

Data model for traffic light information

2013-08-22

Version 01-00-00

Jörg Freudenstein, AlbrechtConsult GmbH

in cooperation with

SEAMLESS, an ERA-NET ROAD project (<http://www.eranetroad.org>)

Content

Preliminary note	3
Data model for traffic light information.....	3
Static part of the model - StaticTrafficSignalPublication.....	6
TrafficStreams and StopLinePoints	6
Dynamic part of the model – DynamicTrafficSignalPublication.....	12
Next signal states by prognosis	15
Next signal states by vector	17
Queue information.....	21
Georeferencing.....	23
Linear element defined by points or id (ISO 19148)	25
OpenLR	26
Annex.....	28
Basics	28
DATEX II	28
Enterprise Architect.....	28
Version of the schema-file	28
Key to the UML representation.....	29
ETRS89.....	30
Versioning and IDs of elements in DATEX II (VersionedIdentifiables)	30
XML-Examples (Instances).....	31
StaticTrafficSignalInformation.....	31
DynamicTrafficSignalInformation I.....	33
DynamicTrafficSignalInformation II.....	35
TrafficSignalQueueInformation.....	37

Preliminary note

Elements not shown at all in any of the diagrams (nor entailed by another element shown on diagrams), are excluded from the profile. E.g. if a subset of literals is shown for an enumeration, the literals not listed shall not be used in this profile; if a subset of attributes is shown on a class, the attributes not shown shall not be used etc. The corresponding schema file does follow these rules.

Red comments in the definition tables indicate special remarks regarding the traffic light information-topic presented here. In these cases, the definitions originate from the DATEX Level A model and though are too generic for this special purpose.

For further explanation of notation see Appendix ,Key to the UML representation'.

Data model for traffic light information

This data model can be used to transfer forecasts for the signalization of traffic lights between a content provider and a service provider. The model consists of a static part which defines traffic streams and corresponding stop lines, as well as of a dynamic part in which all signal forecasts belonging to a traffic stream are transmitted.

Furthermore, there is a separate message for traffic queue information which also refers to the static model. Entry into both parts of the model is done via the **PayloadPublication**, which was expanded by means of so called Level-B extensions by 3 messages.

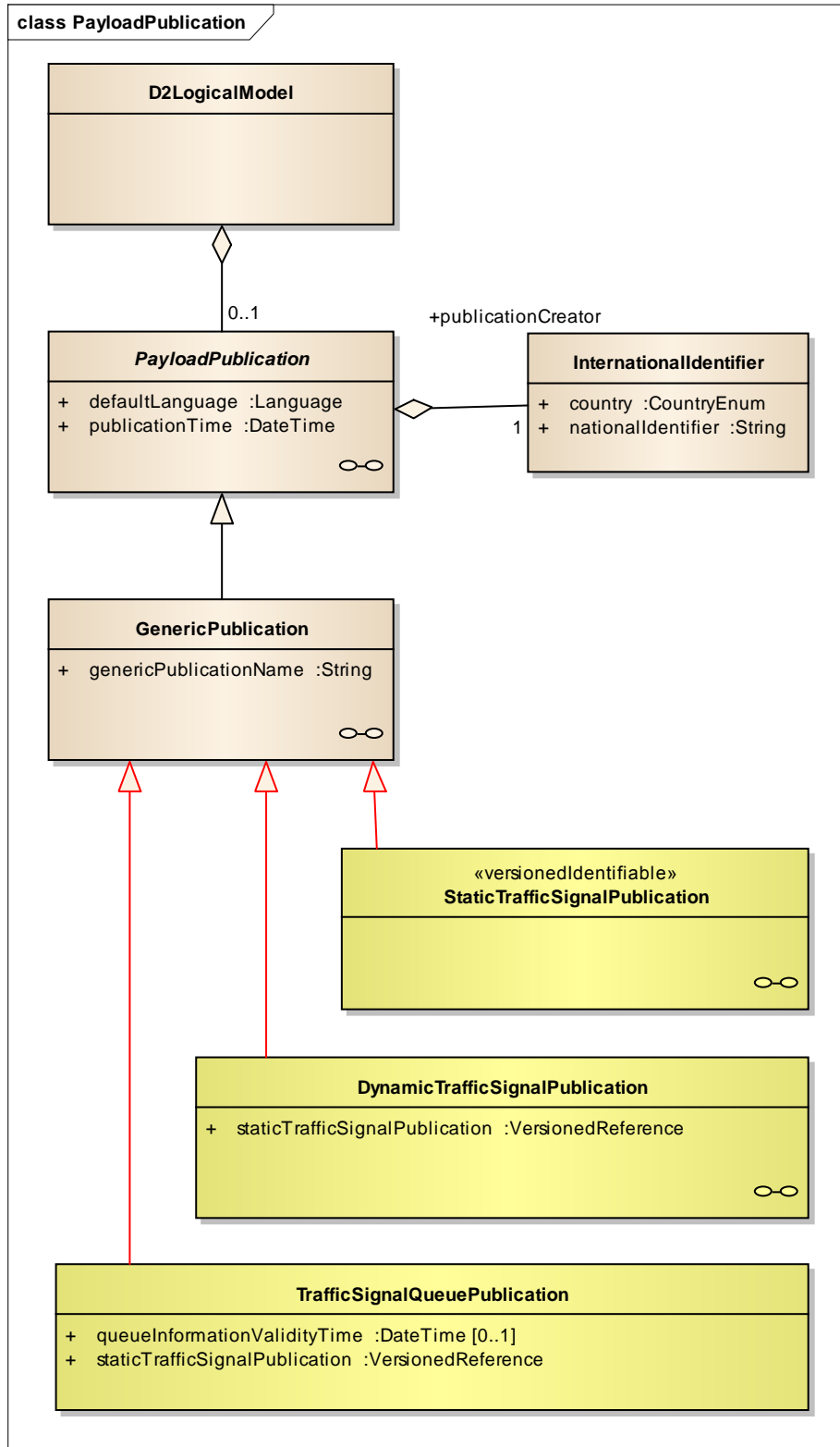


Figure 1: Basic model – three types of messages

Component	Definition
D2LogicalModel	The DATEX II logical model comprising exchange, content payload and management sub-models.
PayloadPublication	A payload publication of traffic related information or associated management information created at a specific point in time that can be exchanged via a DATEX II interface.
InternationalIdentifier	An identifier/name whose range is specific to the particular country.
GenericPublication	A publication used to make level B extensions at the publication level.

Table 1: Components of basic structure

Component	Attribute name	Definition	Multi- plicity	Type
Payload Publication	defaultLanguage	The default language used throughout the payload publication.	1	Language
	publicationTime	Date/time at which the payload publication was created.	1	DateTime
	publicationCreator	An identifier/name whose range is specific to the particular country.	1	InternationalIdentifier
International Identifier	country	ISO 3166-1 two character country code. <i>Also see below.</i>	1	Country Enum
	nationalIdentifier	Identifier or name unique within the specified country.	1	String
Generic Publication	genericPublication Name	The name of the generic publication. <i>Look at the different messages for the value of this String.</i>	1	String

Table 2: Attributes of basic structure



Language and country: In several parts of a message (including the data type ‘Multilingual String’) a declaration of language and/or country is expected. This should be expressed as a **two-letter code in lower case** according to ISO 639-1 resp. ISO 3166-1¹, e.g. **de** for German and Germany.

¹ In fact, ISO 3166-1 requires uppercase codes. There’s a pending DATEX issue to change the enumeration values to uppercase.

Static part of the model - StaticTrafficSignalPublication

TrafficStreams and StopLinePoints

In the **StaticTrafficSignalPublication** traffic streams are grouped together, which can have multiple **StopLinePoints**, i.e. intersections with stop lines. **StopLinePoints** are marked as 'identifiable' in order to reference them in a **TrafficSignalQueuePublication**.

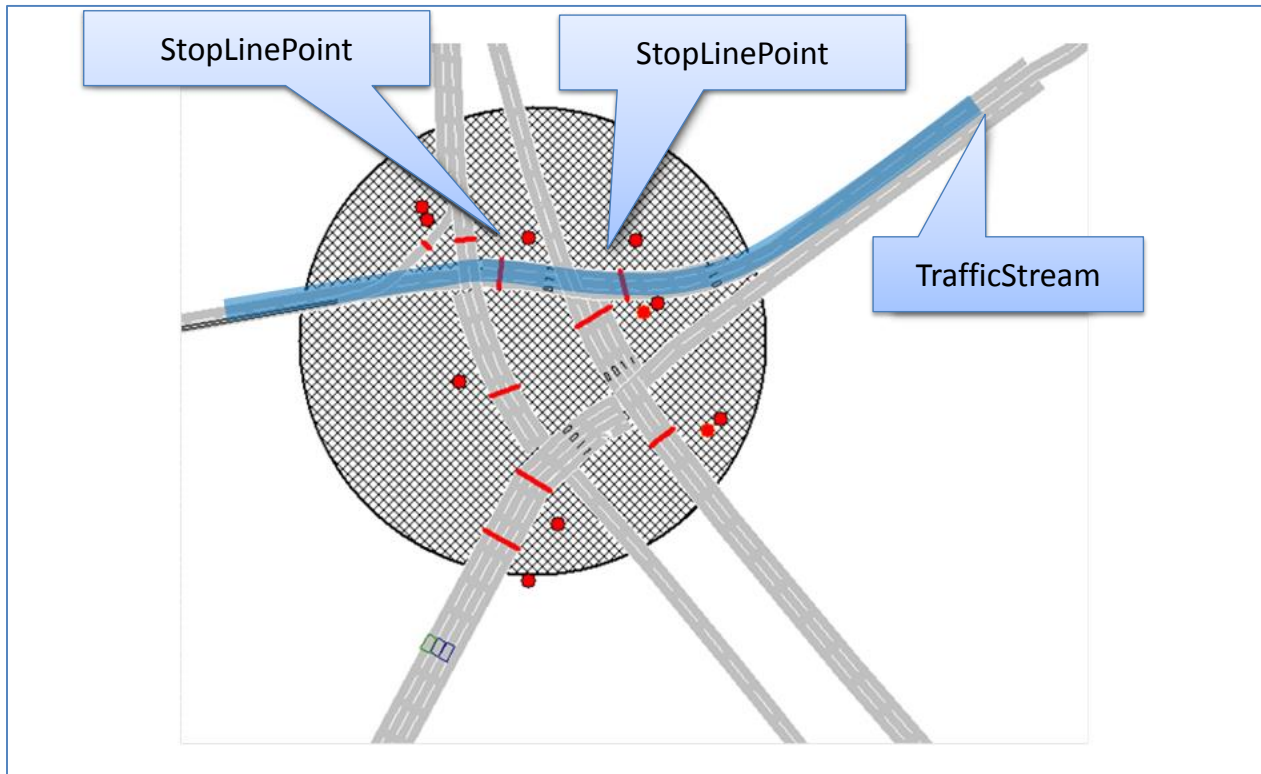


Figure 2: TrafficStreams and StopLinePoints

For two or more **StopLinePoints** which are touching the same stop line, the following cases can occur:

1. Different driving directions of the traffic streams (e.g. right and straight), potentially on identical lane(s):

Two separate **StopLinePoints** have to be modelled. In case the same signal group is responsible for both traffic streams, the respective attribute **mainSignalGroup** is simply filled with the same id.

