complexContent>
1063         <extension base="xls:Abstract1EDPLMType">
1064             <sequence>
1065                 <element name="Intersection" type="xls:AbstractJunctionType">

```

```
1066         <documentation>
1067             Type of intersection encoded as a AbstractJunctionType
1068         </documentation>
1069     </annotation>
1070 </element>
1071 </sequence>
1072 </extension>
1073 </complexContent>
1074 </complexType>
1075 <!-- Simple Types -->
1076 <simpleType name=" RoadDirectionType">
1077     <annotation>
1078         <documentation>
1079             Enumeration of possible turn directions along a road
1080             at the start of a route.
1081         </documentation>
1082     </annotation>
1083     <restriction base=" string">
1084         <enumeration value=" straight" />
1085         <enumeration value=" left" />
1086         <enumeration value=" right" />
1087     </restriction>
1088 </simpleType>
1089 <simpleType name=" StartLMRelationType">
1090     <annotation>
1091         <documentation>
1092             Enumeration of possible spatial relations of a start landmark to
1093             the starting point of a route.
1094         </documentation>
1095     </annotation>
1096     <restriction base=" string">
1097         <enumeration value=" towards" />
1098         <enumeration value=" away" />
1099     </restriction>
1100 </simpleType>
1101 <simpleType name=" EndLMRelationType">
1102     <annotation>
1103         <documentation>
1104             Enumeration of possible spatial relations of a end landmark to
1105             the destination of a route.
1106         </documentation>
1107     </annotation>
1108     <restriction base=" string">
1109         <enumeration value=" opposite" />
1110         <enumeration value=" left" />
1111         <enumeration value=" right" />
1112     </restriction>
1113 </simpleType>
1114 <simpleType name=" SpatialRelationLinearIdentType">
1115     <annotation>
1116         <documentation>
1117             Enumeration of possible spatial relations of a linear-function
1118             landmark to the route.
1119         </documentation>
1120     </annotation>
1121     <restriction base=" string">
1122         <enumeration value=" along" />
1123         <enumeration value=" after" />
1124     </restriction>
1125 </simpleType>
1126 <simpleType name=" SpatialRelationNonDPTType">
1127     <annotation>
1128         <documentation>
1129             Enumeration of possible spatial relations of a linear-function
1130             landmark to the route.
1131         </documentation>
```

```

1132     </annotation>
1133     <restriction base="string">
1134         <enumeration value="pass" />
1135         <enumeration value="cross" />
1136         <enumeration value="through" />
1137     </restriction>
1138 </simpleType>
1139 <simpleType name="SpatialRelationGSOType">
1140     <annotation>
1141         <documentation>
1142             Enumeration of possible spatial relations of a linear-function
1143             landmark to the route.
1144         </documentation>
1145     </annotation>
1146     <restriction base="string">
1147         <enumeration value="at" />
1148         <enumeration value="after" />
1149         <enumeration value="before" />
1150     </restriction>
1151 </simpleType>
1152 <simpleType name="TurnDirectionTFType">
1153     <annotation>
1154         <documentation>
1155             Enumeration of possible spatial relations of a linear-function
1156             landmark to the route.
1157         </documentation>
1158     </annotation>
1159     <restriction base="string">
1160         <enumeration value="left" />
1161         <enumeration value="right" />
1162     </restriction>
1163 </simpleType>
1164 <simpleType name="TurnDirectionSIType">
1165     <annotation>
1166         <documentation>
1167             Enumeration of possible spatial relations of a linear-function
1168             landmark to the route.
1169         </documentation>
1170     </annotation>
1171     <restriction base="string">
1172         <enumeration value="left" />
1173         <enumeration value="right" />
1174         <enumeration value="straight" />
1175         <enumeration value="slightLeft" />
1176         <enumeration value="slightRight" />
1177         <enumeration value="sharpLeft" />
1178         <enumeration value="sharpRight" />
1179     </restriction>
1180 </simpleType>
1181 <simpleType name="TurnDirectionCType">
1182     <annotation>
1183         <documentation>
1184             Enumeration of possible spatial relations of a linear-function
1185             landmark to the route.
1186         </documentation>
1187     </annotation>
1188     <restriction base="string">
1189         <enumeration value="left" />
1190         <enumeration value="right" />
1191     </restriction>
1192 </simpleType>
1193 <simpleType name="TurnDirectionSRType">
1194     <annotation>
1195         <documentation>
1196             Enumeration of possible spatial relations of a linear-function
1197             landmark to the route.

```

