Open Communication Interface for Road Traffic Control Systems

Offene Schnittstellen für die Straßenverkehrstechnik

OCIT-C Center to Center Data

OCIT-C_Data_V2.0_A02

OCIT Developer Group (ODG) & Partner

OCIT-C Center to Center Data

Document: OCIT-C_Data_V2.0_A02

Issued by: ODG & Partner

Contact: www.ocit.org

Contents

Document history

Version State	Date	Distribution List	Comments
V2.0_A01	2018-05-23	PUBLIC	For OCIT-C V2.0 ODG Homepage
V2.0_A02	2024-04-19	PUBLIC	3.3 Beschreibung ergänzt 3.7 OperatingMessages aktualisiert 3.8 Empfohlene Identifizierung ergänzt 3.10.4 ControlMethodParameter aktualisiert 3.10.5 Beschreibung der LSA-Zustände (1001 bis 1004, 1006, 1007) angepasst 3.10.5 LSA-Zustände 1009 bis 1016 ergänzt 3.11.4 AP-Wert DPV1 ergänzt 3.11.4 Beschreibung der Beauftragung / Filterung von AP Werten angepasst 3.11.5 R09-Telegramm ergänzt 3.12.2 Titel angepasst 3.12.2.3 Oepnv-Meldepunkteliste: Textergänzung 3.12.2.4 Oepnv-Anforderung: Textergänzung

1 Introduction

OCIT-C stands for Open Communication Interface for Road Traffic Control Systems - Center to Center. OCIT-C covers the communication functions between central traffic control and traffic guidance systems:

- Traffic control centers and traffic management centers (urban, regional, interregional)
- Traffic engineer work place with traffic control centers
- Parking guidance systems, parking facility systems
- Roadworks management systems
- Local internet users (city information online)

The definition and maintenance of the OCIT-C interface is carried out by the ODG and their partners.

OCIT-C is a standard that supplements OCIT-O perfectly. Using OCIT-C and OCIT-O for the communication from central systems to field devices covers all requirements from traffic control through to primary traffic management.

OCIT-C is therefore geared towards practical requirements. With its low implementation costs, its use is also suitable for solutions with small budgets.

The characteristic properties of OCIT-C are:

- Exchange protocol with a simple request-response communication pattern (direct querying of data) based on the SOAP standard.
- Definition of a comprehensive data model in the process data area containing all partial sections of the traffic control and traffic guidance, use of OCIT-C supply data models for TSS.
- System integration and desired adaptations are governed in advance by project planning.
- Conformity tests for the protocol are carried out in a test environment provided at www.ocit.org. Tests of entire implementations (protocol and data contents) are carried out on a project-specific basis.
- Expansions to the DATEX II components are possible based on your project requirements.

The OCIT-C interface is open and can be used in various systems, predominantly in the road traffic technology sector. The aim of this document is to describe the standardised data that can be transmitted via OCIT-C. The aim of this document is not to describe the protocol. This is described in the document "OCIT-C protocol".

The following communication modules are currently standardized:

- **Traffic messages**
  This communication module is used to transmit messages regarding roadworks, events, and faults with the following attributes: Location, time period, daily validity, status,

classification, impact, e.g. capacity reduction, restrictions, e.g. speed or weight, referencing regarding overlapping messages, type of recording (manual or automatic) and free text.

- **Traffic data**
  This communication module is used to transmit measurements or derived values from various detection devices.

- **Parking data**
  This communication module is used to transmit data from parking facilities: Name, location, short and long term parks, occupancy, vacant parking spots, forecasted values (trend) and status of the parking facility.

- **Weather and environmental data**
  This communication module is used to transmit data from respective detection facilities:

  - Sensor type with the attributes location, point or road related measurement values,

  - Data corresponding to the TLS: Precipitation (intensity, likelihood), temperature (air, ground), wind, humidity, air pressure, visibility, state of the lane surface, sunrise, sunset, brightness, and

  - additional data: Radiation balance, cloud cover level, ozone, sulfur dioxide, benzene, soot, carbon monoxide, dust, weather forecast.

- **Cameras**
  This communication module is used to transmit data to control monitoring cameras and to transfer images.

- **Situations and strategies**
  This communication module is used to transmit the descriptions of traffic situations or control strategies and input for strategic traffic control measures.

- **Operating messages**
  This communication module is used to transmit operating messages from traffic equipment, such as from traffic signal systems.

- **Signs**
  This communication module is used to transmit data for controlling segment and full matrix signs: Status messages from the sign (status, time stamp, temperature and contents) commands to the sign (contents). A differentiation of the contents is made through: Prismatic signs, full matrix displays, display points and their individual displays. For full matrix displays, transferred are: the complete layout (text and image positions, colors, font etc.) and the contents of the layout elements.

- **Public transport passenger information**
  This communication module is used to transmit road, line, or public transport stop related data of the public transport network: Location, VDV passenger information, public transport stop and line, type of PT vehicle (bus, tram, etc.), expected arrival, actual arrival at the public transport stop, status of the PT vehicle.

- **Traffic signal systems (TSS)**
  This communication module is used to transmit data as input for controlling traffic signal systems: Description, actual status, predefined switchings, parameters for traffic and cycle related signaling.