2. Same driving direction of the traffic streams (e.g. coming from different driving directions on complex intersections)

The **StopLinePoint** is modeled once. For the second stream, **StopLinePointByReference** is used, which points to the first **StopLinePoint**. Possibly, some position parameters have to be set (overwritten), because they refer to the second traffic stream.

For an example see the following figure: Both blue traffic streams pass StopLinePoint 3 in the same direction towards A. One of the streams needs to define StopLinePoint 3, the other one just can set up a reference to StopLinePoint 3.

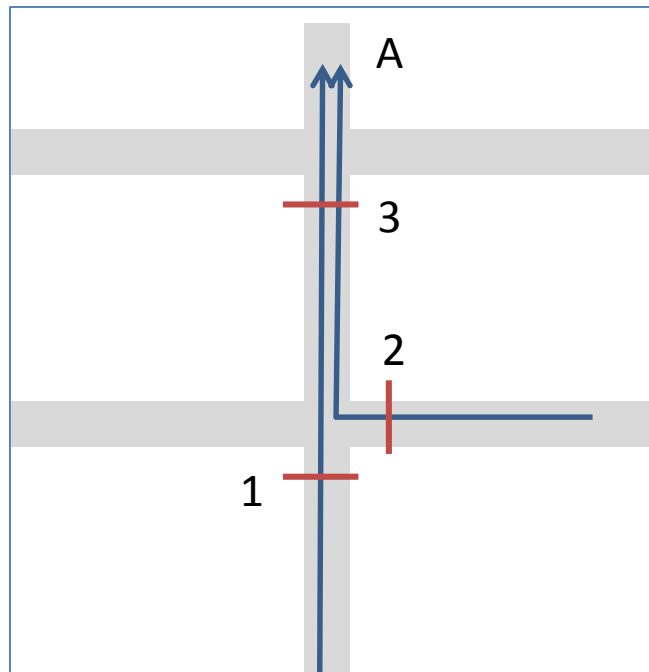


Figure 3: Second case: Same direction for traffic streams

Signal groups can be seen as some 'virtual' static components, which are not part of the model, anyway. They are just referred by ids (`mainSignalGroupId`, possibly `subSignalGroupId` for a second sub group, e.g. some turn right arrow), which are used as well in the dynamic model.

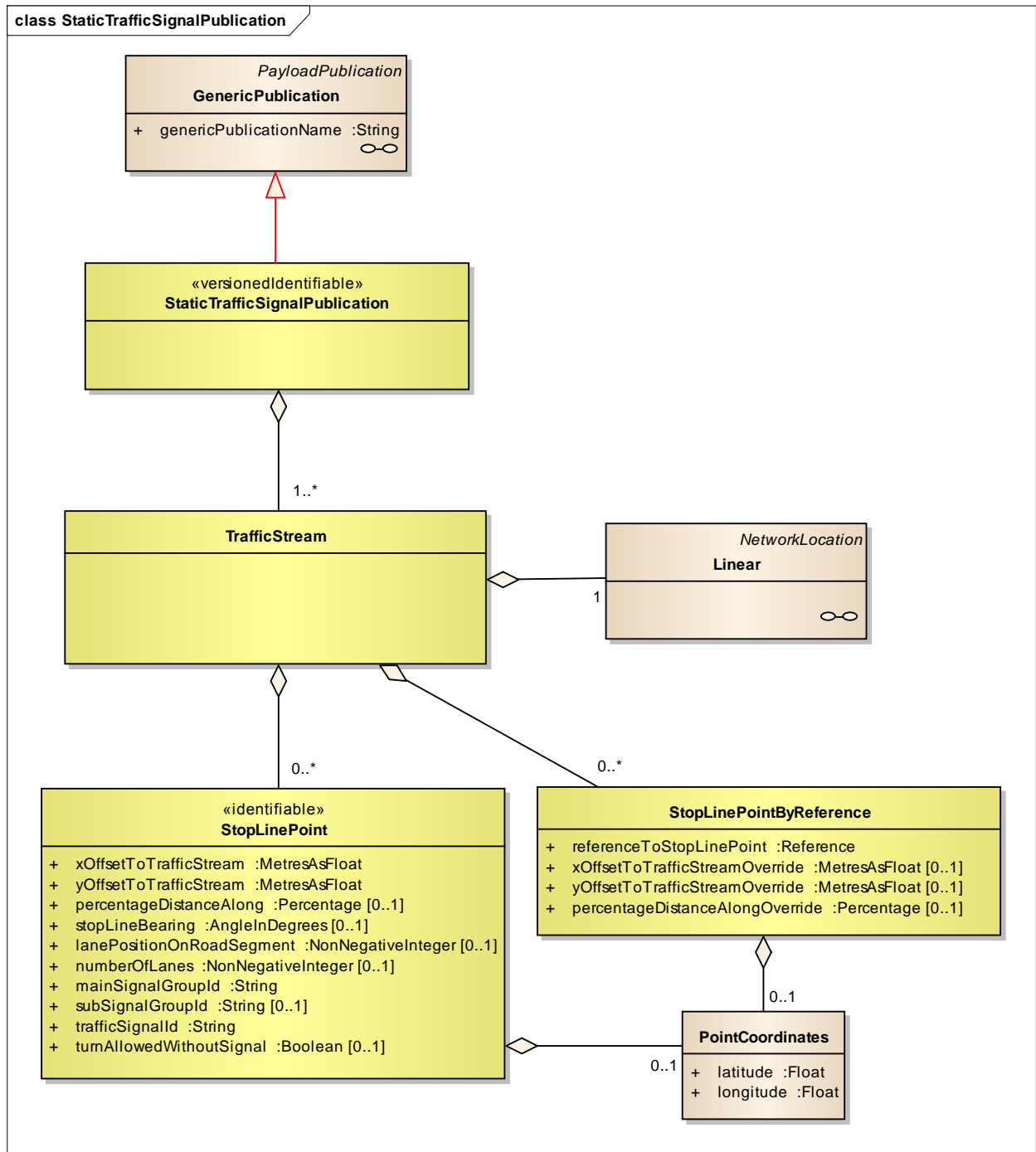




Figure 4: TrafficStream und StopLinePoint



Component	Definition	Stereotype
StaticTrafficSignalPublication	Collection of StopLines and TrafficStreams	versionedIdentifiable
StopLinePoint	A connection point between a StopLine and a TrafficStream with some georeference indication. One main signalGroup and optional a sub signalGroup are associated to this point.	identifiable
StopLinePointByReference	Reference to an existing stop line point in case the corresponding traffic signal groups are identical.	
TrafficStream	A driving connection (usually within an intersection) with references to one or more StopLinePoints.	
Linear	<i>Georeference for the TrafficStream: Usage of container for Linear Objects – see chapter Georeferencing. The linear object is, by definition, the imaginary center line of the traffic stream (i.e. the center of a single lane or their differentiation in case of two lines etc.)</i>	

Table 3: Components for static model



Component	Attribute name	Definition	Multiplicity	Type
Generic Publication	genericPublicationName	<i>For this message to fill with „StaticTrafficSignalInformation“</i>	1	String
StopLinePoint	lanePositionOnRoadSegment	Indicates the position of the right-most lane of the stop line point in correlation to the road segment (from right to left in driving direction). Example: numberOfLanes = 2 and lanePos.. = 3 means that lane 3 and 4 are belonging to the stop line point.	0..1	NonNegativeInteger
	mainSignalGroupId	Identifier of the main signal group. Signal groups are no DATEX components, but the IDs are reused in the dynamic model.	1	String

	numberOfLanes	The number of lanes on some road or for some traffic stream. If omitted, the numberOfLanes is unknown. <i>Here: Number of lanes for the traffic stream.</i> <i>A physical lane can belong to more than one traffic stream.</i>	0..1	NonNegativeInteger
	percentageDistanceAlong	A measure of distance along a linear element from the start of the element expressed as a percentage of the total length of the linear object. <i>Here: Percentage of x- and yOffset (StopLinePoint position) in contrast to the length of the traffic stream.</i>	0..1	Percentage
	stopLineBearing	Bearing of the vertical of the stop line (i.e. driving direction; 360 degree from North)	0..1	AngleInDegrees
	subSignalGroupId	If there is a sub signal group available, it can be referenced here. Signal groups are no DATEX components, but the IDs are reused in the dynamic model.	0..1	String
	trafficSignalId	Identifier of the traffic signal (e.g. FA1). Traffic signals are not DATEX components, the ID has no further effect within this model.	1..1	String
	turnAllowedWithoutSignal	It is allowed to turn right or left (depending on traffic stream) any time (observing priority rules)	0..1	Boolean
	xOffsetToTrafficStream	x offset from startpoint of traffic stream to StopLinePoint	1	MetresAsFloat
	yOffsetToTrafficStream	y offset from startpoint of traffic stream to StopLinePoint	1	MetresAsFloat
StopLinePointByReference	percentageDistanceAlongOverride	Overrides the percentageDistanceAlong information from the original stop line point.	0..1	Percentage

	referenceToStopLinePoint	Reference to an existing stop line point.	1..1	Reference
	xOffsetToTrafficStreamOverride	Overrides the x-offset information from the original stop line point.	0..1	MetresAsFloat
	yOffsetToTrafficStreamOverride	Overrides the y-offset information from the original stop line point.	0..1	MetresAsFloat

