ADASIS Forum

ADASIS v3 Reference

Specification v3.1.0

Draft 1

July 05 2018

Document Control

Document title	ADASIS v3 Protocol Reference
Electronic name	
Main author(s) or editor(s)	
Current author(s)	
Contributing author(s)	

Copyright notes

Document history

Version	3.1.0
Status	
Date	
Authors	
Reviewers	
Summary of Changes	

Abstract

This document provides an API reference for profile types and the Franca IDL types defined by the ADASIs v3 protocol specification.

1 ADASIS v3 Versioning

With official release of ADASIS v3 (3.X.Y.Z) the numbering of versions will not follow strict rules regarding compatibility.

It is expected that major changes for example introduction of new messages or complete change of profiles will increase the "X" version. Minor changes for example extensions of profiles or structs would increase the "Y" version. Bugfixes are implied through increase of "Z".

The version number gives no strict indication about backwards compatibility to older ADASIS v3 versions. It is expected that most updates are downwards compatible if implementer regards following rules. If issues arise with updates of the protocol, it will be mentioned here.

1.1.1 Rules

List of changes which should be downwards compatible:

- **Everybody can/must take into consideration** that
    - (1) it is guaranteed that **no enum value is reassigned, reordered or shifted**; explicit enumeration is used
    - (2) new profile types might be added, therefore it **should be possible to skip "unknown" profile types**
    - (3) it might happen that **new fields will be added at end of a structure**
        - serializer should compatible in both direction and don't do repacking
        - therefore it is accepted that data volume could increase without repacking

2 Profiles Reference

ADASIS v3 offers a rich set of standardized profiles which are summarily listed below. Furthermore, custom profiles can be added as needed for a specific implementation.

2.1 Intersection Profiles

In ADASIS v3, intersections are represented as nodes – locations along a path where multiple roads meet. A node is one single location, i.e. one offset; if a logical intersection has some roads meeting at multiple locations (offsets), even within a very short distance, it is represented in ADASIS v3 as multiple nodes.

Intersections are not tightly linked to side paths anymore, as it was the case in ADASIS v2 with the STUB message. Side paths do not technically need a corresponding node, though in reality, the branching point of a side path will be an intersection that can be described by a node. But if a node is provided, it can give additional information on the relationship between parent and side path (as was provided by the STUB in ADASIS v2) – for this purpose there is a cross-reference between paths and the arms of a node.

2.1.1 Node

The Node profile is used to describe intersections.

Property	Value

Profile Interpolation Type	Spot
Profile Value	NodeProfileValue

2.2 Basic Geometry Profiles

The following profiles describe the road geometry in a simple way that is sufficient for most classical ADAS purposes, without reference to absolute coordinates.

2.2.1 Heading Change

This profile represents the heading angle change of the path at the points of a polyline representation of the path.

Property	**Value**
Profile Interpolation Type	Spot
Profile Value	FloatProfileValue
Unit	deg

2.2.2 Curvature

This profile represents the curvature of the path.

In reality, curvature usually changes continuously along a path. In order to get a good representation of actual curvature, maps and horizon providers need to provide a high number of curvature values, compared to other profiles. To accommodate this high data density without creating an excessive data transfer volume, the ADASIS v3 Curvature profile uses a special data representation where each profile value (“OffsetFloatProfileValue” in Franca) is an array containing multiple pairs of offset and curvature value. The semantics is the same, though, as it would be with sending each curvature value in a separate profile entry for its corresponding offset.
The validity range of the Curvature profile entry is the union of the validity range of all the array entries, i.e. it extends from the offset of the first entry to a point some distance behind its last entry, where the first array element of a subsequent Curvature profile entry would be.
Logically, the curvature values should be interpolated linearly. This interpolation, though, needs to happen between individual curvature array elements – the array-valued profile entries themselves cannot be interpolated. Therefore the Curvature profile is marked as special interpolation, the arrays are passed unchanged. It would be useful for a reconstructor, though, to dissolve the curvature arrays, to store the individual curvature entries in the same way like separate scalar-valued profile values, and then to do linear interpolation between them.

Property	**Value**
Profile Interpolation Type	Special
Profile Value	OffsetFloatProfileValue
Unit	1/m

2.2.3 Slope

This profile represents the slope information of a path.

For Slope, the same considerations with respect to data density apply as for Curvature, and consequently the Slope profile is constructed and handled using array of offsets and values in just the same way as described in the preceding paragraph for Curvature.

Property	**Value**
Profile Interpolation Type	Special
Profile Value	OffsetFloatProfileValue
Unit	%

2.3 Road Model Profiles

The following profiles provide a detailed description of the structure of the road and of the possible ways of driving along it. They describe how the road consists of lanes, if and how lanes can be driven, how the lanes (and their boundaries) look like; and they give the absolute geometry of the lanes and of the road in total.

2.3.1 Road Geometry

This profile provides the geometry of the road reference line, usually as a polyline. Typically, the road reference line is the road centerline, but this is dependent on the underlying road map.

Property	**Value**
Profile Interpolation Type	Spot
Profile Value	RoadGeometryProfileValue

2.3.2 Lane Model

In the real world, a road consists of one or two carriageways resp. roadways, each with one or more lanes and any associated pavement resp. sidewalks, e.g., a footpath, and road verges.

A carriageway consists of a width of road on which a vehicle is not restricted by any physical barriers or separation to move laterally. A carriageway consists of a number of traffic lanes together with any associated shoulder, but may be a sole lane in width, e.g., a highway interchange.

A lane is part of a carriageway that is designated for use by a single line of vehicles, to control and guide drivers and reduce traffic conflicts. A lane is usually a laterally delimited area as part of a carriageway.

A lane has some technical characteristics such as:

- A lane is a part of a carriageway and a carriageway is part of a road.
- A lane is usually a laterally delimited area on the carriageway, depending on the type of usage, the allowed driving direction.
- For each road segment with a specific length, a lane can be delimited by left and right borders, each border is either a visible or a virtual boundary line.
- A lane has defined start and end through opening or closing part, alternatively through splitting of a lane in two or more lanes respectively merging two or more lanes in one lane.

There must be at least one lane to describe the road.

Property	Value
Profile Interpolation Type	Spot
Profile Value	LaneModelValue

2.3.2.1 *Examples*

TODO: Check the two changes figures, check third new figure (and check captions + figure numbers!)

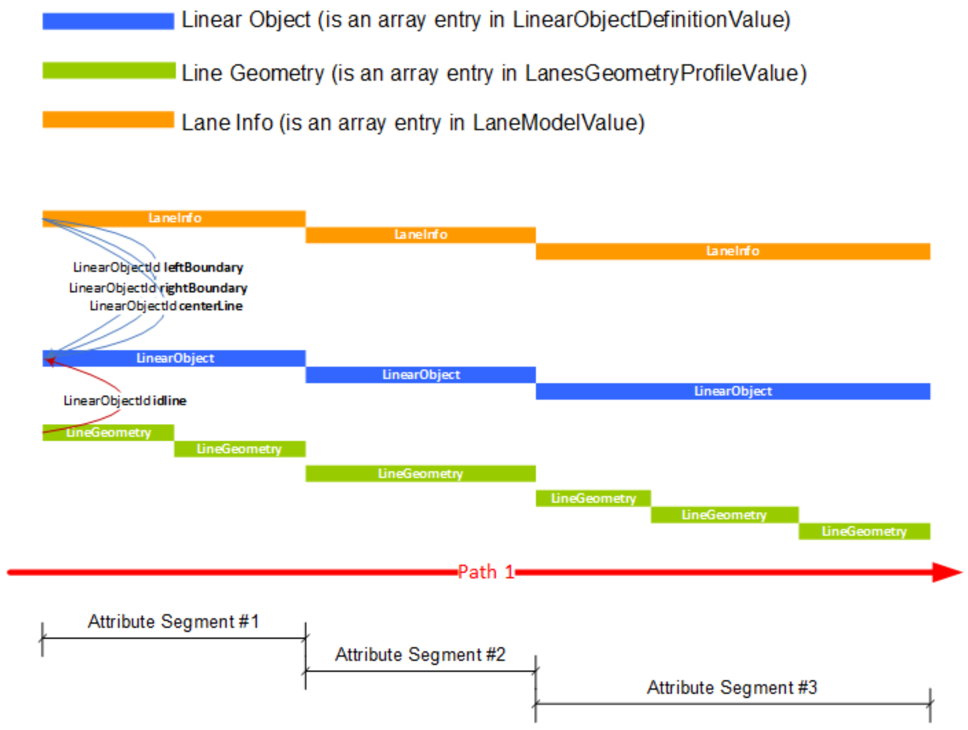

Linear Object (is an array entry in LinearObjectDefinitionValue)
Line Geometry (is an array entry in LanesGeometryProfileValue)
Lane Info (is an array entry in LaneModelValue)
LaneInfo
LaneInfo
LaneInfo
LinearObjectId leftBoundary
LinearObjectId rightBoundary
LinearObjectId centerLine
LinearObject
LinearObject
LinearObject
LinearObjectId idline
LineGeometry
LineGeometry
LineGeometry
LineGeometry
LineGeometry
LineGeometry
Path 1
Attribute Segment #1
Attribute Segment #2
Attribute Segment #3

Figure 1 Lane Model details

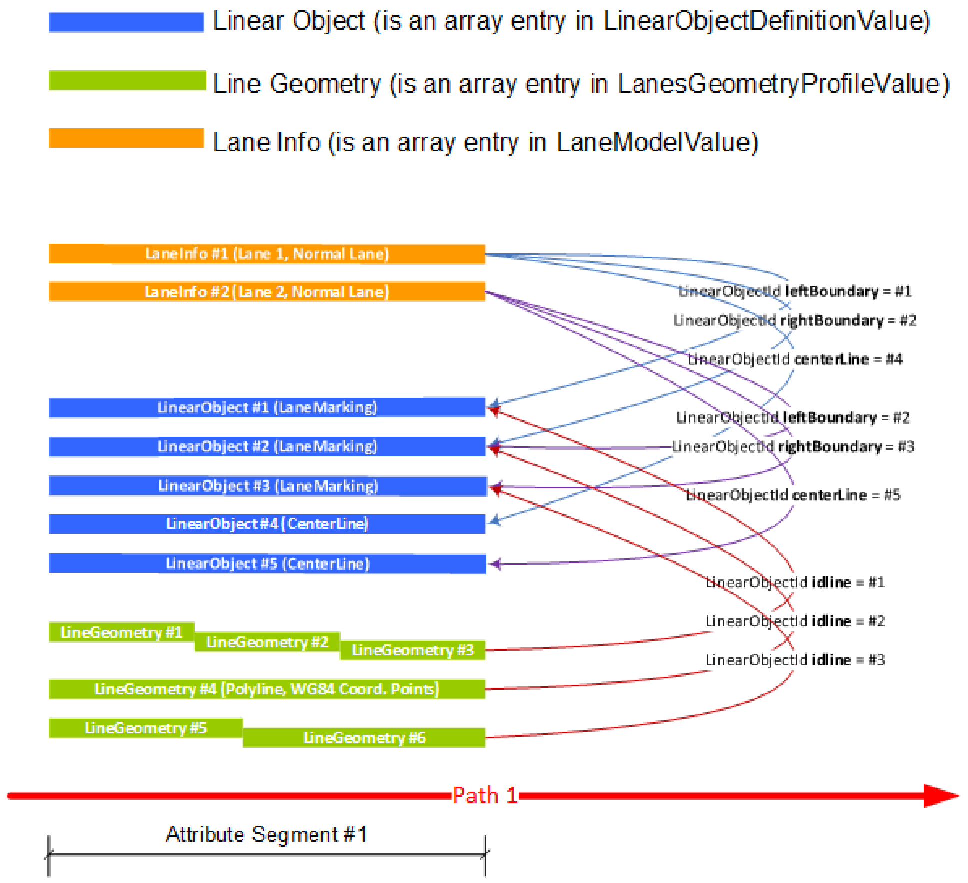

Linear Object (is an array entry in LinearObjectDefinitionValue)
Line Geometry (is an array entry in LanesGeometryProfileValue)
Lane Info (is an array entry in LaneModelValue)
LaneInfo #1 (Lane 1, Normal Lane)
LaneInfo #2 (Lane 2, Normal Lane)
LinearObject #1 (LaneMarking)
LinearObject #2 (LaneMarking)
LinearObject #3 (LaneMarking)
LinearObject #4 (CenterLine)
LinearObject #5 (CenterLine)
LineGeometry #1
LineGeometry #2
LineGeometry #3
LineGeometry #4 (Polyline, WG84 Coord. Points)
LineGeometry #5
LineGeometry #6
LinearObjectId leftBoundary = #1
LinearObjectId rightBoundary = #2
LinearObjectId centerLine = #4
LinearObjectId leftBoundary = #2
LinearObjectId rightBoundary = #3
LinearObjectId centerLine = #5
LinearObjectId idline = #1
LinearObjectId idline = #2
LinearObjectId idline = #3
Path 1
Attribute Segment #1

Figure 2 Lane Model details hierarchy

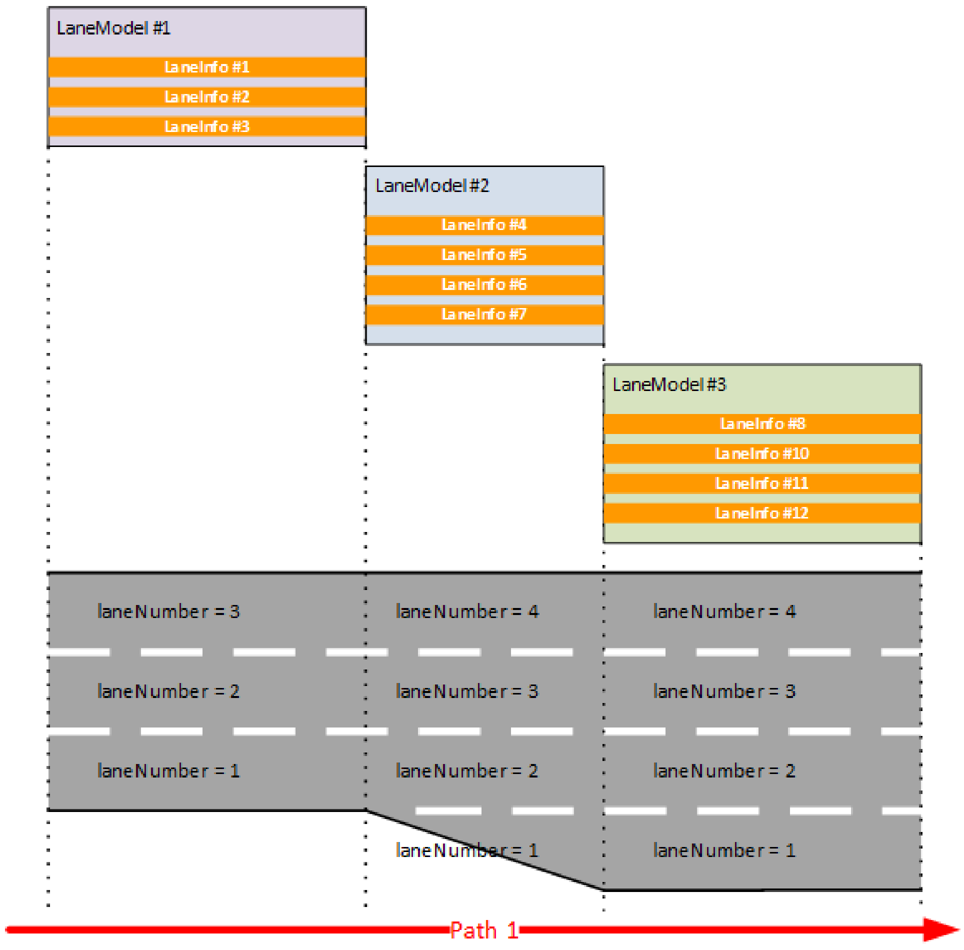


Figure 3 Usage example of Lane Model and Lane Info

2.3.3 Linear Objects

Linear objects describe various kinds of real and virtual objects that can be represented by a line roughly in the direction of the lane:

- Lane boundaries, lane markings
- Lane center lines
- Physical lane dividers, kerbs, guard rails, fences, walls

Linear objects representing lane markings also describe the kind and color of the marking. As a special case, a lane boundary can also be a non-marked line; these are treated as a special case of lane marking lines.