- **TSS raw data**
  This communication module is used to efficiently transmit a large amount of confidential values from traffic signal light systems: Detector edges, signal group statuses, digital outputs, user program values, PT data (additional R09 telegram), individual detector data.

  Note: All other data relevant to the traffic signal controller that is not mass data / raw data can be found distributed into the other schema definitions.

- **Car-2-X Communication**
  This communication module is used to transfer data for Car-2-X communication between control centers.

- **TSS config data**
  This communication module is used for data supply from traffic signal light systems, based on the OCIT Center to Center standard and OCIT Outstation version 2.

- **Binary container**
  This communication module is used to transmit binary encoded data. Data with a transferred description (name type) identifies the type of data in the container and thus makes it possible for the recipient to process.

- **Project-specific extensions**
  of communication modules are possible, however must correspond with certain rules (see chapter 3.15).

  Note: All data, including big data are transmitted using the OCIT-C protocol.

2 Definitions of objects and schemas

2.1 Object

An object is a unit that can be clearly identified, which the data to be exchanged relates to.

Example: a specific unit such as a TSS, detector, parking garage, traffic message

2.2 Object types

An object type denotes a requestable or configurable unit between the interface partners. It is not possible to request partial quantities of the object types.

In general, data from a data type (e.g. detector values) are divided into the following partial aspects. These partial aspects constitute the object types below - i.e. requestable units:

- Describable data are an object type
- Dynamic data are one or more object types
- Commands are a separate object type
- Object types are defined in the schema definitions. The annotation of the root elements of the data type to be transferred document the identification string of the object type in the form *objecttype: <Objekttypidentifizierung>.*

2.3 Object identification

Object identification occurs over one to three levels:

- Object type
- Source information (source of the object). This step is optional.
- Identifier (id)

The actual identifier must be unique within the object type. If the source is used, then the identifier must be unique within the object type and the source.

The identifiers of an object are generally negotiated, common knowledge between the interface partners (e.g. detectors, signs, parking garages).

Other object types (e.g. traffic messages) must be generated and deleted dynamically via the interface. This way it is not possible to pre-define the identifier for this type of objects. For the traffic messages, the identifiers and source information are used for unique identification.

2.3.1 Object identification

The object identification of the OCIT-C in terms of the identifier is open for all data sources. Therefore, the IDs can be freely definable in principle.

OCIT I/O envisages certain classification criteria for TSS-related data. In order to get a clear illustration of the classification criteria for TSS-related data, it provides a regulation for the formation of OCIT-C IDs, derived from the classification criteria of OCIT I/O. This illustration is especially useful when using OCIT-O as a protocol for connecting outdoor systems.

The ID of a measurement in the Identifier/Ident field, for objects whose measurement is clearly defined by the object type, e.g. traffic data (→ Chapter 3.2), Traffic signal systems (→ Chapter 3.10) and raw data (→ Chapter 3.11) should be structured as follows:

- J<SystemNr>_<SubsystemNr>_<UnitNr >_<ObjektNr>

Whereby "J" stands for the special structure of the primary key. The mark "J" may only be used in conjunction with the format described here.

The address of a measurement in the Id field is not used here and it usually repeats the value of the field Identifier/Ident.[1] [2]

For AP values (→ Chapter 3.11.4), the ID in the Identifier/Ident field should be structured as follows:

- J<SystemNr>_<SubsystemNr>_<UnitNr>_<OITD-Nummer>_<ObjektNr>

Note: The identifiers <SystemNr>, <SubsystemNr>, <UnitNr> and <ObjecktNr> have been taken from the "OCIT-I" standard, as well as the term of the OITD number. OITD numbers are to be written in IP notation (57.102)

If there is no object number available or it is a one-off object, "1" is used as the object number.

For systems which neither use the SystemNo nor the SubsystemNo in the primary key, rather only the UnitNo, the following structure is also permissible:

- J<UnitNr>_<ObjektNr> for traffic data and traffic signal systems

- J<UnitNr>_<OITD-Nummer>_<ObjektNr> for AP vaues

The <OITD-Nummer> should be structured as follows:

- <Member-Nummer>.<Unternummer>

Examples of a complete address are therefore:

- J1_12_22555_17
  for the detector with channel number 17 on intersection 1_12_22555

- J1_12_22555_57.102_3
  for the AP value "102" of the Verkehrs-Systeme AG (Member 57) on intersection 1_12_22555;
  the AP value has the channel number 3

These examples with the short address form appear as follows:

[1] This for example makes it possible to identify an isolated measurement (without the protocol part from protokoll.xsd) in log files.

[2] Alternatively, the data supplier is able to use a local identifier.

- J22555_17
- J22555_57.102_3

2.3.2 Object filter

When calling data, an optional filter list can be specified using the protocol's "inquireAll" and "get" methods. The objects are addressed in the filter list using an identifier in the same way as for object identification. These identifiers are made up of:

- Source information (source of the object). This step is optional.
- Identifier (ident)

It is permissible to use only one part of the primary key used for the identifier in the Identifier/Ident field, whereby the primary key in the server is filtered starting from links.