Table 4: Attributes for static model

Dynamic part of the model – DynamicTrafficSignalPublication

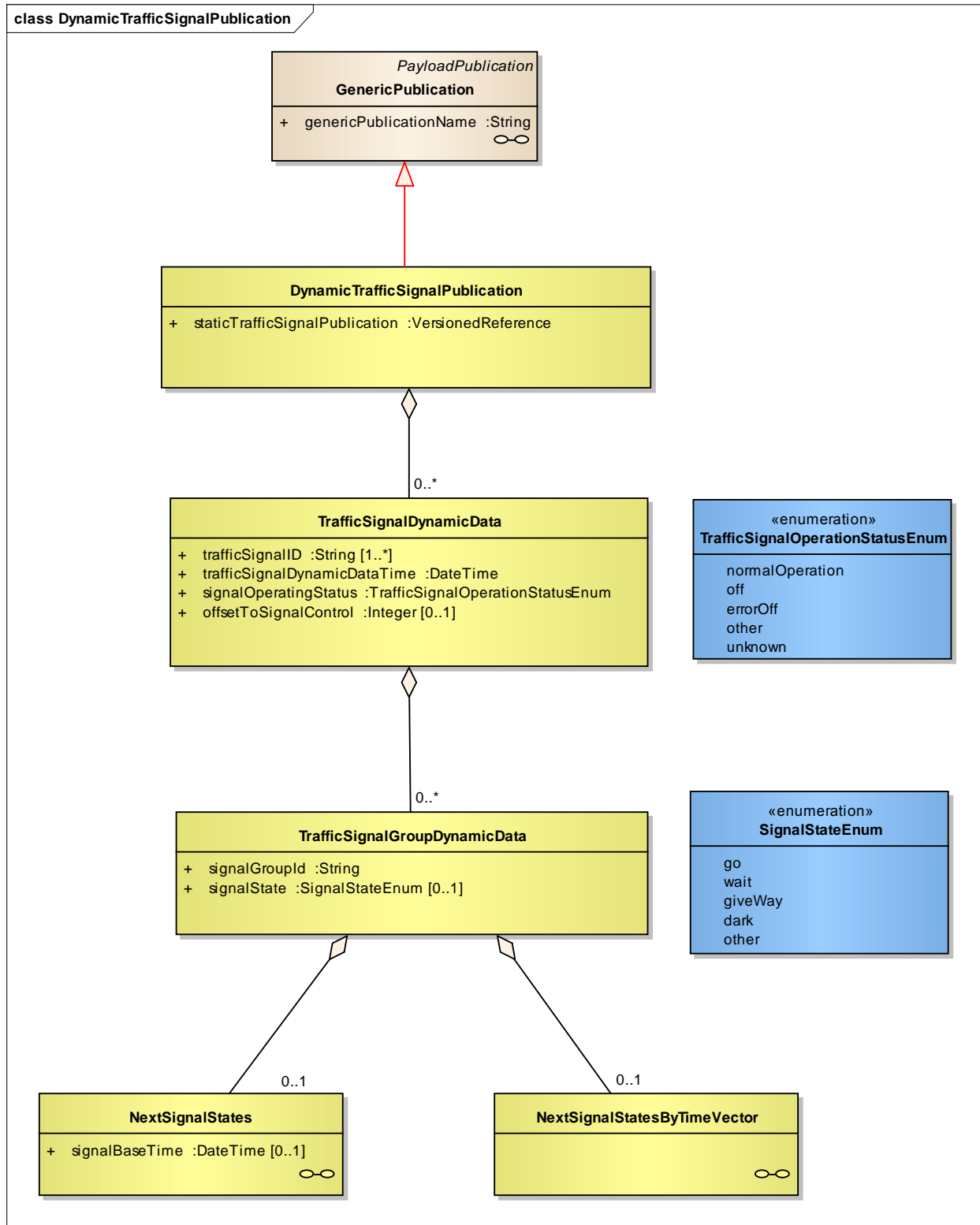


Figure 5: Dynamic traffic signal information

The dynamic data of a signal group is assigned to the static element **StoplinePoint** by a signal group id. Next cycles or changes of a signal group can be described either by a sequence of the next signal states or by a signal program vector (or possibly by both methods). Both methods are described in the following chapters.

Component	Definition	Stereotype
DynamicTrafficSignalPublication	DynamicTrafficSignalPublication	
NextSignalStates	Ordered set of next prognosis for this signal group	
NextSignalStatesByTimeVector	A schedule with vectors, which define the next signal states. Vectors can be stored and reused.	
TrafficSignalDynamicData	Information about one traffic signal	
TrafficSignalGroupDynamicData	Information about each signal group including the current signal state	

Table 5: Components for dynamic model

Component	Attribute name	Definition	Multiplicity	Type
Generic Publication	genericPublicationName	<i>For this message to fill with „DynamicTrafficSignalInformation“</i>	1	String
	staticTrafficSignalPublication	Reference to static model <i>See also in the annex technical details.</i>	1	VersionedReference
NextSignalStates	signalBaseTime	BaseTime for all calculation operations regarding the traffic signals operation	0..1	DateTime
TrafficSignalDynamicData	offsetToSignalControl	Error correction for offset to control systems (Milliseconds!) <i>Time difference between base time / supply data and actual controlling system (in case it is known)</i>	0..1	Integer
	signalOperatingStatus	Current operating status.	1	TrafficSignalOperationStatusEnum <i>See picture above</i>



	trafficSignalDynamicDataTimeStamp	Time stamp for the information about this traffic signal	1	DateTime
	trafficSignalID	Identifier of the traffic signal (e.g. FA1) <i>(one or more)</i>	1..*	String
TrafficSignalGroupDynamicData	signalGroupID	Signal groups have no DATEX component equivalence, but the IDs are reused in the static model (in component StopLinePoint)	1	String
	signalState	Description of the traffic signal state <i>(valid for the following point of time: trafficSignalDynamicDataTimeStamp)</i>	0..1	SignalState Enum

Table 6: Attributes for dynamic model

Next signal states by prognosis

This variant can be used if no cyclic sequence of signal states exists, i.e. especially in the case of traffic-adapted control without fixed cycle time.

An arbitrary number of subsequent states may be transmitted. With each state, numerous timing information, probability data (%) and, of course, the corresponding signal state can be transmitted. The timing information are offsets to the signal base time in seconds, except **signalStateDuration** which is no offset but the minimum release resp. minimum blocking time.

The following figure illustrates the relationships:

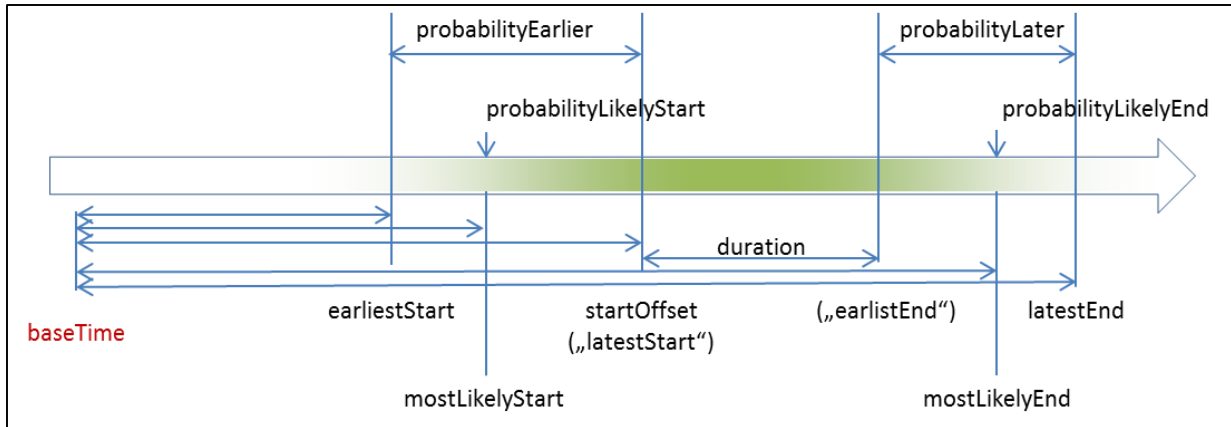


Figure 6: Prognosis-information

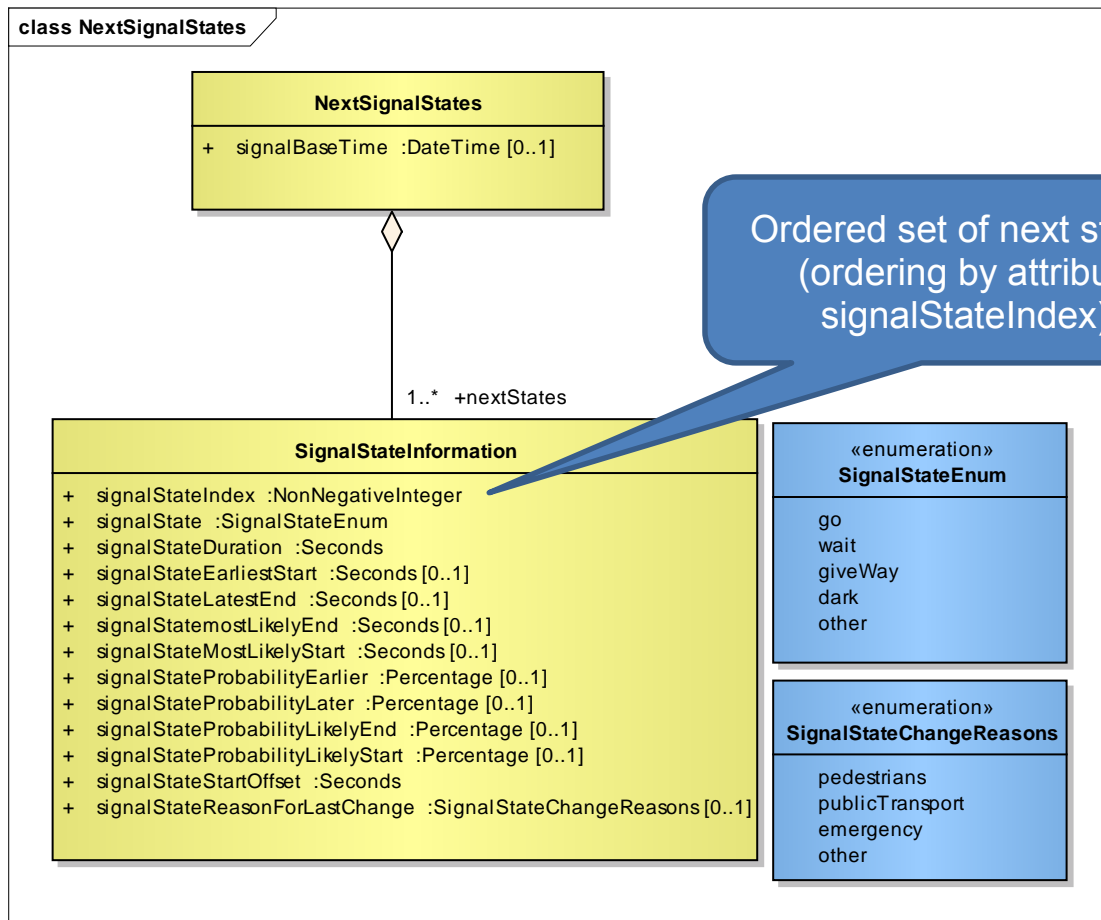


Figure 7: Prognosis information

The value **signalStateIndex** is constructed as attribute and is used to order the sequence of elements (starting at 0 counting up without interruptions).

With the attribute **signalStateReasonForLastChange** it can possibly be told whether pedestrians, public transport and emergency vehicles have 'caused' the last signal change.

Note: The **signalStateInformation** is not persistent, i.e. in case a new message arrives, the receiver deletes all previously obtained **signalStateInformation** and knows only those out of the current message. Thus, the **signalStateIndex** does not relate to any previously sent messages.

Next signal states by vector

In this variant, a vector of seconds (arbitrary size) is used which gives for each second the probability of traffic signal green. By being 'versionedIdentifiables' (see appendix), the vectors can be referenced.

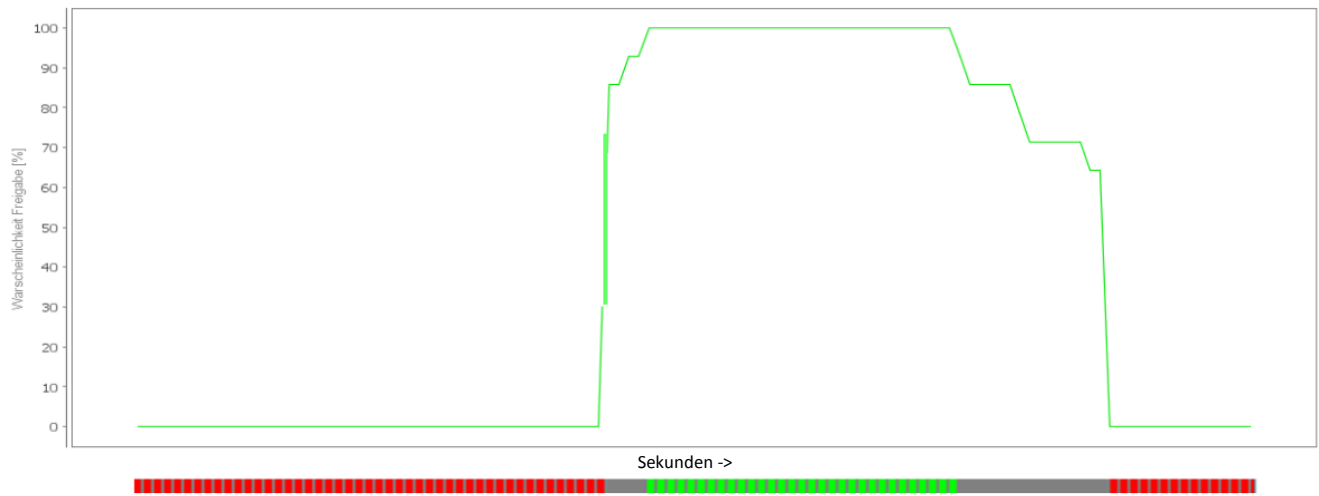


Figure 8: Vector information

The size of the vector can equal the cycle time but needs not. Only in the first case, the counting-second of the vector would correspond to the traffic light cycle second. In the model, both the size of the vector and optionally, if present, the cycle time is given.



As a convention - for bandwidth savings - vector elements may be omitted (even several times) if they contain the same probability value as its predecessor:

Second	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Bsp 1.: Traffic adapted	0%	0%	0%	0%	0%	5%	10%	15%	25%	40%	60%	100%	100%	100%	100%	60%	40%	0%	0%	0%
Efficient transmission	0%					5%	10%	15%	25%	40%	60%	100%				60%	40%	0%		
Bsp. 2: Fixed time control	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	100%	100%	100%	100%	0%	0%	0%	0%
Efficient transmission	0%											100%					0%			

Figure 9: Vector of 20 seconds and efficient transmission (lower lines)

The corresponding timetable is always required in the form of a complete update, i.e. the service provider only knows the latest sent schedule. The elements of the schedule are ordered and will be executed sequentially.

One element of the timetable covers:



- Reference to a vector within the same message
- **Start time** (optional):
If no start time is specified, this entry follows immediately after the previous one. If this entry is the first entry, it will be executed immediately.
Caution: If the first entry has a start time in future, until then no vector is valid (because the schedule must be complete). Best practise is to point with the first entry to the currently active vector (without start time).
A start time in the past is allowed and can be equated with an immediate start.

- **End time (mandatory):**
Usually the vector is processed up to the specified end time of the schedule entry. Only after this the starting time of the next entry will be considered. Any contradictions (e.g. subsequent start time is before end time) are thus ignored. If there is a time gap between the end time and subsequent start time, no vector is valid during this time gap.
If a program should run "until further notice", a maximus future end time has to be scheduled. Finishing this program can be achieved through the deployment of a new schedule, for example.
- **BaseTime:**
Unlike the start time, which relates to the start of the vector per se, the BaseTime gives the basis for the feedback calculation method, that is it specifies cycle second 0. Further cycles can be determined by modulo calculation. A vector can therefore start at 12 clock (StartTime) but have a BaseTime of 11:59:00. Thus, the vector would not start at 12 clock at second 0, but at second 60. Since the BaseTime only serves as modulo reference, it might even have larger intervals to start or end time, in theory, even across days.

A vector will be repeated as often as desired within its validity (until it reaches the EndTime), i.e. after reaching the last second in the vector, the vector runs again from scratch. This method is of use especially for fixed-time control. If a vector should be executed exactly once, the EndTime must be chosen according to the single length of the vector.

To terminate the validity of the signal vector processing program completely, an empty schedule must be sent.

The transmission of exactly one vector and one schedule with exactly one entry referring to this vector (at least with an EndTime) corresponds exactly to a signal program change on the specified point in time.

Further appliances:

- Transmission of multiple vectors and/or comprehensive schedules
- Short term change of vector by transmission of a vector in between (for instance public transport issue)
 - Same format like origin vector
 - Schedule must point to origin vector afterwards
(both vectors have to be transmitted in the message)