A lane center line describes the expected trajectory of the center of a vehicle using this lane. This also covers “virtual lanes” in an intersection area.

The linear objects profile consists of a list of linear objects, ordered from the outermost to the innermost linear object (i.e. starting from the rightmost one for right-hand traffic and from the leftmost one for left-hand traffic). This enables a client to correctly identify and order all linear objects that exist between two adjacent lanes (i.e. between the left boundary of the right lane and the right boundary of the left lane). “Innermost” refers to the other side of the road, which would be the outermost for opposing vehicles, as the Linear Objects profile can include linear objects for the complete road, including those which border lanes of opposite driving direction.

The linear object profile entries must be aligned with lane model and lane geometry profile entries and may not overlap with other linear object profile entries or with (parts of) two or more lane descriptions or geometries.

Property	**Value**
Profile Interpolation Type	Step
Profile Value	LinearObjectDefinitionValue

2.3.4 Lane Geometry

A lane geometry description defines the geometry of all lanes for a section of a path (a section being defined as a range of offsets). This path section should be described by a single lane description (giving a number of lanes, lane types, etc.). In other words, a line geometry description may not overlap with another lane geometry description, and it may not overlap with (parts of) two or more lane descriptions.

The lane geometry is given by the geometries of the linear objects describing the lanes. Primary elements are lane center lines and lane boundaries.

All lines are optional. Lane center lines might be skipped for non-drivable lanes, and lane boundary lines might (but don’t have to) be skipped if not marked on the road.

Property	**Value**
Profile Interpolation Type	Step
Profile Value	LanesGeometryProfileValue

2.3.4.1 Examples

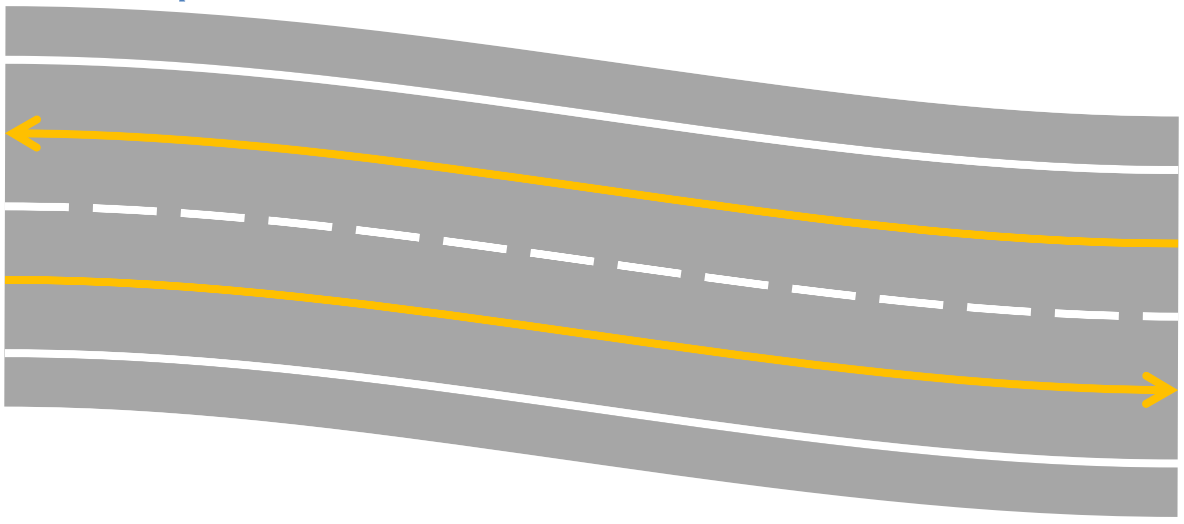


Figure 4 Geometry Example

In Figure 4, we have a common two-lane road with the two lane center lines marked in orange. We have three lanes marking lanes serving as lane boundaries (the center line shared between the lanes). If we wish, we can add the actual roadbed boundaries as auxiliary lines; they are a few centimeters outside of the outer lane markings.

Logically all lane center lines and line boundary lines in a geometry description begin at the same offset along a path, and they end along the same offset along a path. This cannot be made an exact requirement, though: “at the same offset” should mean that lane center/boundary line starts points and end points should be on a perpendicular line to the road center line at that offset. But the heading of the road center line is not always defined exactly, and thus neither is the direction of a perpendicular line.

Each lane center or boundary line can be defined in one of multiple ways. Either it is a polyline, which is a sequence of at least two WGS84 latitude/longitude pairs. Or it is a Bezier spline, which is a sequence of Bezier curves. A Bezier spline consisting of n individual Bezier curves is represented by a sequence of (3n+1) WGS84 latitude/longitude pairs. The first to fourth pairs define the first Bezier curve, the fourth to seventh pairs define the second Bezier curve, etc. – so two adjacent Bezier curves always share a coordinate pair which represents the point connecting these two Bezier curves. For each Bezier curve, the first and last of its four coordinate pairs define the start and end point of the curve while the two intervening pairs define control points of the Bezier curve.
As a consequence, a Bezier spline can be converted into a polyline by keeping the first, fourth, seventh, etc. point and dropping the control points.

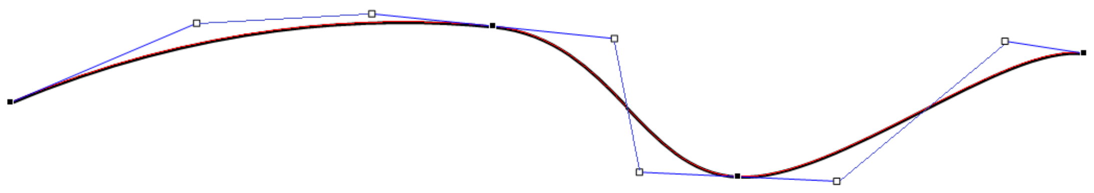


Figure 5 an example of Bezier spline consisting of three Bezier curves and described by 10 points.

Further ways of defining a line may be specified in future versions of this standard (e.g. clothoids or definitions using the offset to another line).

In any case the lines are described in the direction of the path, i.e. the start point of a line corresponds to the lowest offset of the path section and the end point of a line corresponds to the highest offset of a path section.

Sequence of line geometry with the same object ID is allowed in case the line geometry does not fit into one profile message in case of bandwidth limitation; e.g. polyline. In this case, the start offset of the first line geometry is the start offset of the whole line, the end offset of the first line geometry is the start offset of the second line geometry with the same object ID … etc. and the end offset of the last line geometry is the end offset of the whole line.

2.3.5 Lane Connectivity

Lanes are part of a road segment. And typically, road segments are “segments” of the road with a specific characteristic. The most significant property is the *number of lanes*. Therefore from each road segment to the following segment there must be a *connection* which links the lanes of different segments.

Property	**Value**
Profile Interpolation Type	Spot
Profile Value	LaneConnectivityValue

The lane connections between one road segment and the following road segment are described by the **LaneConnectivity Profile**. In Figure 6 Lane Connectivity Example, lanes connections (identified with ID1, ID2 … etc.) are illustrated as logical points – or nodes – where two road segments are connected.

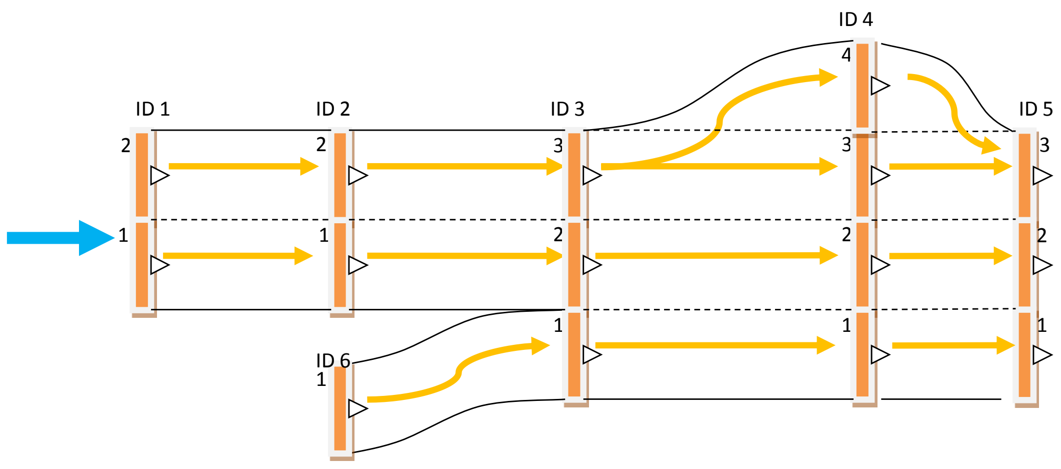


Figure 6 Lane Connectivity Example

Each lane of a segment is connected with no, one or more lanes from at least one predecessor segment. In case of path branches also lanes from several road segments could be connected. The logical point for all connections from one road segment to another is the same and therefore one specific offset.

In contrast to the lane connectivity *between* road segments, *transition zones* are embedded into one road segment. As a result, each **opening, closing, merging or splitting of a lane takes place within a single road segment** and is finished until road segment ends.
They have to be considered as single lanes with property „transition” which is part of the Lane Model Profile.

The Lane Connectivity profile contains information about how the lanes on a path or several different paths are connected. With every Lane Model profile entry, an Av3HP should generate a **corresponding Lane Connectivity** profile entry at the same offset on the path. The Lane Connectivity profile entry contains a list of connectivity pairs that show how the lanes described in the Lane Model profile entry are reachable from the lanes defined previously on the same path or on different paths.
This list does not include lanes which can be reached only through a regular lane change maneuver.

The LaneConnectivity Profile entry contains one *LaneConnectivityPair* for each combination: If one lane has more than one successor, there has to be one pair for each successor. The other way round for several predecessors connected to one lane.

If there is no connection to or from a lane (a lane starts without predecessors or ends without a successor) is indicated **implicitly through absence of a connectivity pair**. Some examples of Lane Connectivity pairs for different situations is give below

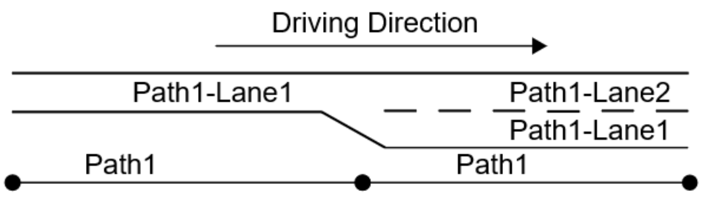


Figure 7 New Lane on right side

Situation: New Lane on right side

Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane1, Path1 : Lane2)

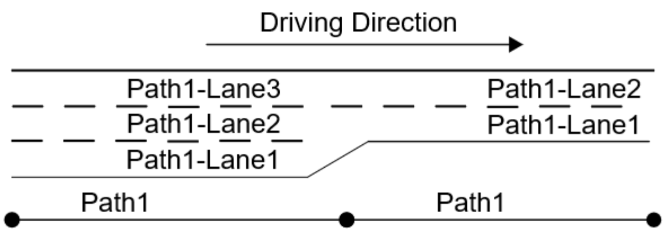


Figure 8 Lane ends

Situation: Lane ends

Pairs:

- (Path1 : Lane2, Path1 : Lane1)
- (Path1 : Lane3, Path1 : Lane2)

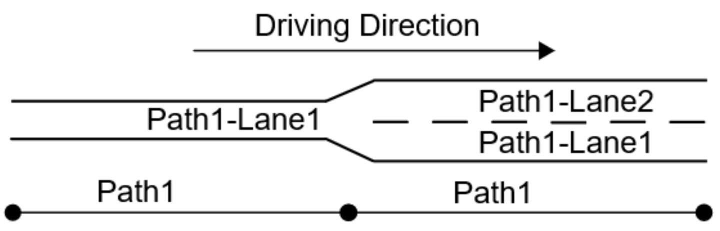


Figure 9 Single Lane widens into 2 Lanes

Situation: Single Lane widens into 2 Lanes

Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane1, Path1 : Lane2)

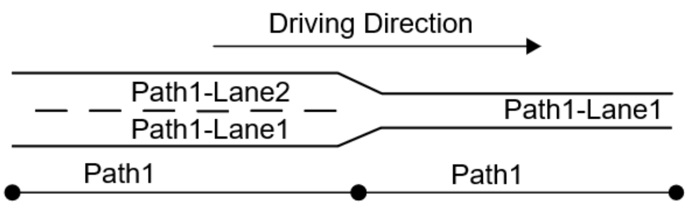


Figure 10 Merging Lanes without Priority

Situation: Merging Lanes without Priority

Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane2, Path1 : Lane1)

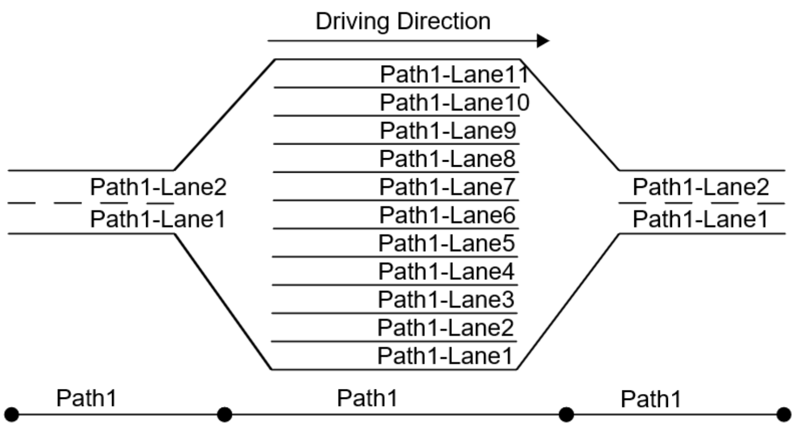


Figure 11 Toll Plaza

Situation: Toll Plaza

First connectivity point Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane1, Path1 : Lane2)
- (Path1 : Lane1, Path1 : Lane3)
- (Path1 : Lane1, Path1 : Lane4)
- (Path1 : Lane1, Path1 : Lane5)
- (Path1 : Lane1, Path1 : Lane6)
- (Path1 : Lane2, Path1 : Lane7)
- (Path1 : Lane2, Path1 : Lane8)
- (Path1 : Lane2, Path1 : Lane9)
- (Path1 : Lane2, Path1 : Lane10)
- (Path1 : Lane2, Path1 : Lane11)