Example of a filter identifier for the object type TrafficData_detector_currentValue:

- J1_12_22555
  for all detectors on intersection 1_12_22555
- J1_12
  for all detectors of subsystem 12 on system 1
- J1_12_22555_17
  Only the measurements of the detector with channel number 17 on intersection 1_12_22555

If the short form of the address is used, it is only possible to filter by the unit number:

- J22555
  for all detectors on intersection 22555
- J22555_17
  Only the measurements of the detector with channel number 17 on intersection 22555

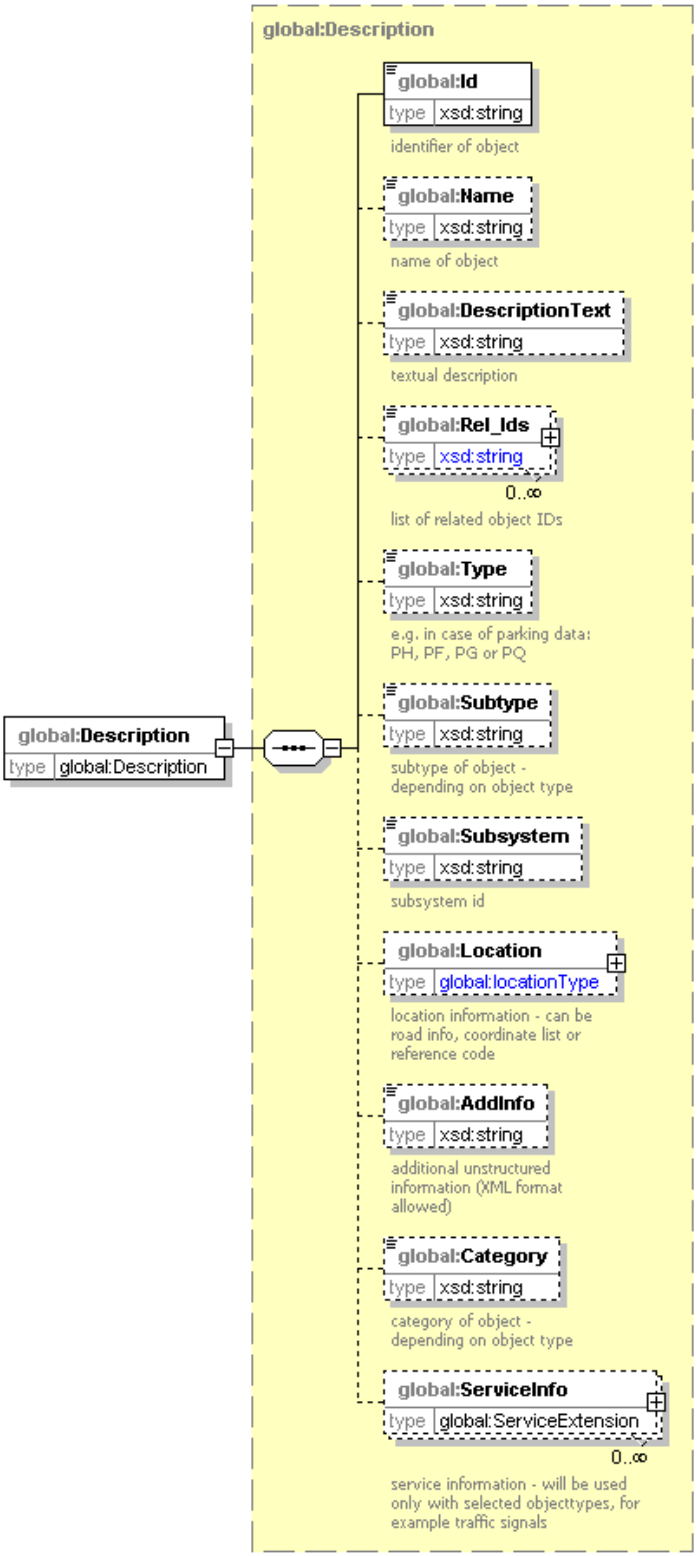
2.4 Global structures (global.xsd)

The exact description of the data model, as well as the elementary description of the attributes and structure elements takes place completely within the individual schema definitions in the form of XML schema definitions (XSD).

The globally used data structures will be defined in global.xsd. A brief description of the structure of global.xsd will be given here.

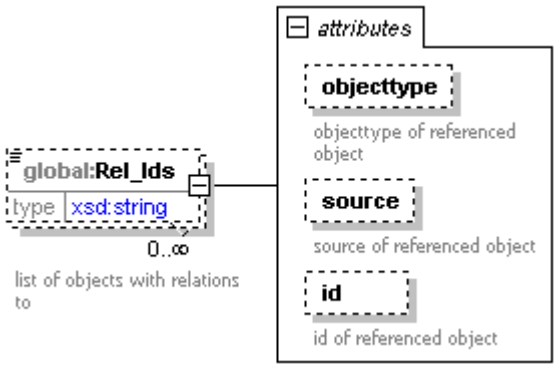
2.4.1 Element Description

All described data are entered into this structure. This data structure is referenced through the domain-specific object types.



2.4.2 Element Rel_Ids

The structure element Rel_Ids allows the referencing or the topology under objects to be described. Therefore, these structure elements refer to other objects and therefore include the complete identification comprising object type, source and ID.