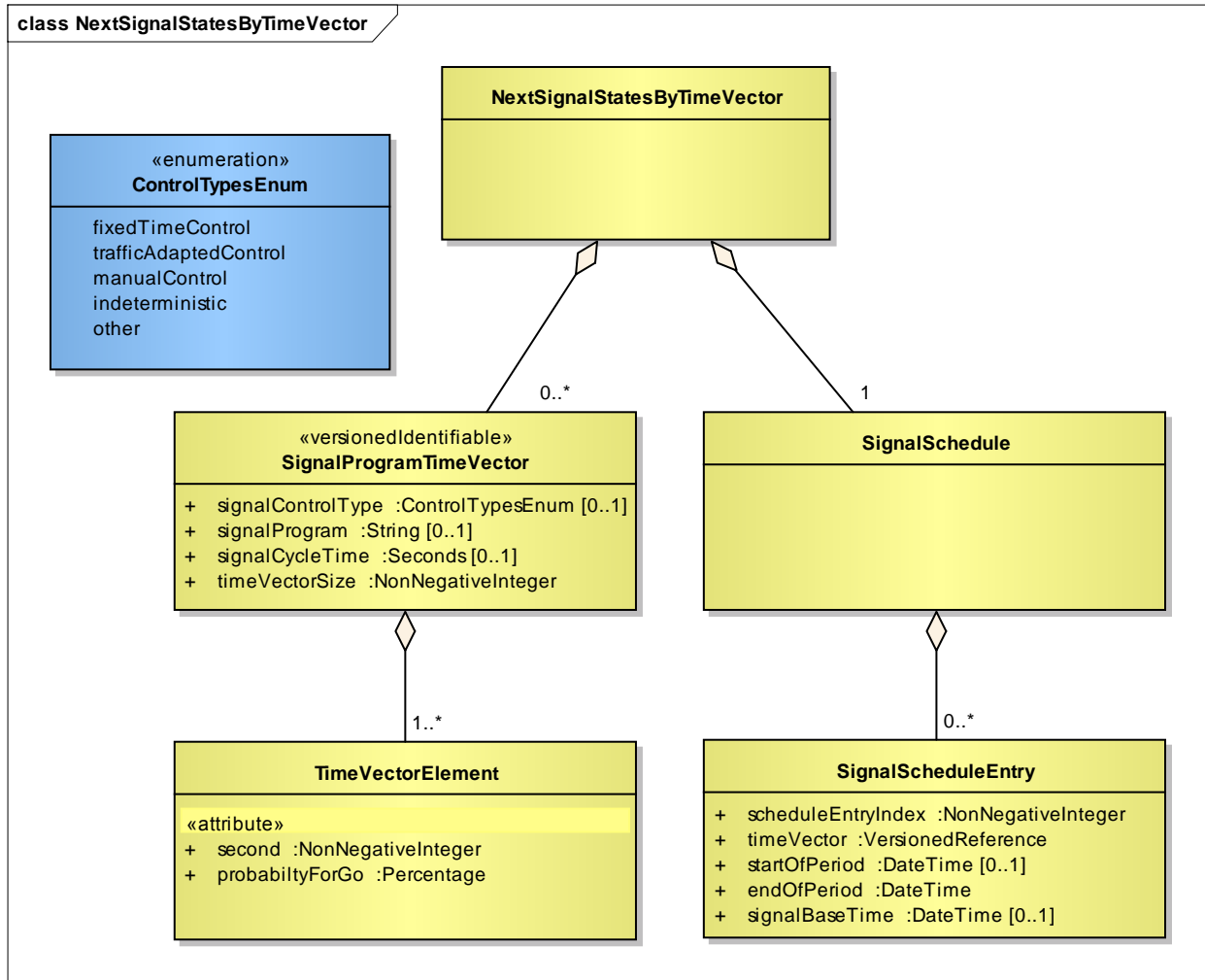


Figure 10: TimeVector and schedule

Component	Definition	Stereotype
NextSignalStatesByTimeVector	A schedule with vectors, which define the next signal states. Vectors can be stored and reused.	
SignalProgramTimeVector	Vector-Information. Vector elements can be left out, when having the same percentage value than their antecessor (compression issue)..	versionedIdentifiable
SignalSchedule	A schedule for the signalProgramVectors.	
SignalScheduleEntry	One single Entry for the schedule. For detailed rules refer to documentation.	
TimeVectorElement	One single vector element (one second).	

Table 7: Components for NextSignalStatesByTimeVector

Component	Attribute name	Definition	Multiplicity	Type
SignalProgram TimeVector	signalControlType	Type of control (if applicable).	0..1	ControlTypesEnum
	signalCycleTime	CycleTime of the current signal program, if applicable. This value is only for information, not for calculation issues!	0..1	Seconds
	signalProgram	Current signal program (an alphanumeric code is possible).	0..1	String
	timeVectorSize	Nominal size of vector (= seconds). Can be higher than real size of attached elements, because elements can be left out, when having the same value than their antecessor (compression issue).	1..1	NonNegativeInteger
SignalScheduleEntry	endOfPeriod	End of a period. <i>i.e. end of schedule entry – see above</i>	1..1	DateTime
	scheduleEntryIndex	Defines an order for the scheduleEntries. Must be serially numbered starting at 0.	1..1	NonNegativeInteger
	signalBaseTime	BaseTime for all calculation operations regarding the traffic signals operation. The current position within the vector is calculated as follows: $(\text{current time} - \text{signalBaseTime}) \bmod \text{signalVectorSize}.$ <i>See also description above.</i>	0..1	DateTime
	timeVector	Reference to the timeVector.	1..1	VersionedReference
	startOfPeriod	Start of period. <i>i.e. start of schedule entry – see above.</i>	0..1	DateTime

TimeVectorElement	probabilityForGo	Probability for green traffic light signal (Red signal = 0 %).	1..1	Percentage
	second	An index value for the signalVector. Starts at 0, but may have discontinuities (esp. to save communication range - see specification)	1..1	NonNegativeInteger

Table 8: Attributes for NextSignalStatesByTimeVector

Queue information

Queue information is transmitted in form of an autonomous message. An arbitrary number of queue information can be combined into one message, each corresponding to one **StopLinePoint**.

The validity timestamp (in the main component) can also point into the future to indicate prognosis. In every single queue information, this value can be adjusted by an offset of seconds.

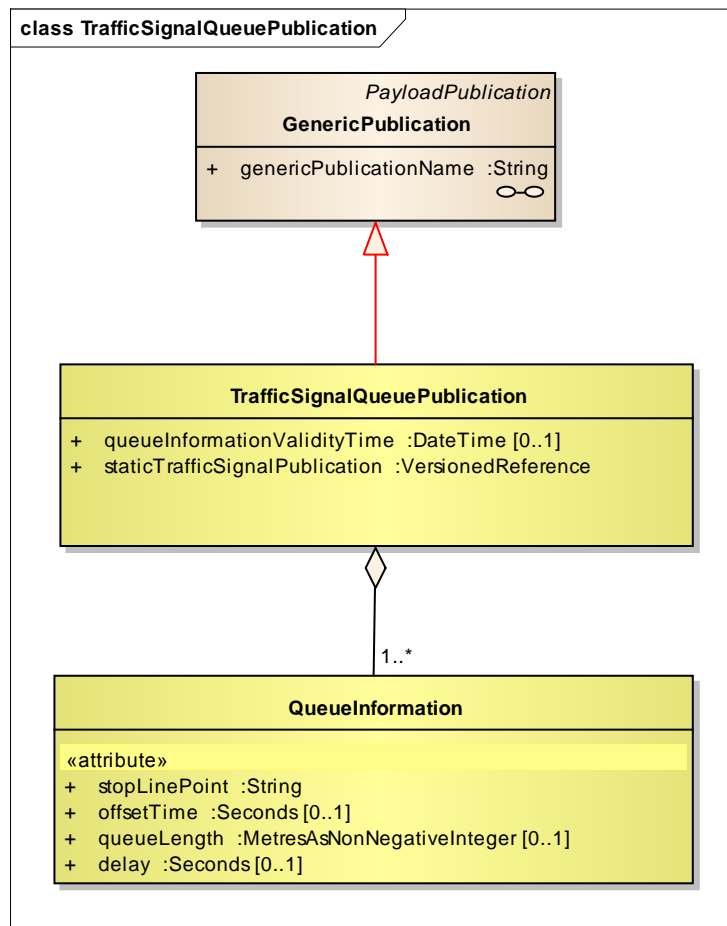


Figure 11: TrafficSignalQueuePublication

Component	Definition
QueueInformation	Traffic signal queuing information.
TrafficSignalQueuePublication	Publication for queuing information on traffic lights.

Table 9: Components for Queueinformation

All attributes of the **QueueInformation** component are designed as inline-attributes.

Component	Attribute name	Definition	Multiplicity	Type
Generic Publication	genericPublicationName	<i>For this message to fill with „TrafficSignalQueueInformation“</i>	1	String
QueueInformation	delay	The value of the additional travel time due to adverse travel conditions of any kind, when compared to "normal conditions", given in seconds.	0..1	Seconds
	offsetTime	The queueInformationValidityTime plus this offset gives the point of time for the present queue information. This might be a prediction.	0..1	Seconds
	queueLength	The length of a queue or the average length of queues in separate lanes due to a situation.	0..1	MetresAsNonNegativeInteger
	stopLinePoint	Reference to a static stopLinePoint. This attribute is not of type Reference because its attribute-property.	1..1	String
TrafficSignal QueuePublication	queueInformationValidityTime	A time stamp the information is valid for. Can be in the future to indicate prognosis. If omitted, the trafficSignalPublicationTimeStamp gives information about the validity. The value can be modified by an offset defined in the queue information.	0..1	DateTime
	staticTrafficSignalPublicationReference	Reference to static model	1..1	VersionedReference

Table 10: Attributes for Queueinformation

Georeferencing

The following table shows the available geo-referencing methods for the Traffic Light Information profile:

Traffic Light Information	Traffic stream	Stop line
Point		
PointByCoordinates		●
Alert C Point		
LocationForDisplay (Coordinates)		
TPEG-Loc		
PointAlongLinearElement (ISO 19148)		
OpenLR Point		
X/Y-offset to start point of traffic stream		●
Linear		
Alert C Linear	M2, M4	
TPEG-Loc		
LinearWithinLinear (ISO 19148)	●	
OpenLR Linear	●	
Area		
Alert C Area		
TPEG-Loc		
PolygonArea		
Other		
Supplementary PositionalDescription		
Predefined Locations		
ExternalReferencing		

color key
DATEX level A model
DATEX level B extension

text key
M2: ALERT-C method 2 (without offset)
M4: ALERT-C method 4 (with offset)
●: Available reference method

Figure 12: Overview Georeferencing

This table can be regarded as mandatory, i.e. variations not marked are not allowed.