Second connectivity point Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane2, Path1 : Lane1)
- (Path1 : Lane3, Path1 : Lane1)
- (Path1 : Lane4, Path1 : Lane1)
- (Path1 : Lane5, Path1 : Lane1)
- (Path1 : Lane6, Path1 : Lane1)
- (Path1 : Lane7, Path1 : Lane2)
- (Path1 : Lane8, Path1 : Lane2)
- (Path1 : Lane9, Path1 : Lane2)
- (Path1 : Lane10, Path1 : Lane2)
- (Path1 : Lane11, Path1 : Lane2)

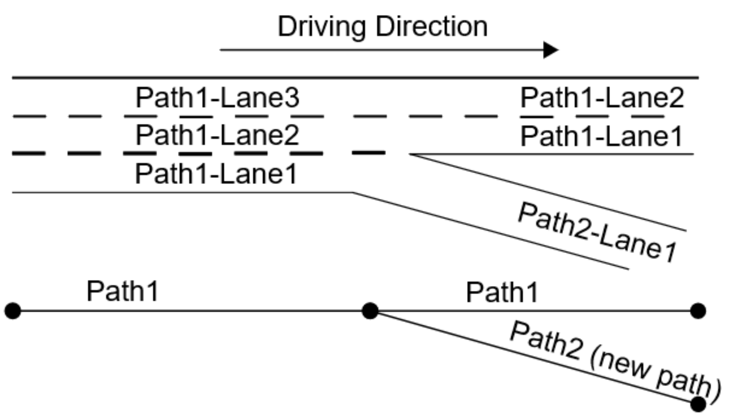


Figure 12 Exit Lane

Situation: Exit Lane

Pairs:

- (Path1 : Lane1, Path2 : Lane1)
- (Path1 : Lane2, Path1 : Lane1)
- (Path1 : Lane3, Path1 : Lane2)

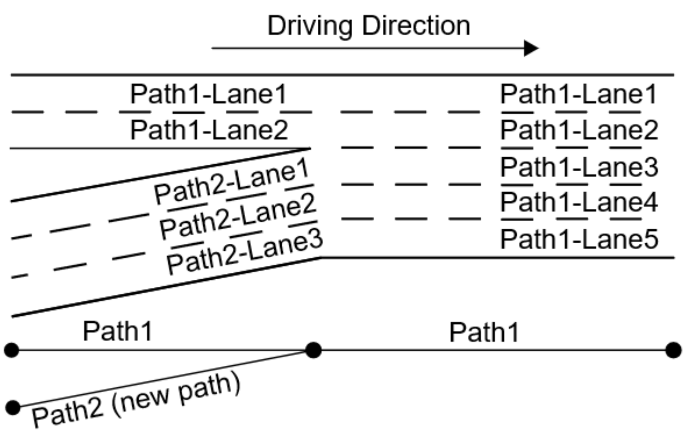


Figure 13 Merging Motorways

Situation: Exit Lane

Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane2, Path1 : Lane2)
- (Path2 : Lane1, Path1 : Lane3)
- (Path2 : Lane2, Path1 : Lane4)

- (Path2 : Lane3, Path1 : Lane5)

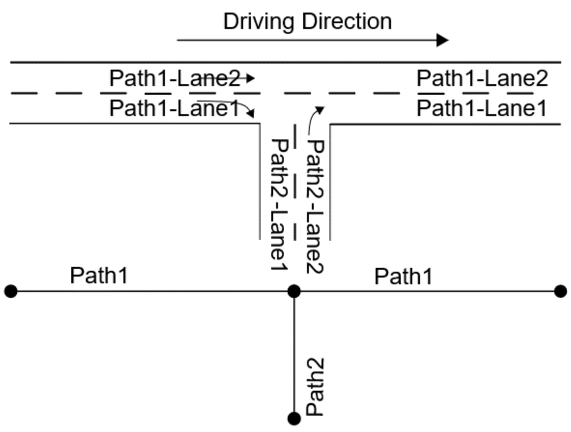


Figure 14 Turning Lane

Situation: Turning Lane

Pairs:

- (Path1 : Lane1, Path1 : Lane1)
- (Path1 : Lane2, Path1 : Lane2)
- (Path1 : Lane1, Path2 : Lane1)
- (Path2 : Lane2, Path1 : Lane1)

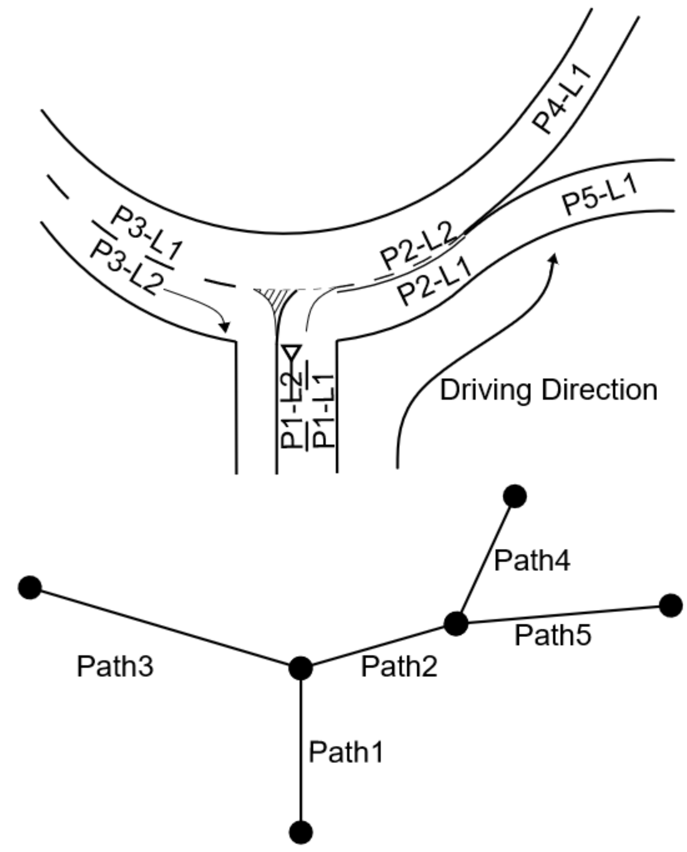


Figure 15 Roundabout

Situation: Roundabout

First connectivity point Pairs:

- (Path1 : Lane1, Path2 : Lane1)
- (Path1 : Lane1, Path2 : Lane2)
- (Path1 : Lane3, Path3 : Lane2)

Second connectivity point Pairs:

- (Path2 : Lane1, Path5 : Lane1)
- (Path2 : Lane2, Path4 : Lane1)

2.3.6 Number of Lanes per Direction

This profile represents the number of lanes on a specific path in a specific relative driving direction.

Usually two profile entries exist to describe the number of lanes in one direction and the number of lanes in the opposite direction separately. Bidirectional lanes are counted in both directions, thus the sum of the number of lanes drivable in one direction and drivable in the opposite direction is equal or greater than the total number of drivable lanes. To derive the total number of drivable lanes it is advised to transmit a third profile entry containing the number of lanes drivable in both directions which is subtracted from the sum. Alternatively if the Lane Model profile contains a representation of all lanes, the total number of lanes can be derived. Thus it might be useful to transmit the Lane Model profile with all lanes even if the Lane Info entries have no further lane information and no references to left resp. right border lines and centerlines.

Property	**Value**
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue

2.3.7 Lane Width

The lane width profile provides the width of the drivable area on a lane in centimeters. This information can be used as a simplification, instead of calculating it from the lane boundary geometries.

Property	**Value**
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue containing Distance

2.3.8 Location Object

This profile provides information on objects near the road that can be used to precisely locate the ego vehicle.

Property	**Value**
Profile Interpolation Type	Spot
Profile Value	LocationObject

2.4 Speed Profiles

There are several profiles describing the speeds that may be driven or actually are driven on a road (or on a set of lanes). They share the issue of having the choice between two systems of measurement, metric units using km/h and imperial units using mph. (While m/s would also be a metric unit, it is not used for speed profiles in ADASIS v3.).).

A special data type *Speed* exists in ADASIS v3 to combine both the numerical value and the unit of measurement.

2.4.1 Effective Speed Limit

The effective speed limit describes the speed limit that currently applies to the ego vehicle. Besides the limit itself, there is brief information whether the speed limit is signposted or derived from general legal regulations, and whether it is dependent on conditions. In the latter case, the conditions obviously are fulfilled; otherwise the speed limit would not be effective.

Property	**Value**
Profile Interpolation Type	Step
Profile Value	EffectiveSpeedLimit

2.4.2 Extended Speed Limit

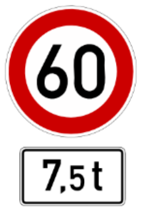
The Extended Speed Limit profile can describe speed limits that might apply to a road (or set of lanes), possibly depending on a variety of conditions. This profile allows providing information on speed limits that do not (currently) apply to the ego vehicle – possibly applying to other vehicles, possibly not applying currently at all.

Besides the actual speed limit value and a list of conditions (empty for an unconditional speed limit), an indication is provided whether the speed limit is signposted or derived from general legal regulations.

Property	Value
Profile Interpolation Type	Step
Profile Value	ExtendedSpeedLimitValue

2.4.3 Examples

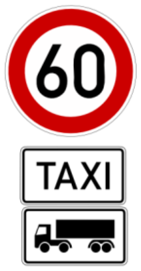
2.4.3.1 60 km/h with weight restriction



situation: vehicle is car

```
Path 2, offset 0: ExtendedSpeedLimitValue{
      value: 60,
      source: explicit,
      conditions {
            ConditionNumeric {
                  type: conditionTypeWeight,
                  appliesToEgoVehicle: false,
                  value: 7500
            }
      }
}
```

2.4.3.2 Example 60 km/h for Taxi and Truck



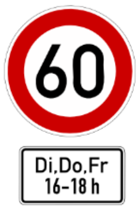
situation: vehicle is taxi

```
Path 2, offset 0: ExtendedSpeedLimitValue {
        value: 60,
        source: explicit,
        conditions {
                ConditionVehicleType {
                        type: conditionTypeVehicle,
                        appliesToEgoVehicle: true,
                        vehicleTypeMask: Taxi
                }
        }
}
```

vehicle is truck

```
Path 2, offset 0: ExtendedSpeedLimitValue {
        value: 60,
        source: explicit,
        conditions {
                ConditionVehicleType {
                        type: conditionTypeVehicle,
                        appliesToEgoVehicle: true,
                        vehicleTypeMask: Truck
                }
        }
}
```

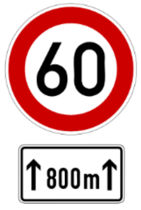
2.4.3.3 Example 60 km/h with time and day restrictions



situation: vehicle is car, it is Tuesday, 14:00

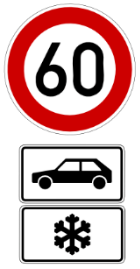
```
Path 2, offset 0: ExtendedSpeedLimitValue {
        value: 60,
        source: explicit,
        conditions {
                ConditionNumeric {
                        type: conditionTypeDaysOfWeek
                        appliesToEgoVehicle: true,
                        value: Tuesday | Thursday | Friday
                },
                ConditionTimeOfDay {
                        type: conditionTypeTimeOfDay
                        appliesToEgoVehicle: false,
                        startMinutes: 960,
                        endMinutes: 1080
                }
        }
}
```

2.4.3.4 Example 60 for next 800m



```
Path 3, offset 100:
ProfileEntry {
      …
      offset: 100,
      endOffset: 900,
      …
      type: ExtendedSpeedLimit
      …
      value: ExtendedSpeedLimitValue {
            value: 60,
            source: explicit,
            conditions {}
      }
}
```

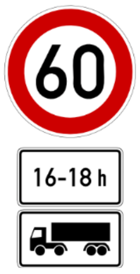
2.4.3.5 Example 60 km/h for cars when snow



situation: vehicle is car, no information regarding current weather

```
Path 3, offset 0: ExtendedSpeedLimitValue {
      value: 60,
      source: explicit,
      conditions {
            ConditionVehicleType {
                  type: conditionTypeVehicle,
                  appliesToEgoVehicle: true,
                  vehicleTypeMask: Trucks
            },
            ConditionWeather {
                  type: conditionTypeWeather,
                  appliesToEgoVehicle: unknown,
                  weather: Snow
            }
      }
}
```

2.4.3.6 *Example 60 km/h for Truck with Trailer and time restricted*



situation: vehicle is car, it is 16:30

```
Path 4, offset 500: ExtendedSpeedLimitValue {
      value: 60,
      source: explicit,
      conditions {
            ConditionVehicleType {
                  type: conditionTypeVehicle,
                  appliesToEgoVehicle: false,
                  vehicleTypeMask: Trucks
            },
            ConditionTimeOfDay {
                  type: conditionTypeTimeOfDay
                  appliesToEgoVehicle: true,
                  startMinutes: 960,
                  endMinutes: 1080
            }
      }}
```

2.4.4 Average Speed

This profile provides information on the speed actually driven according to historical statistics.

Property	Value
Profile Interpolation Type	Step
Profile Value	SpeedProfileValue

2.4.5 Flow Speed

This profile provides information on the speed currently driven according to live measurements.

Property	Value
Profile Interpolation Type	Step
Profile Value	SpeedProfileValue

2.5 Other Profiles

There is a lot of minor profiles describing various aspects of a path, often just a single Boolean flag. Some other profiles are slightly more complex but still do not fall into any group of related profiles that jointly describe certain aspects.

2.5.1 Probability

This profile provides an estimate for the probability, in percent, that the vehicle will reach a certain location in the ADASIS horizon.

Property	Value
Profile Interpolation Type	Step
Profile Value	FloatProfileValue contains Probability

2.5.2 Complex Intersection

This is a Boolean flag marking a part of a path that is inside an intersection.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.5.3 Link Identifier

This profile gives the ID of the map database link to which a specific part of the path belongs to.

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt64ProfileValue

2.5.4 Form of Way

This profile represents the type or form of a path; a path can be a tunnel, a bridge, a divided road, etc.

Property	Value
Profile Interpolation Type	Step
Profile Value	FormOfWayProfileValue

2.5.5 Road Accessibility

This profile represents the information about which classes of actors can access the path.

Property	Value
Profile Interpolation Type	Step
Profile Value	Int32ProfileValue bit field of RoadAccessFlags

2.5.6 Access Restriction

Driving restrictions for a path.

Property	Value
Profile Interpolation Type	Step
Profile Value	ConditionalRestrictionProfileValue

2.5.7 Overtaking Restriction

Overtaking restrictions for a path.