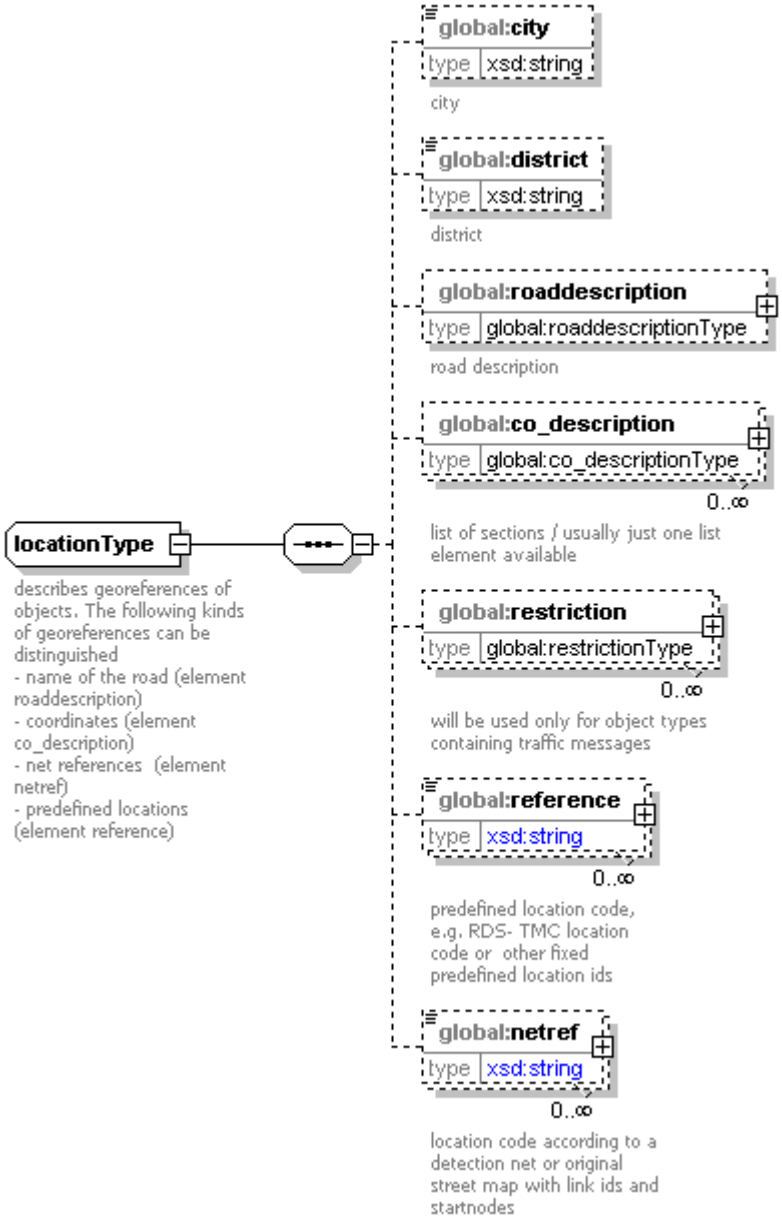


Name	Type	Use	Description
objecttype	xsd:string	optional	Object type of the referenced object
source	xsd:string	optional	Source of the referenced object
id	xsd:string	optional	Identification of the referenced object

2.4.3 Element Location

This structure element allows objects to be geo-referenced. This can take place in different ways:

- Text description of the roads and crossing description
- Coordinates
- Pre-defined geo-references (e.g. TMC locations)
- Reference to networks (link-ids)



Each object description contains this structure. The occupancy must be matched in the respective project.

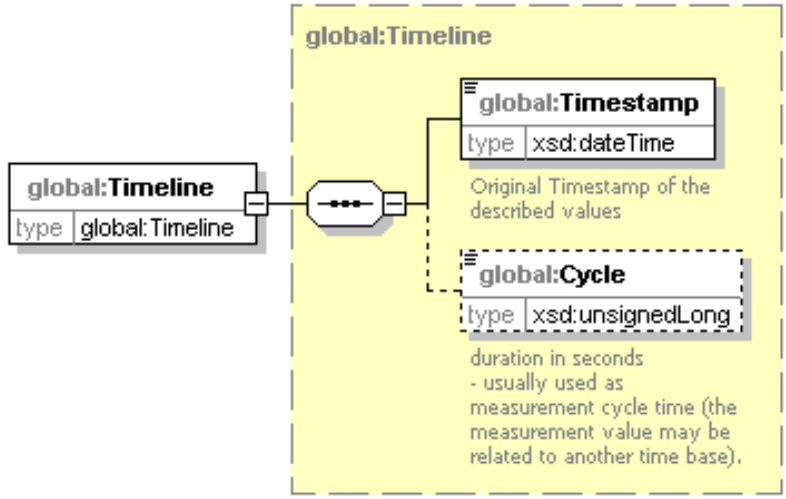
2.4.4 State

Dynamic data have a status. This is displayed by the state. All dynamic data use the globally defined *State.*

Sort	Value	Description
enumeration	o.k.	Field device o.k.: Values can be used
enumeration	n.o.k.	Field device not o.k.: Values cannot be used
enumeration	p.o.k.	Field device partially o.k. There may be a problem when using the values
enumeration	unknown	Field device status is unknown: Values should not be used
enumeration	offline	Field device is not connected.
enumeration	busy	Required value does not match the present value (e.g. since a switching operation is currently taking place).
enumeration	substitute_value	Devicesoftware substitutes a value (e.g. switches to a new device profile)

2.4.5 Timeline

Dynamic data have a time reference. This is displayed by the time stamp. All dynamic data use the globally defined timeline.