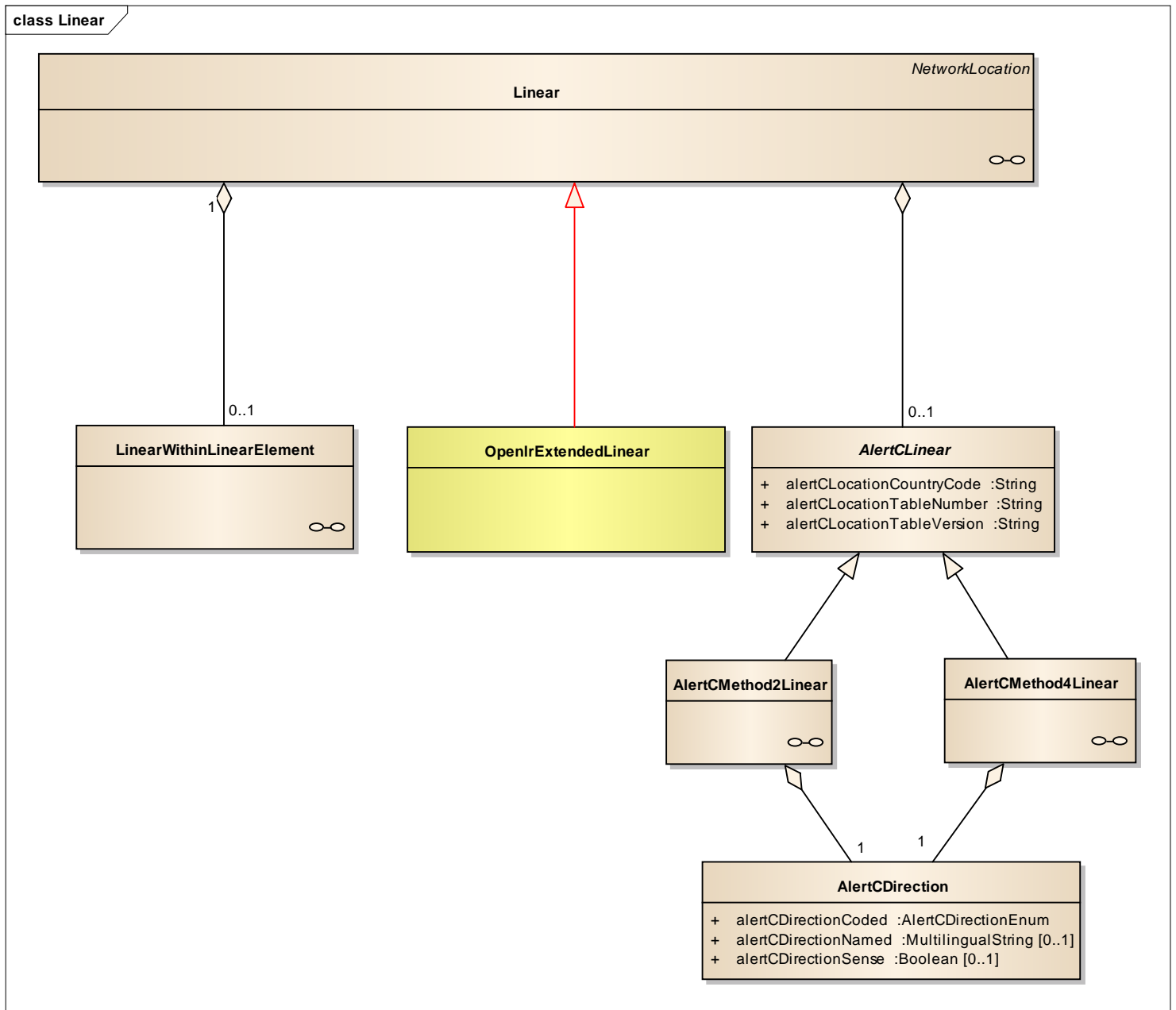


Figure 13: Linear Georeferencing for traffic streams

Linear element defined by points or id (ISO 19148)

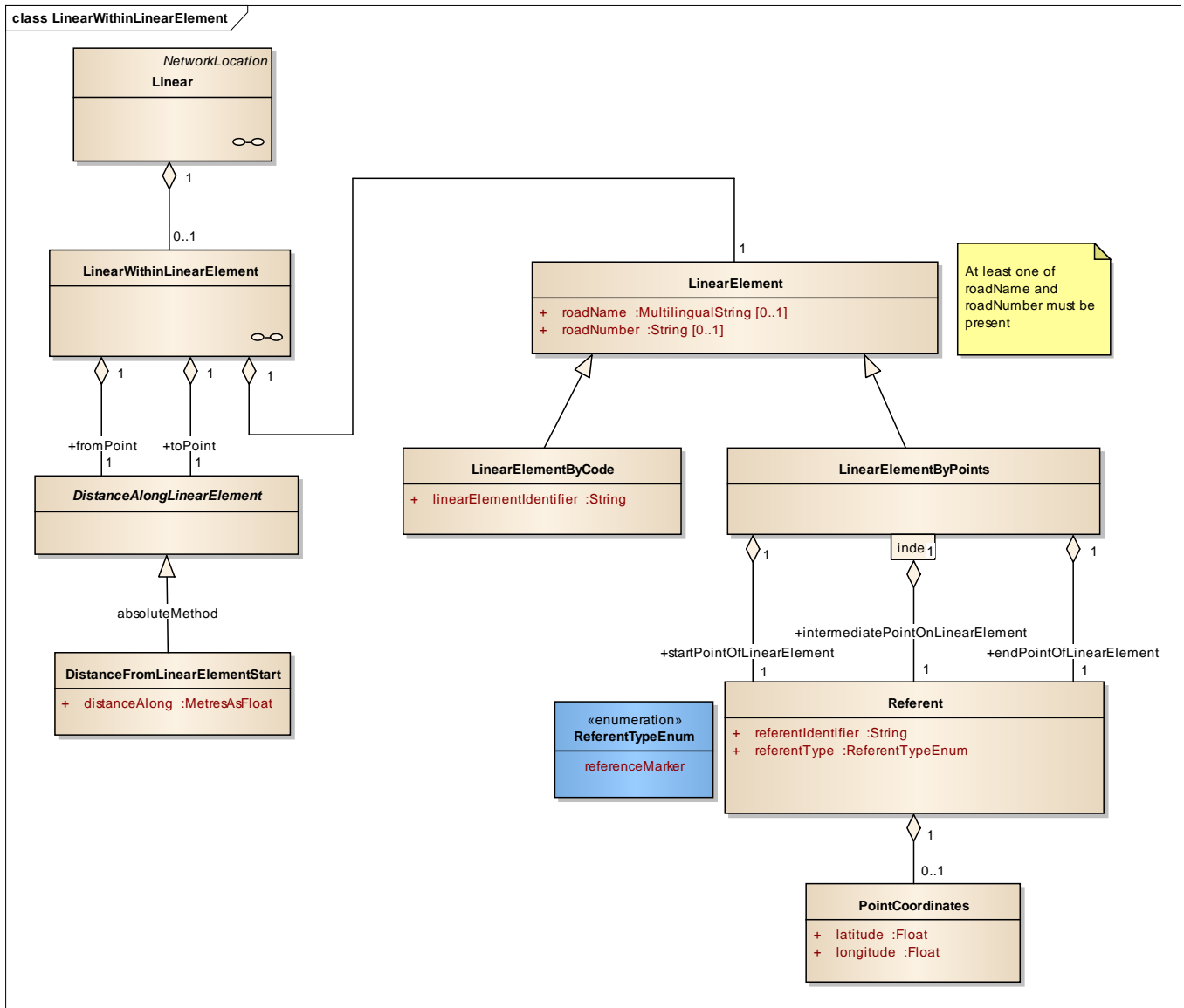


Figure 14: Details to ISO 19148 georeferencing

OpenLR

Another possibility of localisation, the DATEX II extension for OpenLR, based on open source, is available (<http://www.openlr.org>). Details are presented only in the form of the next two figures below; all other information can be obtained from the above website².

It is also possible to use an OpenLR base64 coded string instead of defining the location with OpenLR georeferencing methods (see next figure).

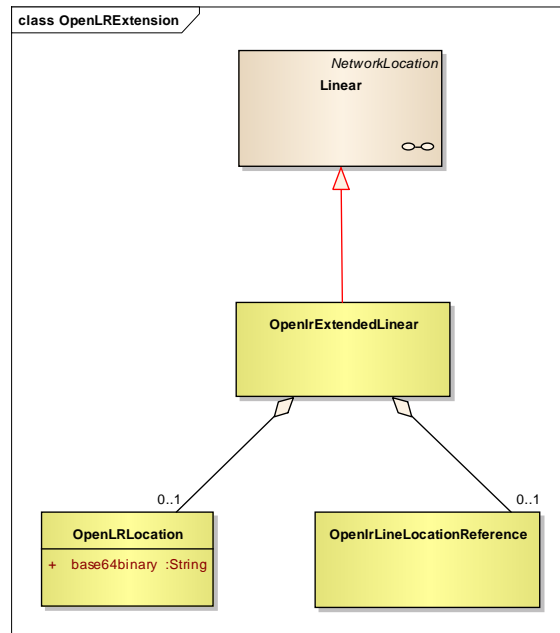


Figure 15: OpenLR with possibility of base64 coding

² Or the direct documentation via the following address:

http://www.datex2.eu/sites/www.datex2.eu/files/OpenLR_DATEX_II_extension_0.pdf

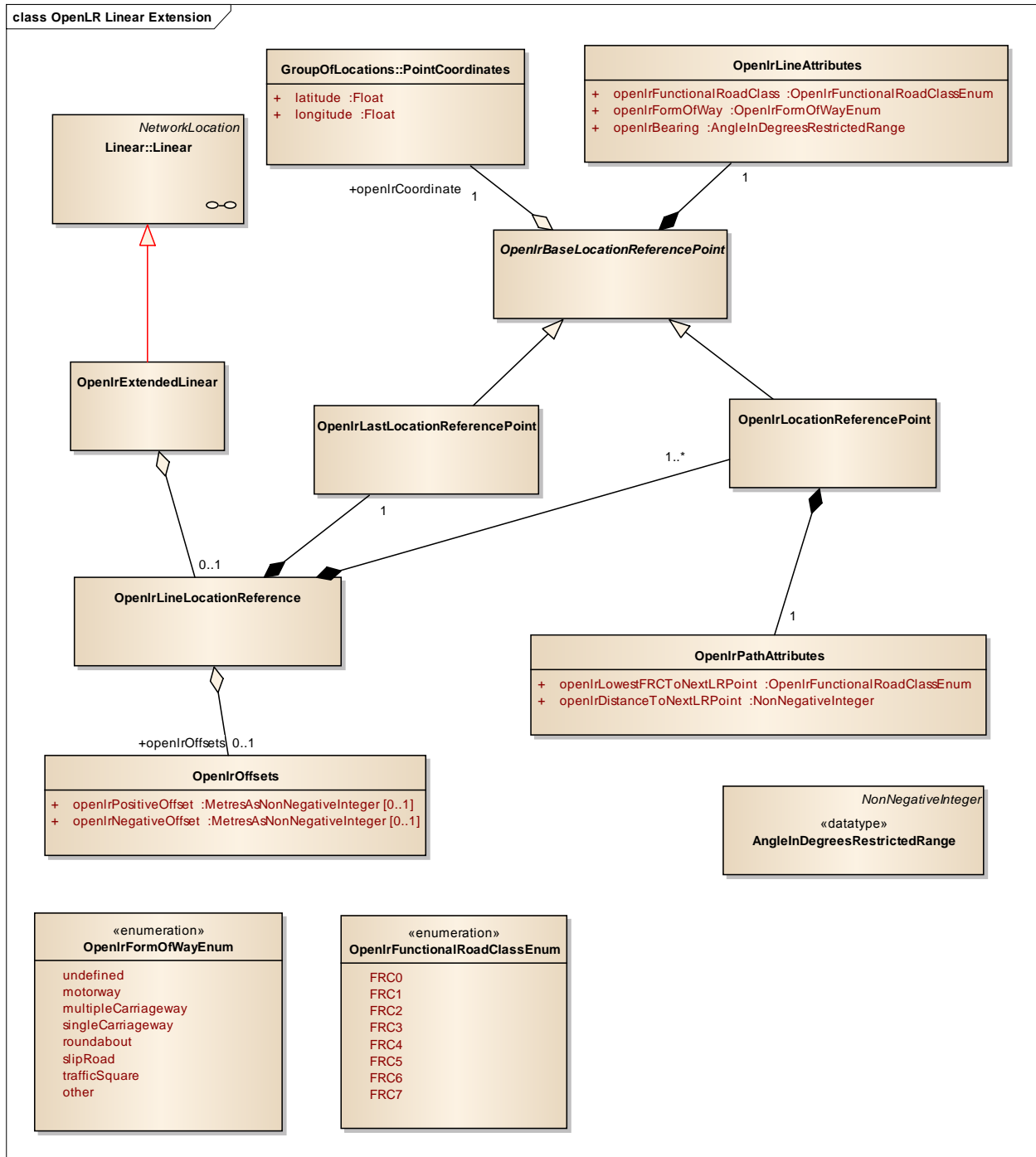


Figure 16: OpenLR linear extension

Annex

Basics

DATEX II

DATEX II provides a comprehensive data model for traffic related information. In the case topics are not covered by the basic model ("Level A"), it is possible to create a DATEX II extension ("Level B") with custom content.

To keep the specification clear, not the entire DATEX II data model is required, only a dedicated part of it. This part can be found in the attached schema file and in the present description as well.

For more information on DATEX visit the website www.datex2.eu.

Current version used for this profile is DATEX II version **2.2**.

Enterprise Architect

The corresponding UML model for this DATEX II profile is available for Enterprise Architect. This is an affordable UML modelling tool and can be purchased on this website <http://www.sparxsystems.com/>.

A free viewer for Enterprise Architect can be obtained from this address:

<http://www.sparxsystems.com.au/bin/EALite.exe>.

The UML model (Enterprise Architect file, *.eap) is based on the current version of DATEX II v2.1. In contrast to schema file, the UML model covers the whole set of DATEX version 2.1 and is not reduced to the profile itself.

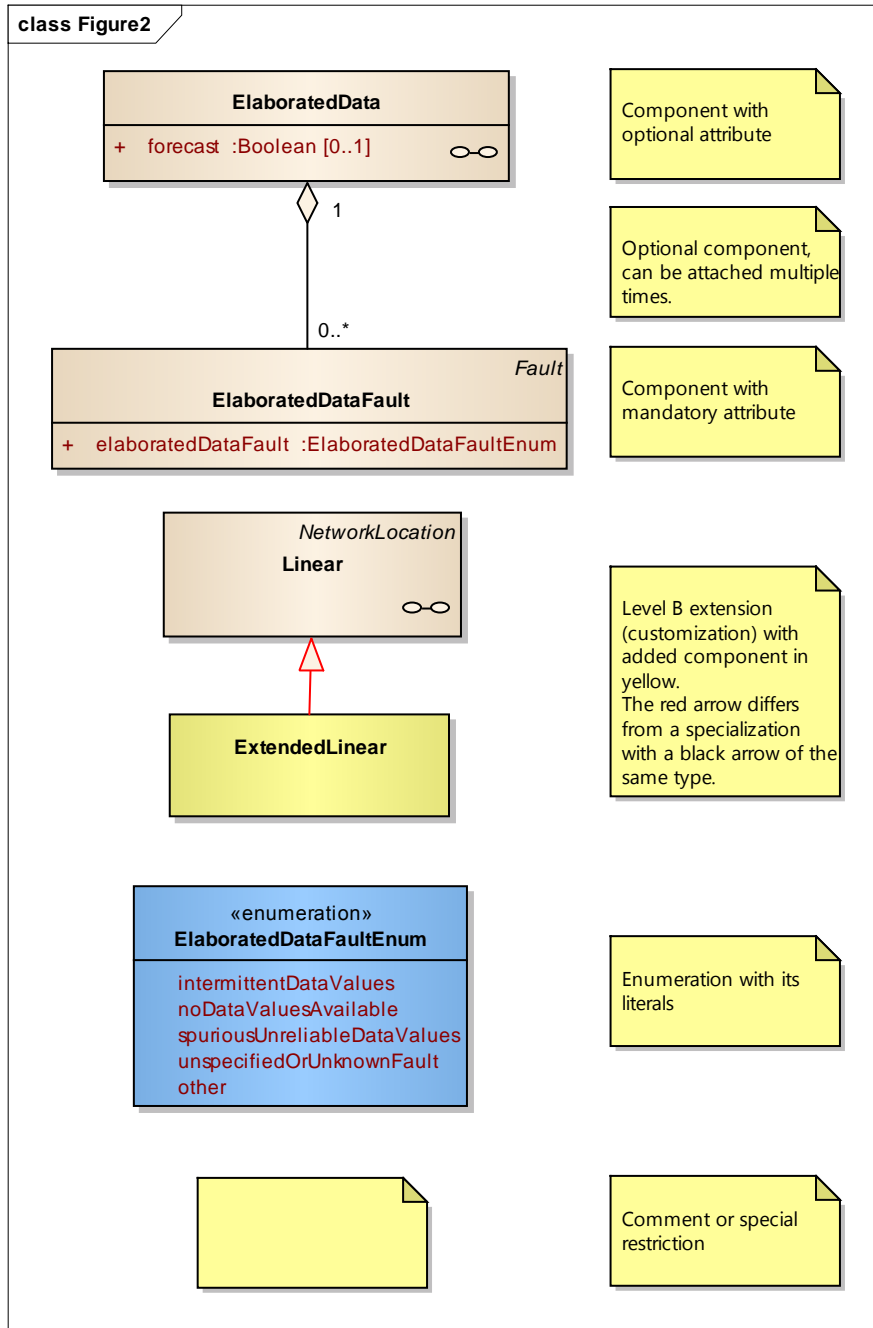
Version of the schema-file

The associated schema file is versioned in the same manner as this documentation. Information about its version can be found within the files in the following line:

```
<xs:attribute name="extensionVersion" use="optional" default="xx-yy-zz" />
```

Key to the UML representation

The following colours and semantics are used inside this document:



A red star * in the text or tables indicates obligatory elements.



A yellow exclamation mark indicates special restrictions or agreements that do not reveal themselves from the data model or the DATEX conventions.

ETRS89

DATEX II requires using geodetic coordinates according to the **European Terrestrial Reference System 1989** (ETRS89) for all coordinates. This was decided to be the unified official position reference system for all Germany in 1991 by the *Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland*. In fact, however, many systems still work with other systems of reference and / or using Cartesian coordinates.

It has to be checked for the filling of the data model whether the coordinates are according to ETRS89 or if appropriate conversions are provided (this especially applies to coordinates in Cartesian projection).

In many places coordinates according to WGS84 are in place; these can also be used (without conversion) because they correspond (with a slight inaccuracy) to the ETRS89-values (the deviation is about 1.20m +2 cm / year).

Versioning and IDs of elements in DATEX II (VersionedIdentifiables)

Elements resp. components, which are declared as **Identifiable** or **VersionedIdentifiable** have additional attribute(s) **id** or **id** and **version**. They can be referenced by these attributes. DATEX claims uniqueness („in time and space“) of the **id** resp. of **id** and **version** and points out GUIDs³ as an example.

For the data consumer, layout and mode of creation of the ID is not relevant – he only uses it to map and filter objects belonging together.

Reusing the same **id** and same **version** is only allowed, when all content of this element is exactly identical, for instance in case of sending ‘a copy’ of a **SituationRecord**. In all other cases, you have to increment the **version** or to use a different **id** (in case of a different meaning).

(Versioned) Identifiable elements are referenced by attributes with data type **Reference** resp. **VersionedReference**.

³ Refer to http://de.wikipedia.org/wiki/Globally_Unique_Identifier

XML-Examples (Instances)

StaticTrafficSignalInformation

This example includes a stop line, a traffic stream, the corresponding StopLinePoint as well as the associated SignalGroup.

The TrafficStream uses ALERT-C georeferencing (using a dummy value).

```
<?xml version="1.0" encoding="UTF-8"?>
<d2LogicalModel xmlns="http://datex2.eu/schema/2/2_0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://datex2.eu/schema/2/2_0 StaticTrafficSignalInformation.xsd" modelBaseVersion="2">
  <exchange>
    <supplierIdentification>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </supplierIdentification>
  </exchange>
  <payloadPublication xsi:type="GenericPublication" lang="en-us">
    <publicationTime>2012-05-01T11:12:20.0Z</publicationTime>
    <publicationCreator>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxx</nationalIdentifier>
    </publicationCreator>
    <genericPublicationName>StaticIntersectionInformation</genericPublicationName>
    <genericPublicationExtension>
      <staticTrafficSignalPublication id="064564C5-4429-4EF8-BF06-B962D6F13A52" version="4">
        <trafficStream>
          <stopLinePoint id="V501-87C">
            <xOffsetToTrafficStream>50</xOffsetToTrafficStream>
            <yOffsetToTrafficStream>3</yOffsetToTrafficStream>
            <percentageDistanceAlong>57</percentageDistanceAlong>
            <stopLineBearing>51</stopLineBearing>
            <lanePositionOnRoadSegment>4</lanePositionOnRoadSegment>
            <numberOfLanes>2</numberOfLanes>
            <mainSignalGroupId>IV2</mainSignalGroupId>
            <subSignalGroupId>IV3b</subSignalGroupId>
            <trafficSignalId>FN6</trafficSignalId>
            <turnAllowedWithoutSignal>false</turnAllowedWithoutSignal>
            <pointCoordinates>
              <latitude>1.23456</latitude>
              <longitude>1.23456</longitude>
            </pointCoordinates>
          </stopLinePoint>
        </trafficStream>
      </staticTrafficSignalPublication>
    </genericPublicationExtension>
  </payloadPublication>
  <linear>
    <alertCLinear xsi:type="AlertCMethod4Linear">
      <alertCLocationCountryCode>D</alertCLocationCountryCode>
      <alertCLocationTableNumber>1</alertCLocationTableNumber>
      <alertCLocationTableVersion>11.0</alertCLocationTableVersion>
      <alertCDirection>
        <alertCDirectionCoded>positive</alertCDirectionCoded>
      </alertCDirection>
      <alertCMethod4PrimaryPointLocation>
        <alertCLocation>
          <specificLocation>1234</specificLocation>
        </alertCLocation>
      </alertCMethod4PrimaryPointLocation>
    </alertCLinear>
  </linear>
</d2LogicalModel>
```

```
</alertCLocation>
<offsetDistance>
  <offsetDistance>354</offsetDistance>
</offsetDistance>
</alertCMethod4PrimaryPointLocation>
<alertCMethod4SecondaryPointLocation>
  <alertCLocation>
    <specificLocation>12345</specificLocation>
  </alertCLocation>
  <offsetDistance>
    <offsetDistance>450</offsetDistance>
  </offsetDistance>
</alertCMethod4SecondaryPointLocation>
</alertCLinear>
</linear>
</trafficStream>
</staticTrafficSignalPublication>
</genericPublicationExtension>
</payloadPublication>
</d2LogicalModel>
```