Property	Value
Profile Interpolation Type	Step
Profile Value	ConditionalRestrictionProfileValue

2.5.7.1 Example Overtaking Restriction for all vehicles

vehicle is car

```
Path 2, offset 0:
ProfileEntry {
      …
      offset: 0,
      endOffset: 0,
      …
      type: OvertakingRestriction
      …
      value: ConditionalRestrictionProfileValue {
            allowed: false
      }
}
```

2.5.7.2 Example Overtaking Restriction for Trucks

vehicle is truck

```
Path 2, offset 0:
ProfileEntry {
      …
      offset: 0,
      endOffset: 0,
      …
      type: OvertakingRestriction
      …
      value: ConditionalRestrictionProfileValue {
            allowed: false
            conditions {
                  ConditionVehicleType {
                        type: conditionTypeVehicle,
                        appliesToEgoVehicle: true,
                        vehicleTypeMask: Trucks
                  }
            }
      }
}
```

2.5.8 Tunnel

The road is in a tunnel.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.5.9 Bridge

The road is on a bridge.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.5.10 Divided Road

There is a division between the lanes of opposite driving directions.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.5.11 Functional Road Class

This is a classification of the road with respect to its importance for routing. Lower values correspond to higher importance. The actual range of numbers is dependent on the map provider, as is their exact definition. Please note that there is no strict correspondence between the functional class and physical properties of the road. (In some parts of the world, an important international through-route might not even be paved…)

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue

2.5.12 Route Number Types

This profile represents the assigned route number types to the path.

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue as bit field (for compatibility to ADASIS v2)

2.5.13 Built up Area

There are buildings present next to the road.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.5.14 In Town

The road is legally considered an in-city road.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.5.15 Surface

This profile describes the physical road surface.

Property	Value
Profile Interpolation Type	Step
Profile Value	SurfaceConditionProfileValue

2.5.16 Weather

This profile describes the weather conditions at some location in the ADASIS horizon.

Property	Value
Profile Interpolation Type	Step
Profile Value	WeatherProfileValue

2.5.17 Traffic Light

The Traffic Light profile describes traffic lights along a path. It can be sent for a whole path or for each lane individually. In the latter case it is a must to describe each lane. Traffic Light profile contains information about the position of the traffic light, light cycle time, traffic light current state and the “green arrow” information which is represented by “turn on red allowed” flag.

Property	Value
Profile Interpolation Type	Spot
Profile Value	TrafficLightProfileValue

2.5.18 Traffic Sign

The traffic sign profile conveys information about roadside traffic signs.

Traffic sign information is expected to be used both for general ADAS purposes (where the meaning of the sign is of importance) and to augment the camera-based traffic sign recognition (where the visual appearance of the sign is of importance).

For traffic signs, the offset given in the profile structure defines the point on the road where the sign is effective. Signs also have a physical location. Both locations can differ for various reasons (see examples in Figure 16 and Figure 17). The point where the sign is effective as well as validity length can be given explicitly as an additional panel to the traffic sign.

Property	Value
Profile Interpolation Type	Spot
Profile Value	TrafficSignValue

2.5.18.1 List of all traffic signs

Traffic Sign	Vienna Convention	MUCTD	Number	Remarks
Curvy Road	A,1		17	

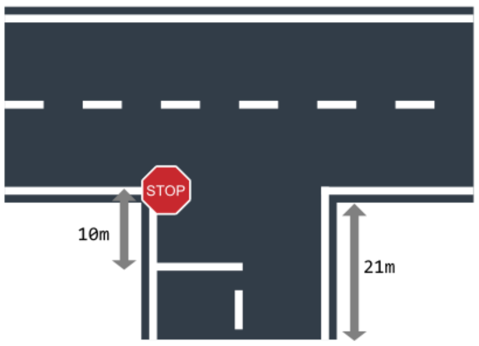
Left Bend	A,1a	W1-1L	10	
Right Bend	A,1b	W1-1R	9	
Double Bend Left First	A,1c	W1-4	14	
Double Bend Right First	A,1d	W1-4	13	
Steep Descent	A,2	W7-1b	68	If value is not 0, then it is the grade in percent
Steep Ascent	A,3	W7-3P	67	If value/ is not 0, then it is the grade in percent
Carriageway Narrows	A,4a		40	
Carriageway Narrows Right	A,4b		414141	
Carriageway Narrows Left	A,4b reversed		424242	
Swing Bridge	A,5		89	
River Bank	A,6		29	
River Bank Left	A,6 reversed		30	
Uneven Road	A,7a	W8-11	69	
Hump	A,7b	W8-1 W17-1	70	
Dip	A,7c	W8-21	71	
Slippery Road	A,9	W8-5	66	
Falling Rocks	A,11	W8-14	60	
Falling Rocks Right	A,11		62	
Falling Rocks Left	A,11 reversed		61	
Children	A,13		52	
School Zone	A,13		53	
Cyclists	A,14		54	
Domestic Animals	A,15a	W11-4	4	
Wild Animals	A,15b	W11-3	5	
Road Works	A,16	W21-1	6	
Light Signals	A,17	W3-3	31	Traffic Sign warning of light signals, not the signals themselves (see code number 254).
Dangerous Intersection	A,18		22	
Intersection	A,18		35	
Intersection with Priority to the Right	A,18a		37	
Intersection with Minor Road	A,19		36	
Two-Way Traffic	A,23	W6-3	55	
Traffic Congestion	A,24		75	
Railroad with Gates	A,25		56	
Railroad without Gates	A,26		57	
Tramway	A,27		59	
Danger	A,32		0	
Yield	B,1	R3-1	32	
Stop	B,2	R1-1	33	
Priority Road	B,3		34	
Priority for Oncoming Traffic	B,5		82	
Priority over Oncoming Traffic	B,6		81	
Overtaking Prohibited	C,13aa C,13ab	R4-1	46	

Overtaking by Goods Vehicles Prohibited	C,13ba C,13bb		20	Goods vehicles/trucks above 3.5 tons, unless stated otherwise on an additional panel.
Speed Limit	C,14	R2-1	87	Value encodes speed as in the SEGMENT message (see table 12).
End of All Prohibitions	C,17a		79	
End of Speed Limit	C,17b	R2-1 with R3-9dP	88	Value encodes speed as in the SEGMENT message (see table 12).
Overtaking Prohibited End	C,17d	R4-1 with R3-9dP	47	
Overtaking by Goods Vehicles Prohibited End	C,17d		21	
Direction to the Left	D,1 reversed		39	
Direction to the Right	D,1a		38	
Pass Right Side	D,2	R4-7	3	
Pass Left or Right Side	D,2 left+right	W12-1	1	
Pass Left Side	D,2 reversed	R4-8	2	
Beginning of Built-up Area	E,7		77	
End of Built-up Area	E,8		83	
Tunnel	E,11a		24	
Residential Area	E,17a		7	In European Supplement.
End of Residential Area	E,17b		8	In European Supplement.
Lane Merge Left	G,12a	W4-1L	434343	
Lane Merge Right	G,12a reversed	W4-1R	444444	
		? + R3-9dP	48	
Humpback Bridge		Bridge with W8-1	28	
		Chapter 2L	65	See also Value Code 80 for variable signs with light elements.
		Chapter 2L	80	See also Value Code 80 for variable signs with mechanical elements.
Narrow Bridge		W5-2	26	
Steep Drop Left		W8-17L	63	
Steep Drop Right		W8-17R	64	
Road Floods		W8-18	72	
Side winds		W8-21	74	Sign 117-10 in Germany
Icy Road		W8-5aP	73	
		W9-2	45	
			49	
High Accident			76	Sign A-34 in Poland, Supplementary Sign 1006-36 in Germany,.
			78	

Unknown/Reserved Code			255	Unknown traffic sign type.

2.5.18.2 *Examples*

To model all possible combinations the fields *shift*, *distance* and *length* can be filled in the traffic sign structure (see TrafficSignValue). The following examples illustrate how these values are used:



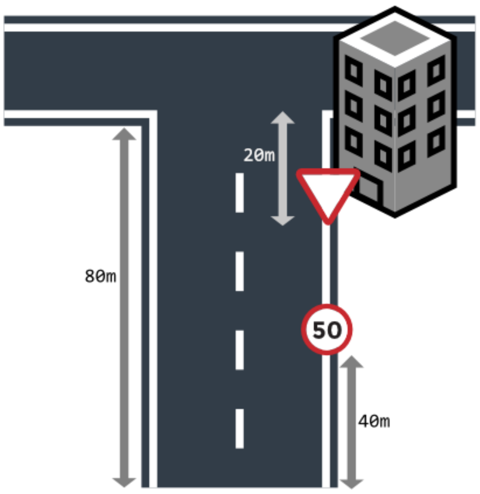


Figure 16 Stop sign with a positive shift, Right: Give way sign with negative shift

The stop sign in Figure 16 on the left is located at the intersection but vehicles have to stop at a line before. The physical location does not match the point where the sign is effective. The *offset* of the sign is 11m because it is effective at the line at offset 11m. The *shift* is 10m because the sign is located 10m behind the stop line. The physical location of the traffic sign can be calculated as *offset+shift* =21m. The *distance* and *length* values are both at their default (0m).

The yield sign in Figure 16 Stop sign with a positive shift, Right: Give way sign with negative shift on the right is located before the intersection for structural reasons, the building is close to the road and there is no room to position the sign exactly to the location where it is effective. The *offset* of the sign is 80m because it is effective after the building. The shift is -20m because the sign is located 20m before the offset. The physical location of the traffic sign can be calculated as *offset+shift* =60m. The*distance* and *length* values are both at their default (0m).

The speed limit sign in Figure 16 Stop sign with a positive shift, Right: Give way sign with negative shift on the right is located at the same point where it is effective. The *offset* of the sign is 40m. The *distance*, *length* and *shift* values are both at their default (0m).

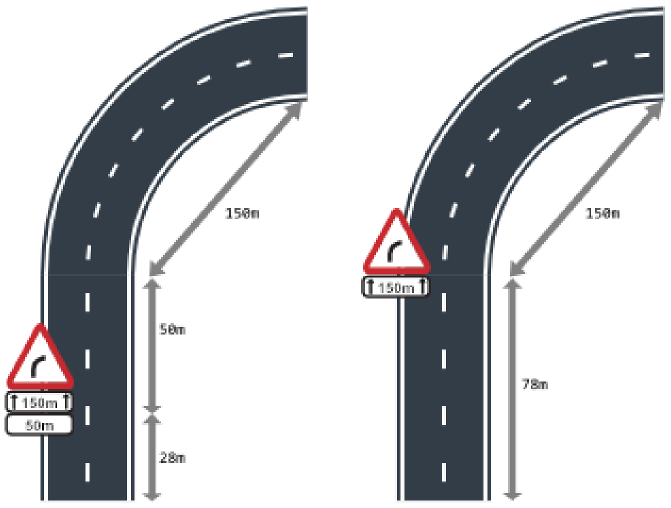


Figure 17 Left: Curve warning ahead sign with explicit validity length and shift, Right: Curve warning sign with explicit validity length

The curve warning ahead sign in Figure 17 on the left has 2 panels that explicitly state *distance* and *length*. The *offset* of the sign is 28m. The *distance* is 50m and *length* is 150m, *shift is 0*.

The curve warning sign in Figure 17 on the right is positioned directly at the beginning of the curve and has 1 panel that explicitly states the validity length of the warning sign. The *offset* of the sign is 78m. The *length* is 150m. The *shift* and *distance* values are both at default (0m).

2.5.19 Special Situation

This profile can flag several situations that each occurs comparatively rarely:

Property	**Value**
Profile Interpolation Type	Step
Profile Value	SpecialSituationProfileValue

2.5.20 Road Condition

The Road Condition profile describes the current condition of the road and the weather. In distinction to the Surface Profile Type this profile type defines the dynamic and changeable aspects of the road surface and the environment. It can be sent for a whole path.

Property	**Value**
Profile Interpolation Type	Step
Profile Value	RoadConditionProfileValue

2.5.21 Part of Calculated Route

This profile marks a part of a path that is a part of the route currently used by a navigation system.

Property	Value
Profile Interpolation Type	Step
Profile Value	BooleanProfileValue

2.6 Global Data Profiles

There is some data that does not usually depend on the current road but describes global properties of the system, the current state of the provider, or information that only very roughly depends on the location, typically because it is fixed per country. The profiles in this section have been created for providing such data.

Generally it is expected that they are transmitted (if supported at all) with low frequency in Global Data messages.

It still is possible to attach these profiles to a path, which is useful in particular for country dependent profiles when parts of the ADASIS horizon fall into a different country than the current vehicle position.

2.6.1 Country Code

The current ISO 3166-1 three-letter country code, as a three-byte array (without a trailing zero).

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue contains ISO 3166-1 (numeric)

2.6.2 Region Code

The current ISO 3166-2 three-letter subregion code, as a three-byte array (without a trailing zero).

Property	Value
Profile Interpolation Type	Step
Profile Value	RegionCodeValue

2.6.3 Driving Side

The legal driving side (right-handed or left-handed traffic).

Property	Value
Profile Interpolation Type	Step
Profile Value	DrivingSideProfileValue

2.6.4 Unit System

The system of measurement in use e.g. on speed-limit signs (metric or imperial).

Property	Value
Profile Interpolation Type	Step
Profile Value	UnitSystemProfileValue

2.6.5 Version Protocol

The version of the ADASIS protocol used by the provider.

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue
Encoding	$2^{24} \cdot major + 2^{16} \cdot minor + sub$

2.6.6 Version Hardware

A version number for the hardware of the system the provider is running on (implementation specific).

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue

2.6.7 Version Map

A version number for the map used at the current vehicle location (implementation specific).

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue

2.6.8 Map Age

Actually the release date of the map used at the current vehicle location, in days since 1970-01-01.

Property	Value
Profile Interpolation Type	Step
Profile Value	UInt32ProfileValue

2.6.9 Map Provider

An enumeration value identifying the provider of the map used at the current vehicle location.

Property	Value
Profile Interpolation Type	Step
Profile Value	MapProviderProfileValue

2.6.10 Map Status

Information about the availability of the map at the current vehicle location.

Property	Value
Profile Interpolation Type	Step
Profile Value	MapStatusProfileValue

2.6.11 System Status

This profile describes whether a navigation system is actively guiding, and additionally flags driving simulation.

Property	Value
Profile Interpolation Type	No Interpolation (global data only)
Profile Value	SystemStatusProfileValue

2.6.12 Time Zone Offset

This profile provides the current offset of the legal time from UTC in minutes. It is used e.g. to translate time restrictions on traffic signs into UTC time. Please note that the time used by the driver or shown on the dashboard clock might have a different UTC offset.

Property	Value
Profile Interpolation Type	Step
Profile Value	Int32ProfileValue

2.6.13 Absolute Vehicle Position

Current absolute position of the vehicle, independently from the map.