3 Data catalogue

The object types described here are defined as data types within OCIT-C.

Graphic portrayals of the schema definitions are used in this document to explain the data structures. Detailed information about this can only be found in the schema definitions. They contain the schema files (.xsd) referenced here in text form.

The schema definitions were written up in English with the exception of *intersection_config_data* (TSS supply data to be predominantly used in German-speaking areas).

All data described here are entered into the protocol.xsd within the "data" element, starting with their root element. The root element is always the element, which is labelled with the comment ("objecttype") within the schema definition in the annotation of the element. The object type listed within is also used as the identifying characteristic of this data type.

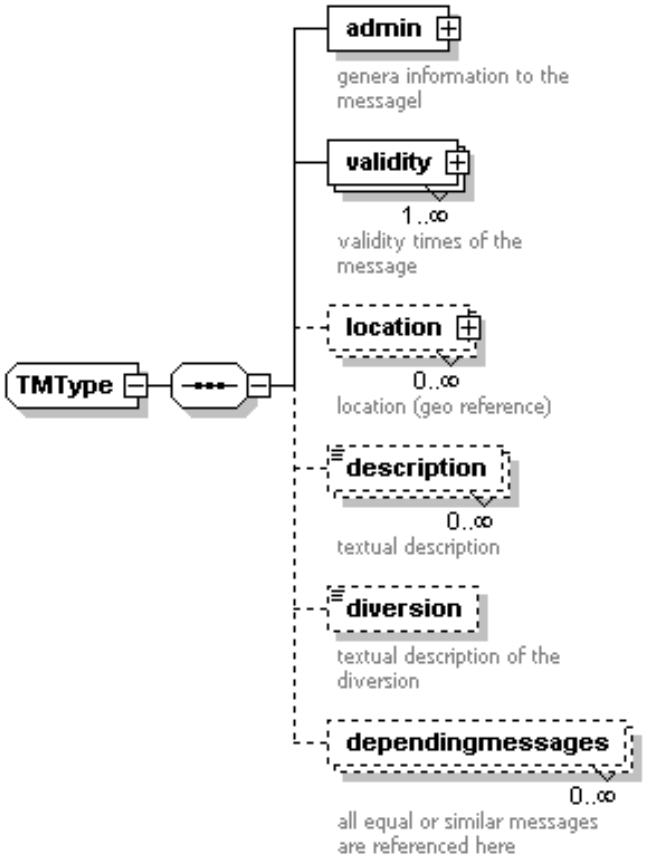
3.1 Traffic messages

This communication module is used to transmit messages regarding roadworks, events, and faults.

Traffic messages are identified using ID and source information.

Available object types:

Object type	Schema file	Short description
TrafficMessage_RoadWorks	traffic_messages.xsd	Roadworks
TrafficMessage_Events		Events
TrafficMessage_Incidents		Faults



3.2 Traffic data

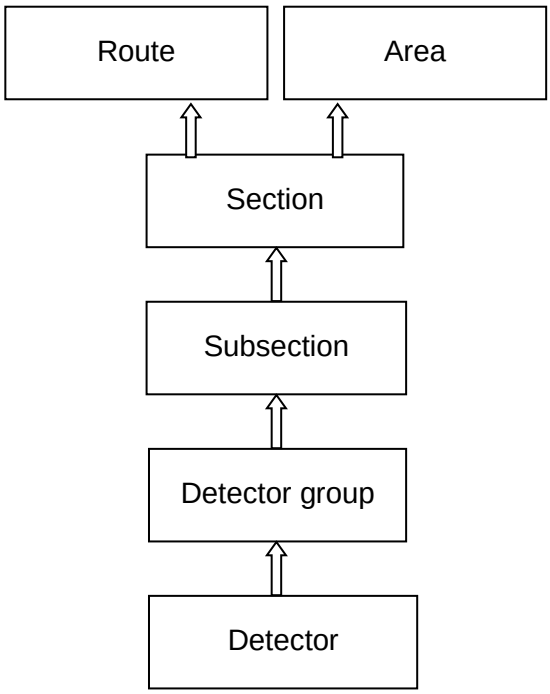
This communication module is used to transmit measurements or derived values from various detection devices. The current and accumulated measurement results from individual detectors or detector groups are transferred, however not detector raw data (impulse).

Available object types:

Object type	Schema file	Short description
TrafficData_detector_Description	Traffic_data.xsd	Descriptive data Detectors
TrafficData_detector_currentValue		Current data from detectors
TrafficData_ detectorGroup_Description		Descriptive data Measurement points
TrafficData_ detectorGroup_currentValue		Current data from a measurement point
TrafficData_ detectorGroup_calculatedValue		Derived data from a measuring point measurement point
TrafficData_ trafficSubSection_Description		Descriptive data Sections

Object type	Schema file	Short description
TrafficData_trafficSection_Description		Data describing sections
TrafficData_trafficArea_Description		Data describing areas
TrafficData_trafficRoute_Description		Data describing routes
TrafficData_ trafficSubSection_accumulatedValue		Dynamic values of subsections
TrafficData_ trafficSection_accumulatedValue		Dynamic values of sections
TrafficData_ trafficArea_accumulatedValue		Dynamic values of areas
TrafficData_ trafficRoute_accumulatedValue		Dynamic values of routes

The following illustration shows the various hierarchies to which the respective object types are assigned.