```
1198     </documentation>
1199 </annotation>
1200 <restriction base="string">
1201   <enumeration value="left"/>
1202   <enumeration value="straight"/>
1203   <enumeration value="right"/>
1204 </restriction>
1205 </simpleType>
1206 <simpleType name="TurnDirectionChunkType">
1207   <annotation>
1208     <documentation>
1209       Enumeration of possible spatial relations of a linear-function
1210       landmark to the route.
1211     </documentation>
1212   </annotation>
1213   <restriction base="string">
1214     <enumeration value="left"/>
1215     <enumeration value="right"/>
1216   </restriction>
1217 </simpleType>
1218 </schema>
```

Bibliography

- [1] Bychowski, T. (2003): OpenGIS Location Services (OpenLS): Part 6 – Navigation Service. OGC Implementation Specification 03-007r1 (Version 0.5.0). Open GIS Consortium Inc.
- [2] Cox, S., Daisey, P., Lake, R., Portele, C., Whiteside, A. (2004): OpenGIS Geography Markup Language (GML) Implementation Specification 03-105r1 Version 3.1.0. Open Gis Consortium Inc.
- [3] Dale, R., Geldof, S., Prost, J.-P. (2003): CORAL: Using natural language generation for navigational assistance. In: Oudshoorn, M. (Ed.), Proceedings of the 26th Australasian Computer Science Conference (ACSC2003), Adelaide, Australia.
- [4] Denis, M. (1997): The description of routes: A cognitive approach to the production of spatial discourse. *Cahiers de Psychologie Cognitive*, 16:409-458.
- [5] Denis, M., Pazzaglia, F., C.Cornoldi & Bertolo, L. (1999): Spatial discourse and navigation: An analysis of route directions in the city of Venice. *Applied Cognitive Psychology* 13:145-174.
- [6] Freed, N., Borenstein, N., (1996): Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. Internet Engineering Task Force.
- [7] Hansen, S., Richter, K.-F., Klippel, A. (2006): Landmarks in OpenLS - a data structure for cognitive ergonomic route directions. In M. Raubal, H. Miller, A. U. Frank, M. F. Goodchild (Eds.), *Geographic Information Science - Fourth International Conference, GIScience 2006* (pp. 128–144). Springer, Berlin.
- [8] International Organization for Standardisation (ISO), (2001): ISO 19118 Draft International Standard: Geographic Information – Encoding. Document ISO/TC211 N1136, Technical Committee 211, ISO Secretariat, Geneva, Switzerland.
- [9] Klippel, A, Dewey, C., Knauff, M., Richter, K.-F., Montello, D. R., Freksa, C., Loeliger, E.-A. (2004): Direction Concepts in Wayfinding Assistance Systems. In: Baus, J., Kray, C., Porzel, R. (Eds.), *Workshop on Artificial Intelligence in Mobile Systems 2004 (AIMS'04)*, SFB 378 Memo 84, Saarbrücken, pp. 1-8.

- [10] Klippel, A., Hansen, S., Davies, J., Winter, S. (2005): A High-Level Cognitive Framework For Route Directions. Proceedings of SSC 2005 Spatial Intelligence, Innovation and Praxis: The national biennial Conference of the Spatial Science Institute. September 2005. Melbourne: Spatial Science Institute. ISBN 0-9581366-2-9
- [11] Klippel, A., Richter, K.-F., Hansen, S. (2005): Structural salience as a landmark. Workshop Mobile Maps 2005, Salzburg, Austria.
- [12] Klippel, A., Tappe, T., Habel, C. (2003): Pictorial representations of routes: Chunking route segments during comprehension. In C. Freksa, W. Brauer, C. Habel, and K.F. Wender (Eds.), Spatial Cognition III. Routes and Navigation, Human Memory and Learning, Spatial Representation and Spatial Learning (pp. 11-33). Springer, Berlin.
- [13] Klippel, A., Tappe, T., Kulik, L., Lee, P.U. (2005): Wayfinding choremes - A language for modeling conceptual route knowledge. Journal of Visual Languages and Computing, 16(4):311-329.
- [14] Klippel, A., Tenbrink, T., Montello, D. R. (submitted): The role of structure and function in the conceptualization of directions.
- [15] Lovelace, K. L., Hegarty, M., and Montello, D. R. (1999). Elements of good route directions in familiar and unfamiliar environments. In C. Freksa & D. M. Mark (eds.), Spatial Information Theory - Cognitive and Computational Foundations of Geographic Information Science, (pp. 65-82). International Conference COSIT, Berlin: Springer.
- [16] Mabrouk, M. (2005): OpenGIS Location Services (OpenLS): Core Services. OGC Implementation Specification 05-016 Version 1.1. Open GIS Consortium Inc.
- [17] Michon, P.-E., Denis, M. (2001): When and why are visual landmarks used in giving directions? In: Montello, D.R. (Ed.), Spatial Information Theory. Foundations of Geographic Information Science (pp. 292-305). International Conference, COSIT 2001. Springer, Berlin.
- [18] The Open Geospatial Consortium (OGC). <http://www.opengis.org>.
- [19] Richter, K.-F., Klippel, A. (2005): A model for context-specific route directions. In: Freksa, C., Knauff, M., Krieg-Brückner, B., Nebel, B., Barkowsky, T. (Eds.), Spatial Cognition IV. Reasoning, Action, and Interaction: International Conference Spatial Cognition 2004 (pp. 58-78). Springer, Berlin.