Property	Value
Profile Interpolation Type	No Interpolation (global data only)
Profile Value	AbsoluteVehiclePositionProfileValue

3 ADASIS v3 Franca IDL Reference

3.1 Prolog and FIDL versioning

The prolog defines the package and version of the IDL. The FIDL versioning uses the X (for major) and Y (for minor) numbers from ADASIS v3 Versioning.

```
package org.adasis.v3

typeCollection Messages {
    version {
        major 1
        minor 0
    }
```

3.2 Typedefs

List of typedefs.

3.2.1 Angle

description

```
typedef Angle is Float
```

Used by: NodeArm, Position

3.2.2 Distance

description

```
typedef Distance is UInt32
```

Used by: Position, TrafficSignValue

3.2.3 InstanceId
description

```
typedef InstanceId is UInt32
```

Used by: ProfileEntry

3.2.4 LaneIdx
description

```
typedef LaneIdx is UInt8
```

Used by: Position, ProfileEntry

3.2.5 LinearObjId
description

```
typedef LinearObjId is UInt32
```

Used by: LaneInfo, LinearObject, LineGeometry

3.2.6 Offset
description

```
typedef Offset is UInt32
```

Used by: OffsetFloatEntry, ProfileControl, Position, ProfileControl, ProfileEntry

3.2.7 PathId
Unique path identifier type, 0 for "no path", whatever this means in some given context.

```
typedef PathId is UInt32
```

Used by: LaneConnectivityPair, NodeArm, PathControl, PathControlMessage, Position, ProfileControl, ProfileEntry

3.2.8 Probability
A probability value in range [0,100].

```
typedef Probability is Float
```

Used by: NodeArm, Position, Probability

3.2.9 SignLocationMask
description

```
typedef SignLocationMask is UInt16
```

Used by: TrafficSignValue

3.2.10 Timestamp
description

```
typedef Timestamp is UInt64
```

Used by: PositionMessage, VehiclePosition

3.2.11 VehicleSpeed

description

```
typedef VehicleSpeed is Float
```

Used by: Position

3.3 Enumerations

List of enumerations.

3.3.1 Availability

description

```
enumeration Availability {
    NotAvailable
    Valid
}
```

Value	Description
`NotAvailable`	The item is not known (e.g. not in map).
`Valid`	Description

Used by: GlobalData, ProfileEntry

3.3.2 ChangeMode

description

```
enumeration ChangeMode {
    Create
    Update
    Delete
}
```

Value	Description
`Create`	Initial transmission, or unmodified retransmission.
`Update`	Modified value, compared to first transmission.
`Delete`	Entry is to be deleted.

Used by: ProfileEntry

3.3.3 ConditionType

description

```
enumeration ConditionType {
    conditionTypeUnknown
    conditionTypeVehicle
    conditionTypeLoad
    conditionTypeTimeOfDay
    conditionTypeDaysOfWeek
    conditionTypeWeight
    conditionTypeTurnDirection
    conditionTypeWeather
    conditionTypeFuzzyTime
}
```

Value	Description
conditionTypeUnknown	Description.
conditionTypeVehicle	Description
conditionTypeLoad	Description
conditionTypeTimeOfDay	Description
conditionTypeDaysOfWeek	ConditionNumeric, bitmask, bit 0: Sunday, ..., bit 6: Saturday
conditionTypeWeight	ConditionNumeric, in kg
conditionTypeTurnDirection	Description
conditionTypeWeather	Description
conditionTypeFuzzyTime	Description

Used by: Condition

3.3.4 CurveType

description

```
enumeration CurveType {
    NotPresent
    Polyline
    BezierSpline
}
```

Value	Description
NotPresent	Description.
Polyline	Description
BezierSpline	Description

Used by: Curve

3.3.5 DrivingSide

description

```
enumeration DrivingSide {
    RightHandDriving = 0
    LeftHandDriving = 1
}
```

Value	Description
RightHandDriving	Description.
LeftHandDriving	Description

Used by: DrivingSideProfileValue

3.3.6 EffectiveSpeedLimitType

description

```
enumeration EffectiveSpeedLimitType {
    Unknown
    Implicit
    ExplicitOnTrafficSign
    ExplicitNight
    ExplicitDay
    ExplicitTimeOrDay
    ExplicitRain
    ExplicitSnow
    ExplicitFog
}
```

Value	Description
Unknown	Description.
Implicit	Description
ExplicitOnTrafficSign	Description
ExplicitNight	Description
ExplicitDay	Description
ExplicitTimeOrDay	Description
ExplicitRain	Description
ExplicitSnow	Description
ExplicitFog	Description

Used by: EffectiveSpeedLimit

- name of a structure using the enumeration, linked to its reference chapter

3.3.7 FormOfWay

description

```
enumeration FormOfWay {
    Unknown = 0
    ControlledAccess = 1
    MultipleCarriageWay = 2
    SingleCarriageWay = 3
    RoundaboutCircle = 4
    SpecialTrafficFigure = 5
    ReservedA = 6
    ReservedB = 7
    ParallelRoad = 8
    RampOnControlledAccess = 9
    RampNotOnControlledAccess = 10
    FrontageRoad = 11
    CarPark = 12
    Service = 13
    PedestrianZone = 14
    NotAvailable = 15
}
```

Value	Description
Unknown	Description.
ControlledAccess	Description
MultipleCarriageWay	Description
SingleCarriageWay	Description
RoundaboutCircle	Description
SpecialTrafficFigure	Description
ReservedA	Description
ReservedB	Description
ParallelRoad	Description
RampOnControlledAccess	Description
RampNotOnControlledAccess	Description
FrontageRoad	Description
CarPark	Description
Service	Description
PedestrianZone	Description
NotAvailable	Description

Used by: FormOfWayProfileValue

3.3.8 FuzzyTime

description

```
enumeration FuzzyTime {
    Unknown
    Day
    Night
}
```

Value	Description
Unknown	Description.
Day	Description

Night	Description

Used by: ConditionFuzzyTime

3.3.9 GuidanceMode

description

```
enumeration GuidanceMode {
    guidanceInactive
    guidanceForUser
    guidanceAutomatic
}
```

Value	Description
guidanceInactive	Description.
guidanceForUser	Description
guidanceAutomatic	Description

Used by: SystemStatusProfileValue

3.3.10 LaneArrowMarking

description

```
enumeration LaneArrowMarking {
    None = 0
    Straight = 1
    SlightRight = 2
    Right = 4
    HardRight = 8
    UTurn = 16
    HardLeft = 32
    Left = 64
    SlightLeft = 128
    NA = -1
}
```

Value	Description
None	Description.
Straight	Description
SlightRight	Description
Right	Description
HardRight	Description
UTurn	Description
HardLeft	Description
Left	Description
SlightLeft	Description

NA	Description

Used by: ConditionTurnDirection

3.3.11 LaneTransition

description

```
enumeration LaneTransition {
    None
    Opening
    Closing
    Merging
    Splitting
}
```

Value	Description
None	Description.
Opening	Description
Closing	Description
Merging	Description
Splitting	Description

Used by: LaneInfo

3.3.12 LaneTypeFlags

description

```
enumeration LaneTypeFlags {
    Unknown = 0
    Normal = 1
    Exit = 2
    Entry = 4
    Auxiliary = 8
    Emergency = 16
    RestrictedForbidden = 32
    RestrictedUsable = 64
    HOV = 128
    Express = 256
    Reversible = 512
    Slow = 1024
    DrivableShoulder = 2048
    TurnOrSuicide = 4096
}
```

Value	Description
Unknown	Description.
Normal	Description
Exit	Description
Entry	Description

Auxiliary	Description
Emergency	Description
RestrictedForbidden	Description
RestrictedUsable	Description
HOV	Description
Express	Description
Reversible	Description
Slow	Description
DrivableShoulder	Description
TurnOrSuicide	Description

Used by: LaneInfo

3.3.13 LateralPosition

Values can be combined with logical OR to create a mask of possibly multiple locations.

```
enumeration LateralPosition {
    Unknown = 0
    Right = 1
    Left = 2
    Above = 4
    Surface = 8
}
```

Value	Description
Unknown	Description.
Right	Description
Left	Description
Above	Description
Surface	Description

Used by: TrafficLightProfileValue

3.3.14 LineMarking

description

```
enumeration LineMarking {
    Unknown
    None
    SolidLine
    DashedLine
    DoubleSolidLine
    DoubleDashedLine
    LeftSolidRightDashed
    RightSolidLeftDashed
    DashedBlocks
    ShadedArea
    PhysicalDivider
}
```

Value	Description
`Unknown`	Description
`None`	Also used for an unmarked lane boundary.
`SolidLine`	Description
`DashedLine`	Description
`DoubleSolidLine`	Description
`LeftSolidRightDashed`	in direction of driving
`RightSolidLeftDashed`	in direction of driving
`DashedBlocks`	Description
`ShadedArea`	Description
`PhysicalDivider`	Description

Used by: LinearObject

3.3.15 LineMarkingColour

description

```
enumeration LineMarkingColour {
    None
    Other
    White
    Yellow
    Orange
    Red
    Blue
}
```

Value	Description
`None`	Description.
`Other`	Description
`White`	Description
`Yellow`	Description
`Orange`	Description
`Blue`	Description

Used by: LinearObject

3.3.16 LinearObjectType

description

```
enumeration LinearObjectType {
    Centerline
    LaneMarking
    Guardrail
    Fence
    Kerb
    Wall
}
```

Value	Description
Centerline	Description.
LaneMarking	Description
Guardrail	Description
Fence	Description
Kerb	Description
Wall	Description

Used by: LinearObject

3.3.17 Load

description

```
enumeration Load {
    loadWaterPolluting
    loadExplosive
    loadOtherDangerous
    loadEmpty
    loadSpecial
    loadGasses
    loadFlammableLiquids
    loadFlammableSolids
    loadOxidizing
    loadToxicInfectious
    loadRadioactive
    loadCorrosive
}
```

Value	Description
loadWaterPolluting	Description.
loadExplosive	Description
loadOtherDangerous	Description
loadEmpty	Description
loadSpecial	Description
loadGasses	Description
loadFlammableLiquids	Description
loadFlammableSolids	Description

loadOxidizing	Description
loadToxicInfectious	Description
loadRadioactive	Description
loadCorrosive	Description

Used by: ConditionLoad

3.3.18 LocationObjectType

description

```
enumeration LocationObjectType {
    GuidePost
}
```

Value	Description
GuidePost	Description.

Used by: LocationObject

3.3.19 MapProvider

The provider of the current map data. The enum value is prefixed with `provider_` and followed by a company name.

```
enumeration MapProvider {
    provider_Unknown
    provider_AND
    provider_AutoNavi
    provider_HERE
    provider_Hyundai
    provider_Navinfo
    provider_TomTom
    provider_Zenrin
}
```

Used by: MapProviderProfileValue

3.3.20 MapStatus

description

```
enumeration MapStatus {
    mapNotAvailable
    mapLoading
    mapAvailable
}
```

Value	Description
mapNotAvailable	Description.
mapLoading	Description
mapAvailable	Description

Used by: MapStatusProfileValue

3.3.21 MessageType

description

```
enumeration MessageType {
    Position
    Profile
    GlobalData
    ProfileControl
    PathControl
}
```

Value	Description
Position	Type is PositionMessage
Profile	Type is ProfileMessage
GlobalData	Type is GlobalDataMessage
ProfileControl	Type is ProfileControlMessage
PathControl	Type is PathControlMessage

Used by: MessageOnBus

3.3.22 ProfileType

Defines the data-type of a profile or global data. Each type is documented in the Profiles Reference.

```
enumeration ProfileType {
    Node
    Probability
    HeadingChange
    LaneModel
    LaneConnectivity
    LinearObjects
    LanesGeometry
    LaneWidth
    RoadGeometry
    NumberOfLanesPerDirection
    ComplexIntersection
    LinkIdentifier
    FunctionalRoadClass
    RouteNumberTypes
    FormOfWay
    RoadAccessibility
    AccessRestriction
    OvertakingRestriction
    Tunnel
    Bridge
    DividedRoad
    Curvature
    Slope
    BuiltUpArea
    InTown
    Surface
    TrafficSign
    TrafficLight
    SpecialSituation
    EffectiveSpeedLimit
    ExtendedSpeedLimit
    AverageSpeed
    FlowSpeed
    RoadCondition
    Weather
    LocationObject
    PartOfCalculatedRoute
    CountryCode
    RegionCode
    DrivingSide
    UnitSystem
    VersionProtocol
    VersionHardware
    VersionMap
    MapAge
    MapProvider
    MapStatus
    SystemStatus
    TimeZoneOffset
    AbsoluteVehiclePosition
}
```

Used by: GlobalData, ProfileEntry

3.3.23 Quality

description

```
enumeration Quality {
    Unknown
    NotAvailable
    Value1
    Value2
    Value3
    Value4
    Value5
}
```

Value	Description
Unknown	Description
NotAvailable	Description
Value1	Best
Value2	Something between
Value3	
Value4	
Value5	worst

Used by: SurfaceConditionProfileValue

3.3.24 RelativeDirection

description

```
enumeration RelativeDirection {
    None
    Both
    AlongPathDirection
    AgainstPathDirection
}
```

Value	Description
None	Description
Both	Description
AlongPathDirection	In the direction of increasing offset.
AgainstPathDirection	In the direction of decreasing offset.