3.2.1 Subsection, sections, traffic areas

Subsections, sections and traffic areas are referenced using pre-defined IDs.

Recommended identifications:

SS<primary key>_<secondary_key> for subsection
S<primary key>_<secondary_key> for section
R<primary key>_<secondary_key> for routes
TA<primary key>_<secondary_key> for traffic areas

The use of the <primary key> is not absolutely required.

3.2.2 Detectors and detector groups

Detectors (e.g. loops) and measuring points (= detector groups) are identified using pre-defined IDs.

Recommended identifications:

- For detection equipment connected via OCIT-O;
  Object identification in accordance with section 2.3.1

- For any other detection equipment:

  D<primary key>_<secondary_key> for detectors
  DG<primary key>_<secondary_key> for measurement points

The use of the <primary key> is not absolutely required.

Note: To identify detectors supplied via OCIT-C VD, only the primary key formed from the system described in Chapter 3.10.7 is used

3.2.2.1 Object type: TrafficData_detector_Description

The data structure is clarified in Chapter 2.4.1

Use of the element **type**

- Description of the detector type: Loop, double loop, camera, etc.

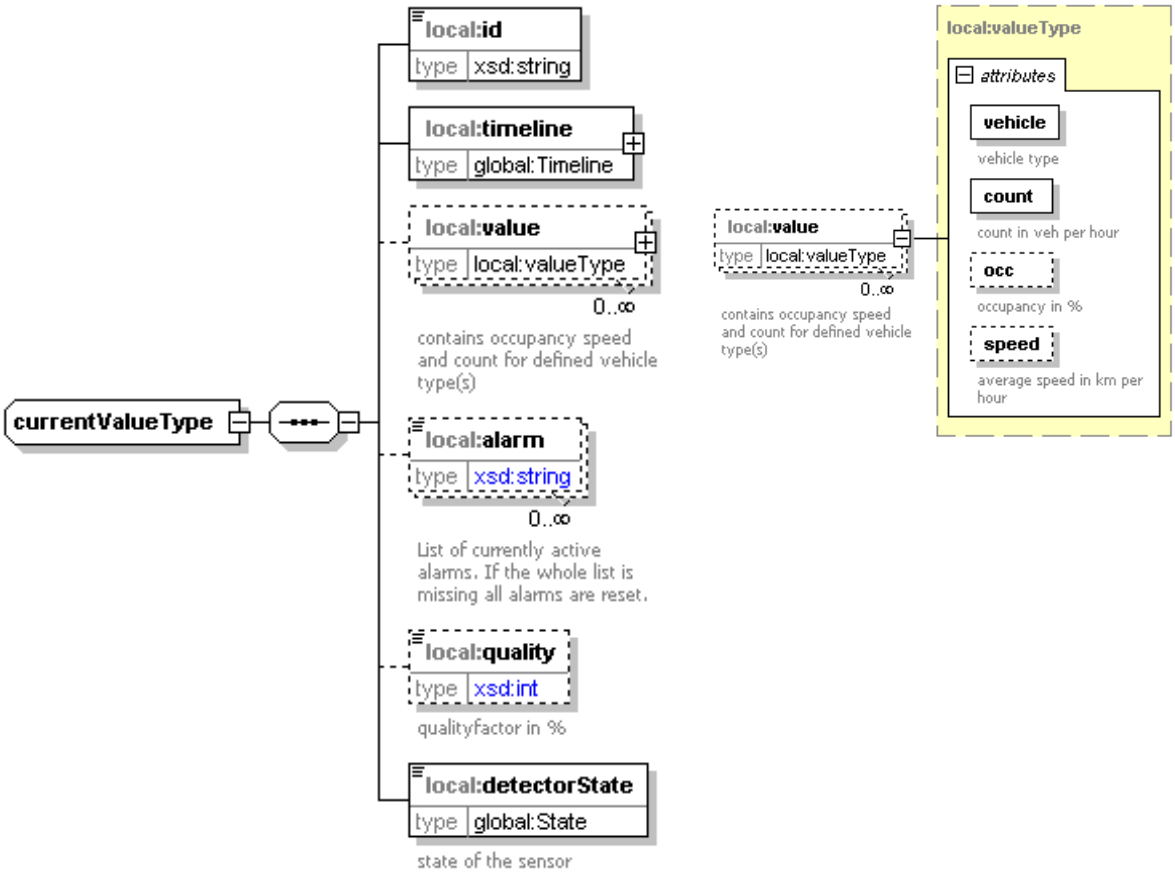
Use element **subsystem**

- Manufacturer/supplier

Use element **Rel_Ids**

- Reference to measuring points
- Reference to TSS

3.2.2.2 Object type: TrafficData_detector_currentValue



The "value" structure is assumed as the field element for each available vehicle type. Types of vehicle which are not available should not be assumed as a field element.