DynamicTrafficSignalInformation I

In this example, two prognosis for a traffic adaptive control are given in a dynamic data message. The IDs are referencing to the static example above.

```
<?xml version="1.0" encoding="UTF-8"?>
<d2LogicalModel xmlns="http://datex2.eu/schema/2/2_0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://datex2.eu/schema/2/2_0 DynamicTrafficSignalInformation.xsd" modelBaseVersion="2">
  <exchange>
    <supplierIdentification>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </supplierIdentification>
  </exchange>
  <payloadPublication xsi:type="GenericPublication" lang="en-US">
    <publicationTime>2012-06-13T18:14:34.0Z</publicationTime>
    <publicationCreator>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </publicationCreator>
    <genericPublicationName>DynamicTrafficSignalInformation</genericPublicationName>
    <genericPublicationExtension>
      <dynamicTrafficSignalPublication>
        <staticTrafficSignalPublication targetClass="StaticTrafficSignalPublication" id="064564C5-4429-4EF8-BF06-B962D6F13A52"
          version="2"/>
        <trafficSignalDynamicData>
          <trafficSignalID>FN6</trafficSignalID>
          <trafficSignalDynamicDataTime>2013-06-13T18:12:00.0Z</trafficSignalDynamicDataTime>
          <signalOperatingStatus>normalOperation</signalOperatingStatus>
          <offsetToSignalControl>50</offsetToSignalControl>
          <trafficSignalGroupDynamicData>
            <signalGroupId>IV2</signalGroupId>
            <signalState>go</signalState>
            <nextSignalStates>
              <signalBaseTime>2013-06-13T18:11:51.0Z</signalBaseTime>
              <signalStateInformation signalStateIndex="0">
                <signalState>wait</signalState>
                <signalStateDuration>30</signalStateDuration>
                <signalStateEarliestStart>265</signalStateEarliestStart>
                <signalStateLatestEnd>370</signalStateLatestEnd>
                <signalStateMostLikelyEnd>345</signalStateMostLikelyEnd>
                <signalStateMostLikelyStart>285</signalStateMostLikelyStart>
                <signalStateProbabilityEarlier>10</signalStateProbabilityEarlier>
                <signalStateProbabilityLater>15</signalStateProbabilityLater>
                <signalStateProbabilityLikelyEnd>80</signalStateProbabilityLikelyEnd>
                <signalStateProbabilityLikelyStart>61</signalStateProbabilityLikelyStart>
                <signalStateStartOffset>305</signalStateStartOffset>
              </signalStateInformation>
              <signalStateInformation signalStateIndex="1">
                <signalState>go</signalState>
                <signalStateDuration>50</signalStateDuration>
                <signalStateEarliestStart>340</signalStateEarliestStart>
                <signalStateLatestEnd>435</signalStateLatestEnd>
                <signalStateMostLikelyEnd>430</signalStateMostLikelyEnd>
                <signalStateMostLikelyStart>350</signalStateMostLikelyStart>
              </signalStateInformation>
            </nextSignalStates>
          </trafficSignalGroupDynamicData>
        </dynamicTrafficSignalPublication>
      </genericPublicationExtension>
    </payloadPublication>
  </d2LogicalModel>
```

```
<signalStateProbabilityEarlier>80</signalStateProbabilityEarlier>  
<signalStateProbabilityLater>80</signalStateProbabilityLater>  
<signalStateProbabilityLikelyEnd>75</signalStateProbabilityLikelyEnd>  
<signalStateProbabilityLikelyStart>75</signalStateProbabilityLikelyStart>  
<signalStateStartOffset>375</signalStateStartOffset>  
</signalStateInformation>  
</nextSignalStates>  
</trafficSignalGroupDynamicData>  
</trafficSignalDynamicData>  
</dynamicTrafficSignalPublication>  
</genericPublicationExtension>  
</payloadPublication>  
</d2LogicalModel>
```

DynamicTrafficSignalInformation II

In this example, a signalProgramVector for fixed time control is transferred. In the corresponding calendar, the vector is started immediately. Again the IDs are referring to the static example above.

```
<?xml version="1.0" encoding="UTF-8"?>
<d2LogicalModel xmlns="http://datex2.eu/schema/2/2_0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://datex2.eu/schema/2/2_0 DynamicTrafficSignalInformation.xsd" modelBaseVersion="2">
  <exchange>
    <supplierIdentification>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </supplierIdentification>
  </exchange>
  <payloadPublication xsi:type="GenericPublication" lang="en-US">
    <publicationTime>2012-06-13T18:14:34.0Z</publicationTime>
    <publicationCreator>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </publicationCreator>
    <genericPublicationName>DynamicTrafficSignalInformation</genericPublicationName>
    <genericPublicationExtension>
      <dynamicTrafficSignalPublication>
        <staticTrafficSignalPublication targetClass="StaticTrafficSignalPublication" id="064564C5-4429-4EF8-BF06-B962D6F13A52"
          version="2"/>
        <trafficSignalDynamicData>
          <trafficSignalID>FN6</trafficSignalID>
          <trafficSignalDynamicDataTime>2013-06-13T18:12:00.0Z</trafficSignalDynamicDataTime>
          <signalOperatingStatus>normalOperation</signalOperatingStatus>
          <offsetToSignalControl>50</offsetToSignalControl>
          <trafficSignalGroupDynamicData>
            <signalGroupID>IV2</signalGroupID>
            <signalState>go</signalState>
            <nextSignalStatesByTimeVector>
              <signalProgramTimeVector id="994BB957-04C9-4A10-92A2-E5562B3C90E6" version="23">
                <timeVectorSize>90</timeVectorSize>
                <timeVectorElement second="0">
                  <probabilityForGo>0</probabilityForGo>
                </timeVectorElement>
                <timeVectorElement second="29">
                  <probabilityForGo>100</probabilityForGo>
                </timeVectorElement>
                <timeVectorElement second="74">
                  <probabilityForGo>0</probabilityForGo>
                </timeVectorElement>
              </signalProgramTimeVector>
            <signalSchedule>
              <signalScheduleEntry scheduleEntryIndex="0">
                <timeVector targetClass="SignalProgramTimeVector" id="994BB957-04C9-4A10-92A2-E5562B3C90E6" version="23"/>
                <endOfPeriod>2012-06-13T19:30:00.0Z</endOfPeriod>
                <signalBaseTime>2013-06-13T18:11:51.0Z</signalBaseTime>
              </signalScheduleEntry>
            </signalSchedule>
          </nextSignalStatesByTimeVector>
        </trafficSignalGroupDynamicData>
      </dynamicTrafficSignalPublication>
    </genericPublicationExtension>
  </payloadPublication>
</d2LogicalModel>
```

```
</trafficSignalDynamicData>  
</dynamicTrafficSignalPublication>  
</genericPublicationExtension>  
</payloadPublication>  
</d2LogicalModel>
```

TrafficSignalQueueInformation

This example gives congestion information for the signal group of the static example. The time stamp (which is in the future in contrast to the message time) indicates a prognosis message. Two messages for two StopLinePoints are described (note that only one of them is well-defined in the static example message).

```
<?xml version="1.0" encoding="UTF-8"?>
<d2LogicalModel xmlns="http://datex2.eu/schema/2/2_0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://datex2.eu/schema/2/2_0 TrafficSignalQueueInformation.xsd" modelBaseVersion="2"
extensionName="TrafficSignalInformation" extensionVersion="00-04-00">
  <exchange>
    <supplierIdentification>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </supplierIdentification>
  </exchange>
  <payloadPublication xsi:type="GenericPublication" lang="en-US">
    <publicationTime>2012-06-13T18:14:34.0Z</publicationTime>
    <publicationCreator>
      <country>de</country>
      <nationalIdentifier>DE-MDM-xxxxxxx</nationalIdentifier>
    </publicationCreator>
    <genericPublicationName>DynamicTrafficSignalInformation</genericPublicationName>
    <genericPublicationExtension>
      <trafficSignalQueuePublication>
        <queueInformationValidityTime>2012-10-05T12:00:00.0Z</queueInformationValidityTime>
        <staticTrafficSignalPublication targetClass="StaticTrafficSignalPublication" id="064564C5-4429-4EF8-BF06-B962D6F13A52"
version="4"/>
          <queueInformation stopLinePoint="V501-87C" delay="100" offsetTime="15" queueLength="125" />
          <queueInformation stopLinePoint="V500-84C" delay="90" queueLength="100" />
        </trafficSignalQueuePublication>
      </genericPublicationExtension>
    </payloadPublication>
  </d2LogicalModel>
```