Used by: LaneInfo, ProfileEntry

3.3.25 RightOfWay

description

```
enumeration RightOfWay {
    Unknown
    MustYield
    HasRightOfWay
}
```

Value	Description
Unknown	Description.
MustYield	Description
HasRightOfWay	Description

Used by: NodeArm

3.3.26 RoadAccessFlags

Values to use in the bit field for Accessibility profiles.

```
enumeration RoadAccessFlags {
    PassengerCars = 1
    Pedestrians = 2
    Bus = 4
    Delivery = 8
    Emergency = 16
    Taxi = 32
    ThroughTraffic = 64
    Trucks = 128
}
```

Value	Description
PassengerCars	Description.
Pedestrians	Description
Bus	Description
Delivery	Description
Emergency	Description
Taxi	Description
ThroughTraffic	Description
Trucks	Description

Used by: ConditionVehicleType, Road Accessibility

3.3.27 RoadCondition

description

```
enumeration RoadCondition {
    Unknown
    Clear
    Wet
    Snowy
    Icy
    Slippery
    Dirt
}
```

Value	Description
Unknown	Description.
Clear	Description
Wet	Description
Snowy	Description
Icy	Description
Slippery	Description
Dirt	Description

Used by: RoadConditionProfileValue

3.3.28 SignType

See Traffic Sign profile description for documentation of each supported sign.

```
enumeration SignType {
    Unknown = 255
    LeftBend = 10
    RightBend = 9
    DoubleBendLeftFirst = 14
    DoubleBendRightFirst = 13
    CurvyRoad = 17
    SteepDescent = 68
    SteepAscent = 67
    CarriageWayNarrows = 40
    CarriageWayNarrowsLeft = 42
    CarriageWayNarrowsRight = 41
    SwingBridge = 89
    RiverBank = 29
    RiverBankLeft = 30
    UnevenRoad = 69
    Hump = 70
    Dip = 71
    SlipperyRoad = 66
    FallingRocksRight = 62
    FallingRocksLeft = 61
    Pedestrians = 50
    PedestrianCrossing = 51
    Children = 52
    SchoolZone = 53
    Cyclists = 54
    DomesticAnimalsCrossing =4
    WildAnimalsCrossing = 5
    RoadWorks = 6
    LightSignals = 31
    DangerousIntersection = 22
    Intersection = 35
    IntersectionWithPriorityToTheRight = 37
    IntersectionWithMinorRoad =36
    TwoWayTraffic = 55
    TafficCongestion = 75
    RailwayCrossingWithGates =56
    RailwayCrossingWithoutGates = 57
    TramWay = 59
    RailwayCrossing = 58
    Danger = 0
    GiveWay = 32
    Stop = 33
    PriorityRoad = 34
    PriorityForOncomingTraffic = 82
    PriorityOverOncomingTraffic = 81
    OvertakingProhibited = 46
    OvertakingByGoodsVehiclesProhibited = 20
    SpeedLimit = 87
    EndOfAllProhibitions = 79
    EndOfSpeedLimit = 88
    EndOfProhibitionOnOvertaking = 47
    EndOfProhibitionOnOvertakingForGoodsVehicles = 21
    DirectionToTheRight = 38
    DirectionToTheLeft = 39
    PassRightSide = 3
    PassLeftOrRightSide = 1
    PassLeftSide = 2
    BeginningOfBuiltUpArea =77
    EndOfBuiltUpArea = 83
    Tunnel = 24
    ResidentialArea =7
    EndOfResidentialArea = 8
    LaneMergeRight = 44
    LaneMergeLeft = 43
    ProtectedPassingEnd = 48
    HumpbackBridge = 28
    VariableSignMechanicElements = 65
    VariableSignLightElements = 80
    FerryTerminal = 25
    NarrowBridge = 26
    IcyRoad = 73
    SteepDropLeft = 63
    SteepDropRight = 64
    RoadFloods = 72
    SideWinds = 74
    LaneMergeCenter = 45
    HighAccidentArea = 76
    AudibleWarning = 78
}
```

Used by: TrafficSignValue

3.3.29 SpecialSituationType

description

```
enumeration SpecialSituationType {
    DeadEnd = 248
    FerryTerminal = 249
    TollBooth = 250
    RailroadCrossing = 251
    PedestrianCrossing = 252
    SpeedBump = 253
}
```

Value	Description
DeadEnd	Path will never be extended past this point.
FerryTerminal	Description
TollBooth	Description
RailroadCrossing	Description
PedestrianCrossing	Description
SpeedBump	Description

Used by: SpecialSituationProfileValue

3.3.30 SpeedLimitSource

description

```
enumeration SpeedLimitSource {
    Unknown
    Implicit
    Explicit
}
```

Value	Description
Unknown	Description.
Implicit	Description
Explicit	Description

Used by: ExtendedSpeedLimitValue

3.3.31 SurfaceCondition

description

```
enumeration SurfaceCondition {
    Unknown
    PavedRigid
    PavedFlexible
    Blocks
    Gravel
    Dirt
}
```

Value	Description
Unknown	Description.
PavedRigid	Description
PavedFlexible	Description
Blocks	Description
Gravel	Description
Dirt	Description

Used by: SurfaceConditionProfileValue

3.3.32 TrafficLightLongitudinalPosition

description

```
enumeration TrafficLightLongitudinalPosition {
    Unknown
    BeforeIntersection
    AfterIntersection
}
```

Value	Description
Unknown	Description.
BeforeIntersection	Description
AfterIntersection	Description

Used by: TrafficLightProfileValue

3.3.33 TrafficLightState

description

```
enumeration TrafficLightState {
    Unknown
    Invalid
    Off
    Green
    Yellow
    Red
}
```

Value	Description
Unknown	Description.
Invalid	Description
Off	Description
Green	Description
Yellow	Description
Red	Description

Used by: TrafficLightProfileValue

3.3.34 UnitOfSpeed

description

```
enumeration UnitOfSpeed
{
    KpH
    MpH
}
```

Value	Description
KpH	Kilometers per Hour
MpH	Miles per Hour

Used by: Speed, UnitSystemProfileValue

3.3.35 Weather

description

```
enumeration Weather {
    Unknown
    NoSpecial
    Sunshine
    Rain
    Fog
    Snow
    Ice
    StrongWind
}
```

Value	Description
Unknown	Description

NoSpecial	None of those below.
Sunshine	Description
Rain	Description
Fog	Description
Snow	Description
Ice	Description
StrongWind	Description

Used by: ConditionWeather, WeatherProfileValue

3.3.36 YesNoUnknown

description

```
enumeration YesNoUnknown {
    Unknown
    Yes
    No
}
```

Value	Description
Unknown	Not known for the current position.
Yes	Description
No	Description

Used by: Condition, TrafficLightProfileValue, TrafficSignValue, YesNoUnknownProfileValue

3.4 Structures

List of structures.

3.4.1 General Data Types

3.4.1.1 Vector

Usually used for vehicle relative coordinates. In this case: All values in meters, x forward, y to the right, z up.

```
struct Vector {
    Float x
    Float y
    Float z
}
```

Name	Value	Description
x	Float, meters	forward
y	Float, meters	to the right
z	Float, meters	Up

Used by: TrafficLightProfileValue

3.4.1.2 Speed

Description of the structure.

```
struct Speed {
    UInt8 value
    UnitOfSpeed unit
}
```

Name	Value	Description
value	UInt8	Speed value, 0: unknown and 255: unlimited
unit	UnitOfSpeed	Unit of the speed value.

Used by: EffectiveSpeedLimit, ExtendedSpeedLimitValue, SpeedProfileValue

3.4.1.3 WGS84Point

Description of the structure.

```
struct WGS84Point {
    Int32 latitude
    Int32 longitude
    Int32 altitude
}
```

Name	Value	Description
latitude	Int32	East positive, 2^32 is 360 degrees.
longitude	Int32	North positive, 2^32 is 360 degrees.
altitude	Int32	cm above the reference ellipsoid; -1000000 if not available.

Used by: Curve, LocationObject, VehiclePosition

3.4.1.4 VehiclePosition

Description of the structure.

```
struct VehiclePosition extends WGS84Point {
    Timestamp timestamp
    Float heading
}
```

Name	Value	Description
timestamp	Timestamp	from the Vehicle Time Master
heading	Float	Description

Used by: AbsoluteVehiclePositionProfileValue

3.4.1.5 Position

Description of the structure.

```
struct Position {
    PathId pathId
    Offset offset
    Distance accuracy
    Int32 deviation
    VehicleSpeed speed
    Angle relativeHeading
    Probability probability
    LaneIdx currentLane
    PathId preferredPath
}
```

Name	Value	Description
`pathId`	PathId	Path where the position is located, 0 if off road
`offset`	Offset	Distance from the start of the path
`Accuracy`	Distance	Standard deviation of the `offset`
`Deviation`	Int32 in cm	cm distance between map matched position and estimated position. The exact semantic is implementation specific, for example: • Shortest (straight) distance between map-matched position and estimated position • Perpendicular distance to the extension of the road.
`Speed`	VehicleSpeed	Projected onto the road line.
`relativeHeading`	Angle	Angle from the heading of the road reference line at the given position to the actual heading of the vehicle (clockwise positive), i.e. vehicle heading minus road heading
`Probability`	Probability	Estimated probability that the vehicle is at this position. Note: This is not related to the accuracy of the position but to choosing between multiple candidate positions. Since not all candidate positions that have been computed internally need to be provided in the Position message, the probabilities in a Position message do not need to sum up to 100%; in particular with a Position message containing only a single position, the probability of this position can be less than 100% and then is a measure of how sure the positioning subsystem is about the position.
`currentLane`	LaneIdx	0 if unknown, or off-road.
`preferredPath`	PathId	Tip of the tree where we're expected to arrive. (Will be descendant of `pathId`.)

Used by: PositionMessage

3.4.2 Messages

3.4.2.1 AdasisMessageBase

Description of the structure.

```
struct AdasisMessageBase polymorphic {
}
```

Used by: GlobalDataMessage, PathControlMessage, PositionMessage, ProfileControlMessage, ProfileMessage

3.4.2.2 PositionMessage

The Position Message positions the vehicle on the path network using the path/offset model. Successive Position Messages can describe a movement of the vehicle.

With each message it is possible to send multiple positions with different probabilities to represent the current state of a positioning system. If the vehicle cannot be located on the tree, it is possible to signal this as an off-road or off-map situation. No valid position is indicated by empty position array in the message.

The absolute position of the vehicle is not included in the Position Message, it is provided by AbsoluteVehiclePositionProfileValue.

```
struct PositionMessage extends AdasisMessageBase {
    Timestamp timestamp
    Timestamp positionAge
    Position [ ] positions
}
```

Name	**Value**	**Description**
`timestamp`	Timestamp	Global timestamp from the Vehicle Time Master. It describes the time of the receiving of the GPS position by the positioning system (in the system time of a possible vehicle time master).
`positionAge`	Timestamp	A relative timestamp that describes the timespan between receiving a GPS position and sending the Position Message to the vehicle bus. It can be used regardless whether a vehicle time master is available.
`positions`	Position	Empty if positioning not available or when off-map.

Used by: None

3.4.2.3 ProfileValue

Description of the structure.

```
struct ProfileValue polymorphic {
}
```

Name	**Value**	**Description**
`field1`	Link to type	Description.
`field2`	Link to type	Description.

Used by: all profile type structures

3.4.2.4 ProfileEntry

Description of the structure.

```
struct ProfileEntry {
    InstanceId instanceId
    Boolean isRetransmission
    ChangeMode change
    Float confidence
    PathId pathId
    LaneIdx [ ] laneNumbers
    RelativeDirection direction
    Offset offset
    Offset endOffset
    Boolean endOffsetFinal
    ProfileType type
    Availability available
    ProfileValue value
}
```

Name	Value	Description
`instanceId`	InstanceId	0 if not used
`isRetransmission`	bool	Needed?
`Change`	ChangeMode	Description
`Confidence`	Float	[0%..100%]
`pathId`	PathId	Description
`laneNumbers`	LaneIdx[]	optional
`direction`	RelativeDirection	optional, absent = "both"
`offset`	Offset	Description
`endOffset`	Offset	The endOffset is pptional. Up to here (at least) the profile values are valid, 0 == no known limit.
`endOffsetFinal`	bool	The endOffsetFinal is optional. End offset will not be updated anymore.
`type`	ProfileType	Description
`available`	Availability	Description
`value`	ProfileValue	Description

Used by: ProfileMessage

3.4.2.4.1 Validity Length

The validity length of the value is described with `offset`, `endOffset` and `endOffsetFinal`. For a validity length check and detecting gaps in profiles (resulting from map failure or message transmitting issues) all three are needed. Depending on the interpolation type, the validity length variables have different meanings which are explained in detail in the next table.

Interpolation type	Meaning of offset	Meaning of endOffset*	Meaning of	Meaning of endOffset = 0	Overlapping

		(gap detection)	**endOffset = offset**		
spot	point where a profile applies example: traffic sign “stop sign”	there will not be any other entry of such profile up to this offset	should not happen***	endOffset is unknown, there is no information at which offset the next profile entry will follow	**allowed** different spot profiles can have same offset and different endOffsets
step	start offset for constant value	constant value is valid at least this offset range	should not happen***	endOffset is unknown, there is no information at which offset the next profile entry will follow	**not allowed**
linear	start point for linear interpolation to next profile entry; end point for linear interpolation from preceding profile entry	there will not be another value for interpolation before this offset****	offset of point before jump (discontinuit y); such a profile entry is end point for linear interpolation	endOffset is unknown, there is no information at which offset next profile entry will follow until it is sent	**not allowed** exception: discontinuity
special	start offset for the part of the path described by the profile entry	end of the part of the path described by the profile entry	should not happen***	endOffset is unknown, there is no information at which offset the next profile entry will follow	profile type specific

* endOffset is not fix and can always increase as long as finalOffset flag is not set:

- It is not mandatory to know the real end of each attributes validity at the moment the first profile entry is created.
- In this case it is just known that there will not be a different value for the profile up to some location; then endOffset is set to this location (i.e. basically the validity length of the profile entry preliminary extends to the current end of the horizon).
- Value can be updated at any time.
- It is expected that just updating of endOffset decrease memory consumption on reconstructor side compared to sending several profile entries.

** endOffsetFinal is mandatory in case of endOffset is used:

- indicates, that endOffset is now final and won't increase
- provider knows end of profile entryentry validity; it must follow a new profile message with subsequent (start) offset

*** either endOffset is unknown (use "0") or max. horizon length

**** no indication of interpolation itself, cannot be used for extrapolation

3.4.2.4.2 Completeness check

- endOffset of each profile entry has the same value as offset of direct successor profile entry of the same type
- endOffset itself is exclusive; validity length of a profile entry X is defined by following interval **[offset$_X$, endOffset$_X$)**
- Profile message must be handled as a whole when multiple profile entries are in the message

3.4.2.4.3 Road and Lane Specific Profile Entries

A profile entry can represent the characteristics for the complete road; this profile entry describes a road level attribute, like the functional road class, form of way, or the number of lanes. A profile entry can also represent the characteristics of a specific lane, or of a set of lanes; this profile entry describes a lane level attribute, like the lane type, lane geometry, or the allowed driving direction per lane. It is possible to use profile entries of the same profile type to represent road level or lane level attributes.

In the Profile Entry structure the **laneNumber** array is used to distinguish the usage of the profile: If the array is empty, the profile represents characteristics on the road level. If one or more lane numbers are specified in the laneNumber array, the profile represents characteristics on the lane level for the specified lane or set of lanes.

- It is required to specify the lane numbers in ascending order to simplify the processing and error checking on the receiver side.
- It is an error to give the same lane number multiple times in the laneNumber array of the same profile entry.
- Depending on the profile type, it is allowed, it shall be avoided, or it is an error if more than a single profile entry exists on the same path, offset, and lane.

It is possible that an attribute exists on the road level and that this attribute exists also on the lane level for all or a subset of lanes.

3.4.2.4.4 Time dependency

Since ADASIS v2 it is discussed how time dependent attributes should be handled. Taking time into consideration could cause frequent updates of a profile because of changing prediction.

For example, a provider should provide a typical profile: speed limit information. Also typical is that speed limits are sometimes time dependent. As an example: 100 km/h from 8:00 – 20:00 and 60 km/h from 20:00 – 6:00 (because of noise). When sending out the profiles – it is now a couple of minutes before 20:00 –, which of the both values should be provided?

- **Recommendation**
  - For time dependent attributes it is recommended to provide that attribute value which is valid for the time point of estimated vehicle arrival.
  - To avoid frequent updates a well-considered algorithm with an inert estimation update is recommended.
    Perhaps it might be helpful to apply a more conservative calculation. For example, when calculating energy consumption of an electrical vehicle in marginal situations the most conservative time estimation is that one which has bigger influence on consumption.