As soon as there are valid values available, the vehicle type "all" is delivered.

Vehicle types:

Vehicle type in OCIT-C	**Equivalence in TLS**
<xsd:enumeration value="all"/>	Sum
<xsd:enumeration value="car"/>	Car
<xsd:enumeration value="truck"/>	Truck
<xsd:enumeration value="artic_truck"/>	Semitrailer truck
<xsd:enumeration value="motorcycle"/>	Motorbike
<xsd:enumeration value="unclassified"/>	"Phantom"
<xsd:enumeration value="car_with_trailer"/>	Car + trailer
<xsd:enumeration value="truck_with_trailer"/>	Truck + trailer
<xsd:enumeration value="bus"/>	Bus

Vehicle type in OCIT-C	Equivalence in TLS
<xsd:enumeration value="delivery_truck"/>	Delivery truck

Alarm list:

"NoVideo", "SpeedAlarm", "OccupancyAlarm", "CommunicationError", "StoppedVehicle", "SpeedVariation", “InverseDirection", "SmokeDetected", "BadVideo", "DigitalInput", "UserAlarm", "ConfigChanged","ZoneActive", "OutputGroupActive", "PhaseActive", "IndiffData", "RebootEvent", "Upgraded", "RecallPressed", "DownloadFailed", "IllegalEvent", "SpeedAlarmLevel1", "SpeedAlarmLevel2", "SpeedAlarmLevel3", "IncidentAlarm", "IncidentWarning"

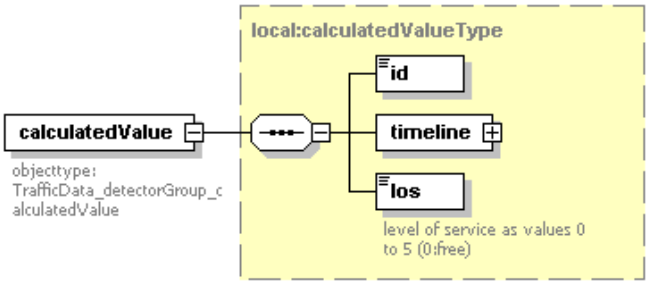
3.2.2.3 Object type: TrafficData_detectorGroup_Description

Used for TrafficData_detector_Description

3.2.2.4 Object type: TrafficData_detectorGroup_currentValue

Used for TrafficData_detector_currentValue

3.2.2.5 Object type: TrafficData_detectorGroup_calculatedValue



3.2.2.6 Object type: TrafficData_trafficSubSection_Description

The data structure is clarified in Chapter 2.4.1

Use element **subsystem**

- Description of the data supplier

Use element **Rel_Ids**

- Description of the assigned detector groups or other data from the traffic data

3.2.2.7 Object type: TrafficData_trafficSection_Description

Used for TrafficData_trafficSubSection_Description

3.2.2.8 Object type: TrafficData_trafficArea_Description

Used for TrafficData_trafficSubSection_Description

3.2.2.9 Object type: TrafficData_trafficSubSection_accumulatedValue

3.2.2.10 Object type: TrafficData_trafficSection_accumulatedValue

Used for TrafficData_trafficSubSection_accumulatedValue

3.2.2.11 Object type: TrafficData_trafficArea_accumulatedValue

Used for TrafficData_trafficSubSection_accumulatedValue

3.3 Parking data

This communication module is used to transmit data from parking facilities: Description and current occupancy of parking facilities[3], parking areas[4], parking districts[5].

Parking data are referenced using pre-defined IDs.

Recommended identification:

PF<primary key>_<secondary_key> for parking facilities
PD<primary key>_<secondary_key> for parking areas and parking spaces

The use of the primary key is not absolutely required.