3.4.2.4.5 Conditions

Condition represents a rule that governs the applicability of a profile entry. For a profile entry, all conditions attached to it shall be combined in a logical AND to make the profile entry applicable or not. For traffic signs, a condition also may represent a panel or supplementary sign. See ConditionType for a list of condition types which are supported.

3.4.2.5 ProfileMessage

The profile is a representation of specific characteristic of a path. Examples of these characteristics are: curvature of the path, the speed limit on this path, number of lanes …etc. The profile message contains an array of profile entries. For completeness checking, a profile message must be handled as a whole.

Each profile entry has a specific range on which it is valid depending on its Profile Interpolation Type (see [1]).

```
struct ProfileMessage extends AdasisMessageBase {
    ProfileEntry [ ] profiles
}
```

Name	Value	Description
profiles	ProfileEntry []	Description.

Used by: None

3.4.2.6 GlobalData

Description of the structure.

```
struct GlobalData {
    ProfileType type
    Availability available
    ProfileValue value
}
```

Name	Value	Description
type	ProfileType	Description.
available	Availability	Description.
value	ProfileValue	value

Used by: GlobalDataMessage

3.4.2.7 GlobalDataMessage

This message type is used to transmit profile entries containing global data if they are not related to a specific path. Examples of such profile entries are AbsoluteVehiclePositionProfileValue and SystemStatusProfileValue. Global data could be information like driving side or unit system which are valid everywhere in a specific region and do not change almost always as long as the vehicle is in one country.

On the one hand, it is not necessarily the case that this message contains profiles which are solely global. But it is used when profile entries shall be transmitted without any path relationship. On the other hand, Global Data Profiles are not restricted to be sent by Global Data Message. For example, the unit system could change at a country's border. Then it would make sense to additionally provide this profile with a Profile Message and specific offset to indicate the location of change. The content of the Global Data Message – suppose it is sent cyclic – would change at the time point the vehicle crosses the border.

```
struct GlobalDataMessage extends AdasisMessageBase {
    GlobalData [ ] data
}
```

Name	Value	Description
`data`	GlobalData[]	Description.

Used by: None

3.4.2.8 ProfileControl

See ProfileControlMessage for more details.

```
struct ProfileControl {
    PathId pathId
    Offset offset
}
```

Name	Value	Description
`pathId`	PathId	An existing path where profile data is erased.
`offset`	Offset	Smallest managed offset of the provider for the specific path. • A value of zero means that all information must be kept for the corresponding path. • The offset is an increasing only value

Used by: ProfileControlMessage

3.4.2.9 ProfileControlMessage

The `ProfileControlMessage` is mandatory.

Profile Control Message contains information for profile store management. Therefore it contains all relevant instructions for managing which profile sentries (attributes) must be stored or can be deleted.

In comparison to Path Control mechanism, an ADASIS v3 Horizon Provides uses Profile Control Messages to declare the existence of profile entries in the data storage of a receiver.

While a Profile Message (ProfileMessage) adds a profile entry to a path, Profile Control Message indicates which profile entries *can* be deleted from paths.

A Profile Control Message contains a list of ProfileControl entries; each refers to a path that should exist in the ADASIS horizon. Not all existing paths must be in the list. Only paths, where provider needs to update the `offset`, need to be listed.

Paths unreferenced by a *Profile Control Message* must be kept unchanged.

```
struct ProfileControlMessage extends AdasisMessageBase {
    ProfileControl [ ] values
}
```

Name	Value	Description
`values`	ProfileControl[]	Array of all profile control values. Empty array means: Data can be kept without changes. Can be empty in case no `offset` should be updated.

Used by: None

3.4.2.10 PathControl

Basic information for a single existing path, see PathControlMessage for more details.

```
struct PathControl {
    PathId Id
    PathId parentId
    Offset offset
}
```

Name	Value	Description
`Id`	PathId	The ID of the active path. The one that should exist.
`parentId`	PathId	The ID of the parent path. (0 if it has no parent)
`offset`	Offset	Offset of the branching point on the parent path.

Used by: PathControlMessage

3.4.2.11 PathControlMessage

The `PathControlMessage` is mandatory.

It contains information for path management, including all relevant instructions for creating, deleting or changing any path relationship in the horizon tree.

An ADASIS v3 Horizon provider uses Path Control Messages to declare which paths exist and shall be kept in the data storage of a receiver. If a path is listed in a Path Control Message, it is created on the receiver side (if it does not exist already); if a path is missing in a Path Control Message, it is deleted on the receiver side (if it has been existing there).

Further information about the ADASIS v3 horizon tree (or forest), for example turn angles or turn probabilities, is provided using the Node profile (see below) which contains data about the arms of a node (i.e. intersection) and can relate these arms to side paths in the tree using their path IDs.

A straightforward implementation would require each Path Control Message to list all paths that should currently be in existence. Obviously, this requires a size of this message in proportion to the number of existing paths. But in practical implementations the size of a message is limited, while the number of simultaneously existing paths should not be limited by such a technical property of the underlying protocol. To accommodate this, the Path Control Message contains the `idFirst` and `idLast` fields – a Path Control Message is authoritative for the range of path IDs from `idFirst` to `idLast`, inclusive, implying that a receiver should locally delete paths in this range that are not explicitly listed as existing in this Path Control Message.

If a provider does not split up path control over multiple messages, then it should set `idFirst` to 0 and `idLast` to the maximum unsigned 32-bit value for the single Path Control Message that it sends. If a provider splits up path control over multiple messages, then the ID ranges from the Path Control Messages should cover the full range from 0 to the maximum unsigned 32-bit value without gaps or overlaps, and the messages should be ordered by the ID ranges.

A receiver should not rely on this, though. It should handle every Path Control Message individually, deleting paths in the corresponding ID range that are not listed in the Path Control Message, and creating paths that are listed in the Path Control Message.

```
struct PathControlMessage extends AdasisMessageBase {
    PathId idFirst
    PathId idLast
    PathControl [ ] values
}
```

Name	**Value**	**Description**
`idFirst`	PathId	Lowest path ID handled by this message. 0 for first message of a set.
`idLast`	PathId	Highest pathID handled by this message. Maximum value for last message of a set.
`values`	PathControl[]	Array of all paths currently managed from the provider. Empty array means to invalidate/erase all data.

Used by: None

3.4.2.12 MessageOnBus

Description of the structure.

```
struct MessageOnBus {
    UInt8 cyclicCounter
    MessageType type
    AdasisMessageBase message
}
```

Name	**Value**	**Description**

cyclicCounter	UInt8	Description.
type	MessageType	Description.
message	AdasisMessageBase	

Used by: None

3.4.3 Standard Profile Types

3.4.3.1 *UInt32ProfileValue*

Description of the structure.

```
struct UInt32ProfileValue extends ProfileValue {
    UInt32 value
}
```

Name	Value	Description
value	UInt32	Description.

Used by: Number of Lanes per Direction, Functional Road Class, Route Number Types, Country Code, Version Protocol, Version Hardware, Version Map, Map Age

3.4.3.2 *Int32ProfileValue*

Description of the structure.

```
struct Int32ProfileValue extends ProfileValue {
    Int32 value
}
```

Name	Value	Description
value	Int32	Description.

Used by: Road Accessibility, Time Zone Offset

3.4.3.3 *UInt64ProfileValue*

Description of the structure.

```
struct UInt64ProfileValue extends ProfileValue {
    UInt64 value
}
```

Name	Value	Description
value	UInt64	Description.

Used by: Link Identifier

3.4.3.4 *FloatProfileValue*

Description of the structure.

```
struct FloatProfileValue extends ProfileValue {
    Float value
}
```

Name	Value	Description
value	Float	Description.

Used by: Heading Change, Probability

3.4.3.5 BooleanProfileValue

Description of the structure.

```
struct BooleanProfileValue extends ProfileValue {
    Boolean value
}
```

Name	Value	Description
value	Boolean	Description.

Used by Tunnel, Bridge, Divided Road, Built up Area, In Town

3.4.3.6 YesNoUnknownProfileValue

For use in custom extensions

```
struct YesNoUnknownProfileValue extends ProfileValue {
    YesNoUnknown value
}
```

Name	Value	Description
value	YesNoUnknown	Description.

Used by:

3.4.4 Enumerated Profile Types

3.4.4.1 FormOfWayProfileValue

Description of the structure.

```
struct FormOfWayProfileValue extends ProfileValue {
    FormOfWay value
}
```

Name	Value	Description
value	FormOfWay	Description.

Used by: Form of Way

3.4.4.2 DrivingSideProfileValue

Description of the structure.

```
struct DrivingSideProfileValue extends ProfileValue {
    DrivingSide value
}
```

Name	Value	Description

value	DrivingSide	Description.

Used by: Driving Side

3.4.4.3 UnitSystemProfileValue

Description of the structure.

```
struct UnitSystemProfileValue extends ProfileValue {
    UnitOfSpeed value
}
```

Name	Value	Description
value	UnitOfSpeed	Description.

Used by: Unit System

3.4.4.4 SpecialSituationProfileValue

Description of the structure.

```
struct SpecialSituationProfileValue extends ProfileValue {
    SpecialSituationType value
}
```

Name	Value	Description
value	SpecialSituationType	Description.

Used by:

3.4.4.5 RoadConditionProfileValue

Description of the structure.

```
struct RoadConditionProfileValue extends ProfileValue {
    RoadCondition value
}
```

Name	Value	Description
value	RoadCondition	Description.

Used by:

3.4.4.6 WeatherProfileValue

Description of the structure.

```
struct WeatherProfileValue extends ProfileValue {
    Weather value
}
```

Name	Value	Description
value	Weather	Description.

Used by: Weather

3.4.4.7 MapProviderProfileValue

Description of the structure.

```
struct MapProviderProfileValue extends ProfileValue {
    MapProvider value
}
```

Name	Value	Description
value	MapProvider	Description.

Used by: Map Provider

3.4.4.8 MapStatusProfileValue

Description of the structure.

```
struct MapStatusProfileValue extends ProfileValue {
    MapStatus value
}
```

Name	Value	Description
value	MapStatus	Description.

Used by: Map Status

3.4.5 Structured Profile Values

3.4.5.1 OffsetFloatEntry

Description of the structure.

```
struct OffsetFloatEntry
{
    Offset offset
    Float value
}
```

Name	Value	Description
offset	Offset	Absolute offset from start of path.
value	Float	Description.

Used by: OffsetFloatProfileValue

3.4.5.2 OffsetFloatProfileValue

Description of the structure.

```
struct OffsetFloatProfileValue extends ProfileValue
{
    OffsetFloatEntry[] entries
}
```

Name	Value	Description
entries	OffsetFloatEntry []	Description.

Used by: Curvature, Slope

3.4.5.3 NodeArm

A node arm describes a side-road (sub-path) branching at a certain offset on a path (parent path).

The side path does not need to exist (i.e. to have been created by listing it in a Path Control Message). If a side path is created later (i.e. after sending the Node profile entry), the side path has to use the path ID listed as subPath in the NodeArm. It is recommended that a provider assign valid subPath values to each NodeArm just for the case that a side path is created at a later point in time; excepted are those node arms where it is known in advance that no side path ever will be created (e.g. for one-way roads seen the wrong way; but this depends on the implementation and configuration of the provider).

```
struct NodeArm {
    PathId subPath
    Probability probability
    Angle turnAngle
    Boolean isComplexIntersection
    RightOfWay rightOfWay
}
```

Name	Value	Description
subPath	PathId	The ID of a side path starting with this road segment. If no ID for a possible side path is provided, then subPath is 0. For the NodeArm corresponding to the continuation of the main path, the subPath value is the ID of the main path.
probability	Probability	Relative probability at this node in percent, 0 if impossible / illegal to enter. Describes the likelihood to turn into this sub-path, assuming that the location is reached via parent path.
turnAngle	Angle	The change of vehicle heading to turn into this node arm.
isComplexInter section	Boolean	The road element is inside an intersection consisting of multiple nodes in the path topology.
rightOfWay	RightOfWay	Information whether the ego vehicle has right of way or has to yield when turning into this node arm.

Used by: NodeProfileValue

3.4.5.4 NodeProfileValue

A Node profile entry consists of an array of NodeArm structures describing side roads meeting a path at a single location (i.e. at a single offset). One Node profile entry is needed for each offset when a complex intersection has side roads meeting the main path at different offsets.

```
struct NodeProfileValue extends ProfileValue {
    NodeArm [ ] arms
}
```

Name	Value	Description
arms	NodeArm []	Array of side roads

Used by: Node

3.4.5.5 SystemStatusProfileValue

Description of the structure.

```
struct SystemStatusProfileValue extends ProfileValue {
    GuidanceMode guidance
    Boolean simulating
}
```

Name	Value	Description
guidance	GuidanceMode	Description
simulating	Boolean	If true, ADASIS data does not correspond to physical reality.

Used by: System Status

3.4.5.6 AbsoluteVehiclePositionProfileValue

Description of the structure.

```
struct AbsoluteVehiclePositionProfileValue extends ProfileValue {
    VehiclePosition position
}
```

Name	Value	Description
position	VehiclePosition	Description.

Used by: Absolute Vehicle Position

3.4.5.7 SurfaceConditionProfileValue

Description of the structure.

```
struct SurfaceConditionProfileValue extends ProfileValue {
    Quality general
    SurfaceCondition surface
}
```

Name	Value	Description
general	Quality	has to be defined what good/bad means
surface	SurfaceCondition	Description.

Used by: Surface

3.4.5.8 SpeedProfileValue

Description of the structure.

```
struct SpeedProfileValue extends ProfileValue {
    Speed value
}
```

Name	Value	Description
value	Speed	Description.

Used by Average Speed, Flow Speed

3.4.5.9 LaneInfo

The Lane Info structure represents the basic description of a lane, and it is a part of each Lane Model profile entry.

Lanes are counted beginning with 1 from outmost to midmost in the direction of the lane including shoulder and other lanes (i.e., counting start from the rightmost lane for right-hand traffic and from the leftmost lane for left-hand traffic).

Within one Lane Model, each lane is represented by a Lane Info profile entry with a different lane number in ascending order.

Each lane has properties and a geometrical description. Lane geometry describes geometrical flow of lane and its geometrical expansion. The properties of a lane are:

- Type of lane (normal, exit, bus lane, HOV, shoulder etc.).
- Driving direction. This is a relative direction, the values are “along path direction” and “against path direction”, as in the definition of a profile.
- Two lines as curve shapes represent the left and right border line of a lane; a border line can be visible as painted line, dots, changed surface material., or a border line can be invisible if there are not paintings.
- One line represents the center line of a lane; the center line is usually not visible.

An ADASIS v3 Horizon Provider shall include a description for all lanes of a road including all carriageways and both directions for a bidirectional road.

Each Lane Info entry in an Lane Model profile entry describes one lane using its lane number, the type of the lane, the allowed driving directions, and a lane transition type. These attributes are mandatory for each lane.

In addition the geometry of the lane is described by optional boundary lines and an optional centerline. Each line geometry is referenced using the identifier to the Linear Object profile entry. If the boundary lines geometry or the centerline geometry is known and will be transmitted, a non-zero Linear Object identifier is used, if the geometry is not known or shall not transmitted, the special identifier 0 is used.

```
struct LaneInfo {
    UInt8 laneNumber
    RelativeDirection direction
    LaneTransition transition
    UInt32 types
    LinearObjId centerline
    LinearObjId leftBoundary
    LinearObjId rightBoundary
}
```

Name	Value	Description
laneNumber	UInt8	Description.
direction	RelativeDirection	The lane direction indicates whether the lane is drivable in the path's relative direction, i.e., the direction along the path in ascending offset order. The vehicle's driving direction is relative to the path direction, either along or against path direction.
transition	LaneTransition	The lane transition field indicates if the lane is currently forming or closing, if the lane starts or ends with or without an indication of the expected driving maneuver. From one road segment to next, especially over crossings, lanes are connected which each other. Simplest case is a one to one connection, e.g. three lanes on a motorway segment are connected with three lanes on the following segment. This means in the real world that all lanes are just continued. Another case is that more than one lane is connected with just one lane in following segments or vice versa than we have merging or splitting situations. The area of merging resp. splitting itself is represented by the lane attributed as merging resp. splitting.
types	UInt32	Each lane has at least one lane type, though in certain cases multiple types may be combined (for example, an entry lane merging into a HOV-only lane may be marked as both entry and HOV lane). Thus this field is a bit mask using values from the LaneTypeFlags enumeration.
centerline	LinearObjId	In a Lane Info entry the lane geometry is optionally described by the left and right boundary lines of the road surface area and an optional centerline. Curve shape of the lines describes the geometry of the lane area or lane centerline. To associate a geometry line, a Lane Info entry has two identifiers referencing indirectly the geometry for the left and the right boundary line, and the entry has an identifier referencing indirectly the geometry of the centerline. If a line identifier is 0, the geometry line is not known or does not exist.
leftBoundary	LinearObjId	
rightBoundary	LinearObjId	

Used by: LaneModelValue

3.4.5.10 LaneModelValue

The Lane Model profile represents a set of lanes for a stretch of a road, a road segment. The Lane Model profile in conjunction with the Lane Connectivity profile describe the logical view of the road which again represents a part of the horizon path.

An ADASIS v3 Horizon Provider shall start a new Lane Model entry when the number of lanes or a lane property changes. A road segment is cut and a new Lane Model entry is started in the following situations:

- Lane connectivity changes, e.g., at an intersection, when a lane starts anew or a lane starts forming or a lane ends or a lane ends merging into another lane.
- Physical divider or lane boundary between lanes starts or ends.
- One of the lane boundary types changes.

A Lane Model entry shall represent a stretch of road with the same set of lanes, no physical changes affecting a vehicle's transition between these lanes, and no legal changes affecting the transitions between the lanes.

Therefore, each road segment is described by a Lane Model entry through a set of detailed information for all lanes, including their driving directions. Group of all lanes transmitted with the horizon can include all imaginable types of lanes. For example:

- Lanes of other vehicles (opposite direction) on all carriageways of the road.
- Lanes for other vehicles like bicycle lanes or footpaths.
- Non-driveable lanes like shoulder lanes.
- Lanes like road verges and all areas beside or between the carriageways (under discussion).

The Lane Model entry has an array of Lane Info entries containing information describing every lane of a road on a path.

```
struct LaneModelValue extends ProfileValue {
    UInt8 totalNumberOfLanes
    LaneInfo [ ] laneInfos
}
```

Name	Value	Description
totalNumberOfLanes	UInt8	Description.
laneInfos	LaneInfo []	Description.

Used by: Lane Model

3.4.5.11 LaneConnectivityPair

This data is attached to the primary path through the intersection, even if it describes connectivity between two side paths.

```
struct LaneConnectivityPair {
    UInt8 initialLaneNumber
    PathId initialPath
    UInt8 newLaneNumber
    PathId newPath
}
```

Name	Value	Description
initialLaneNumber	UInt8	Description
initialPath	PathId	In case of the through path, it's the incoming side of the through path.
newLaneNumber	UInt8	Description
newPath	PathId	In case of the through path, it's the outgoing side of the through path.

Used by: LaneConnectivityValue

3.4.5.12 LaneConnectivityValue

Description of the structure.

```
struct LaneConnectivityValue extends ProfileValue {
    LaneConnectivityPair [ ] connectivityPairs
}
```

Name	Value	Description
connectivityPairs	LaneConnectivityPair []	Description.

Used by: Lane Connectivity

3.4.5.13 LinearObject

Description of the structure.

```
struct LinearObject {
    LinearObjId id
    LinearObjectType type
    LineMarking marking
    LineMarkingColour colour
}
```

Name	Value	Description
id	LinearObjId	Description
type	LinearObjectType	Description
Marking	LineMarking	None if not a LaneMarking.
colour	LineMarkingColour	None if not a LaneMarking.

Used by: LinearObjectDefinitionValue

3.4.5.14 LinearObjectDefinitionValue

Description of the structure.

```
struct LinearObjectDefinitionValue extends ProfileValue {
    LinearObject[] linearObjects
}
```

Name	Value	Description
linearObjects	LinearObject []	Description.

Used by:Linear Objects

- name of a structure using the struct, linked to its reference chapter

3.4.5.15 Curve

Description of the structure.

```
struct Curve {
    CurveType type
    WGS84Point[] points
}
```

Name	Value	Description
type	CurveType	Description.
points	WGS84Point	Description.

Used by: LineGeometry, RoadGeometryProfileValue

3.4.5.16 LineGeometry

Description of the structure.

```
struct LineGeometry {
    LinearObjId idLine
    Curve geometry
}
```

Name	Value	Description
idLine	LinearObjId	Description.
geometry	Curve	Description.

Used by: LanesGeometryProfileValue

3.4.5.17 LanesGeometryProfileValue

Description of the structure.

```
struct LanesGeometryProfileValue extends ProfileValue {
     LineGeometry[] geometries
}
```

Name	Value	Description
geometries	LineGeometry []	Description.

Used by: Lane Geometry

3.4.5.18 *RoadGeometryProfileValue*

Description of the structure.

```
struct RoadGeometryProfileValue extends ProfileValue {
    Curve roadCenterline
}
```

Name	Value	Description
roadCenterline	Curve	Description.

Used by: Road Geometry

3.4.5.19 *TrafficLightProfileValue*

Description of the structure.

```
struct TrafficLightProfileValue extends ProfileValue {
    TrafficLightLongitudinalPosition longitudinalPosition
    LateralPosition lateralPosition
    Double cycleTime
    TrafficLightState currentState
    YesNoUnknown turnOnRedAllowed
    Vector position
    Vector boundingBox
}
```

Name	Value	Description
longitudinalPosition	TrafficLightLongitudinalPosition	Description.
lateralPosition	LateralPosition	Description
cycleTime	Double	Description
currentState	TrafficLightState	Description
turnOnRedAllowed	YesNoUnknown	Unknown if no special sign, or if no information available.
position	Vector	Description
boundingBox	Vector	Description

Used by: Traffic Light

3.4.5.20 *EffectiveSpeedLimit*

Description of the structure.

```
struct EffectiveSpeedLimit extends ProfileValue {
    Speed value
    EffectiveSpeedLimitType type
}
```

Name	Value	Description
value	Speed	Description.
type	EffectiveSpeedLimitType	Description.

Used by: Effective Speed Limit

3.4.5.21 LocationObject

Description of the structure.

```
struct LocationObject extends ProfileValue {
    LocationObjectType type
    Int32 lateralOffset
    WGS84Point absolutePosition
}
```

Name	Value	Description
type	LocationObjectType	Description
lateralOffset	Int32	Centimeters from road centerline, positive to the right.
absolutePosition	WGS84Point	Description

Used by: Location Object

3.4.5.22 RegionCodeValue

String according to ISO 3166-2 encoded in an array of up to three characters.

Codes are standardized, but it's not necessarily well-defined which code to apply to what location.

```
struct RegionCodeValue extends ProfileValue {
    UInt8[] value
}
```

Name	Value	Description
value	UInt8[]	Use only 7-bit ASCII values. The array has an explicit size and will not be null-terminated.

Used by: Region Code

3.4.5.23 Condition

Description of the structure.

```
struct Condition polymorphic {
    ConditionType type
    YesNoUnknown appliesToEgoVehicle
}
```

Name	Value	Description
type	ConditionType	Description.
appliesToEgoVehicle	YesNoUnknown	For the moment when the vehicle will be there.

Used by: ConditionalRestrictionProfileValue, ConditionFuzzyTime, ConditionLoad, ConditionNumeric, ConditionTimeOfDay, ConditionTurnDirection, ConditionVehicleType, ConditionWeather, ExtendedSpeedLimitValue, TrafficSignValue

3.4.5.24 ConditionNumeric

Description of the structure.

```
struct ConditionNumeric extends Condition {
    UInt32 value
}
```

Name	Value	Description
value	UInt32	Description.

Used by:

3.4.5.25 ConditionVehicleType

Description of the structure.

```
struct ConditionVehicleType extends Condition {
    UInt32 vehicleTypeMask
}
```

Name	Value	Description
vehicleTypeMask	UInt32	RoadAccessFlags enum is used in the bit field

Used by:

3.4.5.26 ConditionLoad

Description of the structure.

```
struct ConditionLoad extends Condition {
    Load value
}
```

Name	Value	Description
value	Load	Description.

Used by:

3.4.5.27 ConditionTimeOfDay

Description of the structure.

```
struct ConditionTimeOfDay extends Condition {
     UInt16 startMinutes
     UInt16 endMinutes
}
```

Name	Value	Description
startMinutes	UInt16	0 .. 1439, in local time.
endMinutes	UInt16	1 .. 1440, may be lower than start for a time spanning midnight

Used by:

3.4.5.28 ConditionWeather

Description of the structure.

```
struct ConditionWeather extends Condition {
    Weather weather
}
```

Name	Value	Description
weather	Weather	Description.

Used by:

3.4.5.29 ConditionFuzzyTime

Description of the structure.

```
struct ConditionFuzzyTime extends Condition {
    FuzzyTime fuzzyTime
}
```

Name	Value	Description
fuzzyTime	FuzzyTime	Description.

Used by:

3.4.5.30 ConditionTurnDirection

Description of the structure.

```
struct ConditionTurnDirection extends Condition {
    LaneArrowMarking direction
}
```

Name	Value	Description
direction	LaneArrowMarking	Description.

Used by:

3.4.5.31 ExtendedSpeedLimitValue

Description of the structure.

```
struct ExtendedSpeedLimitValue extends ProfileValue {
    Speed value
    SpeedLimitSource source
    Condition[] conditions
}
```

Name	Value	Description
value	Speed	Description.
source	SpeedLimitSource	Description.
conditions	Condition[]	Combined with logical AND.

Used by: Extended Speed Limit

3.4.5.32 *TrafficSignValue*

Description of the structure.

```
struct TrafficSignValue extends ProfileValue
{
    SignType type
    UInt32 value
    SignLocationMask location
    Int32 shift
    Distance distance
    Distance length
    YesNoUnknown vms
    Condition[] panels
}
```

Name	Value	Description
type	SignType	Description.
value	UInt32	Description.
location	SignLocationMask	Description
shift	Int32	Distance from logical position to which the sign applies to the physical position.
distance	Distance	Distance given on additional panel.
length	Distance	Length of validity given on additional panel.
vms	YesNoUnknown	Sign is changeable.
panels	Condition[]	Information that is explicitly given on further panels.

Used by: Traffic Sign

3.4.5.33 *ConditionalRestrictionProfileValue*

Something is allowed or prohibited if conditions are fulfilled. Can be used without conditions to indicate permission or prohibition for all vehicles.

```
struct ConditionalRestrictionProfileValue extends ProfileValue {
    Boolean allowed
    Condition[] conditions
}
```

Name	Value	Description
allowed	Link to type	Description.
conditions	Link to type	Description.

Used by: Access Restriction, Overtaking Restriction

3.4.5.33.1 Examples

Below are some examples of conditional access restriction using the “accessAllowed”. All examples assume we’re in a passenger car.

Example 1

Situation: side path 2 leads to a road service storage area, access prohibited for all vehicles:
Result profile entry: Path: 2, offset: 0, accessAllowed: false

Usually a provider would not send any data for higher offsets – not of interest, since no vehicle may drive there anyway.

Example 2

Situation: Trucks are prohibited for the first 500 meters of side path 2
Result profile entries:
Profile 1: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, endOffset: 500,
accessAllowed: false
Conditions in profile:

- ConditionVehicleType
  - type: conditionTypeVehicle,
  - appliesToEgoVehicle: false,
  - vehicleTypeMask: Trucks

Profile 2: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 500, accessAllowed: true

Example 3

Situation: Trucks are prohibited only by night (22:00h - 6:00h), and it is 3 o'clock in the morning:
Result profile entriesentries:
Profile 1: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, accessAllowed: false
Conditions in profile:

- ConditionVehicleType
  - type: conditionTypeVehicle,
  - appliesToEgoVehicle: false,
  - vehicleTypeMask: Trucks

- ConditionTimeOfDay
  - type: conditionTypeTimeOfDay
  - appliesToEgoVehicle: true,
  - startMinutes: 1320,
  - endMinutes: 360

Example 4

Situation: Side path 2 is a bus-only road:
Result profile entriesentries:
Profile 1: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, accessAllowed: true
Conditions in profile:

- ConditionVehicleType
  - type: conditionTypeVehicle,
  - appliesToEgoVehicle: false,
  - vehicleTypeMask: Buses

Example 5

Situation: Side path 2 is a pedestrian zone with deliveries allowed from 7 a.m. to 10 a.m.
Result profile entriesentries:
Profile 1: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, accessAllowed: true
Conditions in profile:

- ConditionVehicleType
  - type: conditionTypeVehicle,
  - appliesToEgoVehicle: false,
  - vehicleTypeMask: Pedestrians

Profile 2: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, accessAllowed: true
Conditions in profile:

- ConditionVehicleType
  - type: conditionTypeVehicle,
  - appliesToEgoVehicle: false,
  - vehicleTypeMask: Delivery
- ConditionTimeOfDay
  - type: conditionTypeTimeOfDay
  - appliesToEgoVehicle: false,
  - startMinutes: 420,
  - endMinutes: 600

Example 6

Situation: Side path 2 is closed for all vehicles by night (22:00h to 6:00h), and always for heavy vehicles (above 12 t)
Result profile entriesentries:
Profile 1: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, accessAllowed: false

- Conditions in profile:
  - ConditionTimeOfDay
    - type: conditionTypeTimeOfDay
    - appliesToEgoVehicle: true,
    - startMinutes: 1320,
    - endMinutes: 360

Profile 2: type: ConditionalAccessRestrictionProfileValue, Path: 2, offset: 0, accessAllowed: false
Conditions in profile:

- ConditionNumeric
  - type: conditionTypeWeight,
  - appliesToEgoVehicle: false,
  - value: 12000

3.5 Epilogue

```
}
```

4 References

[1] The ADASIS Forum, "ADASIS v3 Protocol".

[2] ADASIS Forum, "ADASIS Forum," [Online].