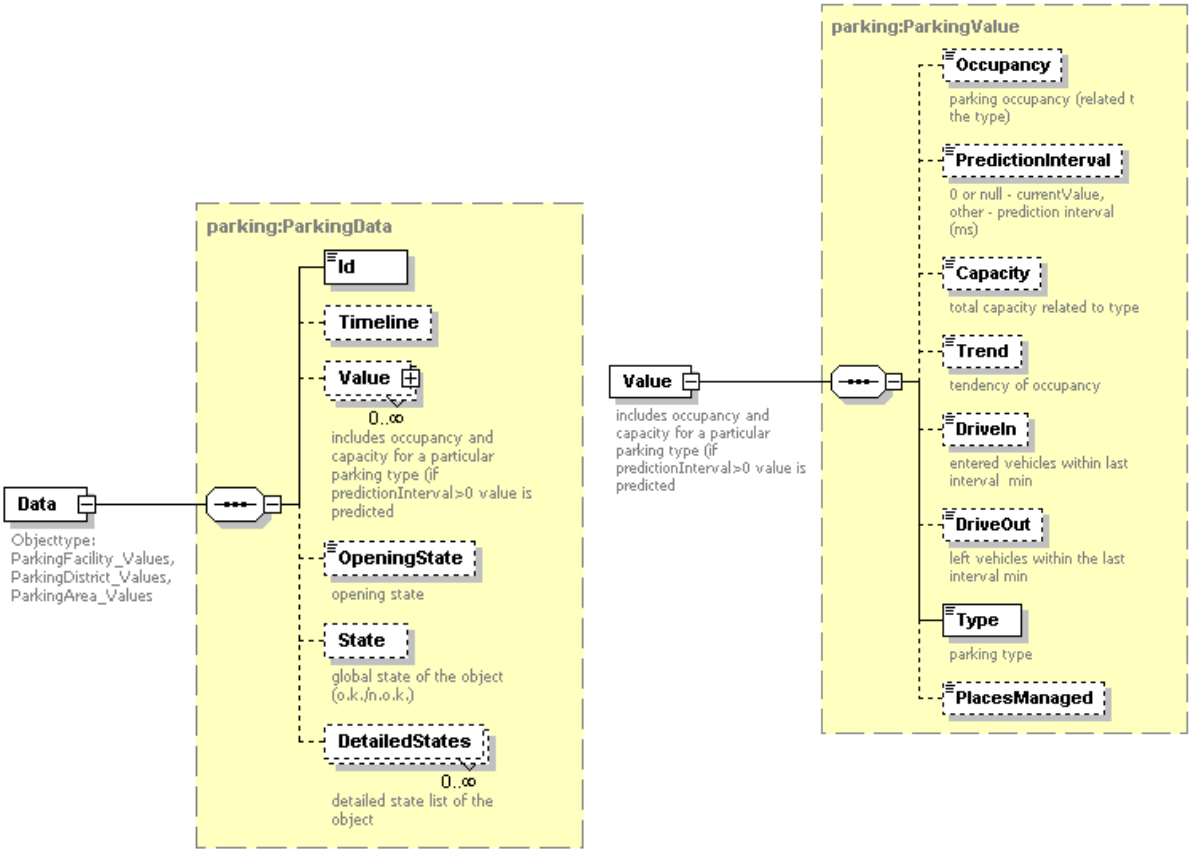
Available object types:

Object type	Schema file	Short description
ParkingFacility_Description	parking.xsd	Description of the parking facility data
ParkingArea_Description		Description of the parking area
ParkingDistrict_Description		Description of the parking district
ParkingFacility_Values		Dynamic data of the parking facility
ParkingArea_Values		Dynamic data of the parking area
ParkingDistrict_Values		Dynamic data of the parking district

[3] ParkingFacility: Individual parking facilities such as multi-storey car parks, underground car parks, restricted parking areas

[4] ParkingArea: Parking areas in the street, e.g. parking license areas with parking ticket machines or similar

[5] ParkingDistrict: Aggregation of ParkingFacilities and/or ParkingAreas, e.g. "Parking in the Old Town"



3.4 Weather and environmental data

This communication module is used to transmit data from respective detection facilities.

Environmental and weather data are identified with pre-defined id's.

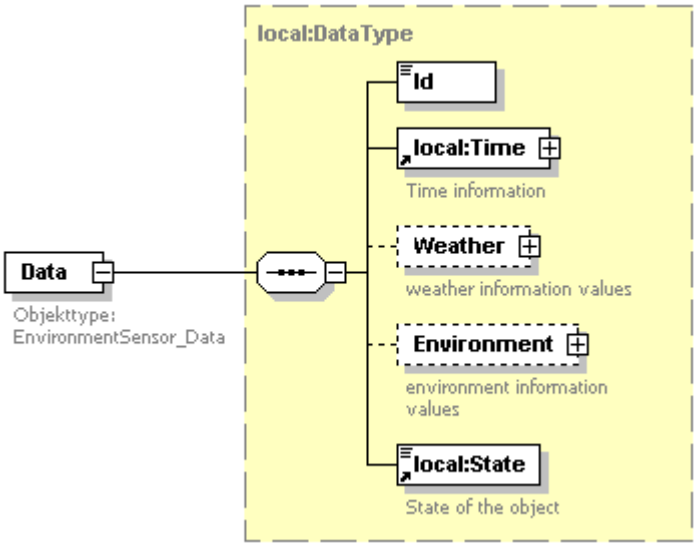
Recommended identification:

W<primary key>_<secondary_key> for weather sensors
E<primary key>_<secondary_key> for environmental sensors

The use of the primary key is not obligatory.

Available object types:

Object type	Schema file	Short description
EnvironmentSensor_Descr	environmentsensor.xsd	Description of weather or environmental data
EnvironmentSensor_Data		Dynamic weather or environmental data



3.5 Cameras

This communication module is used to transmit data to control monitoring cameras and to transfer images.

Camera data are identified with pre-defined id's.

Recommended identification:

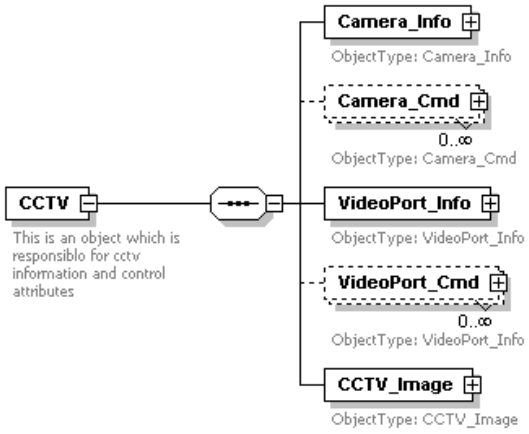
Cam<primary key>_<secondary key>

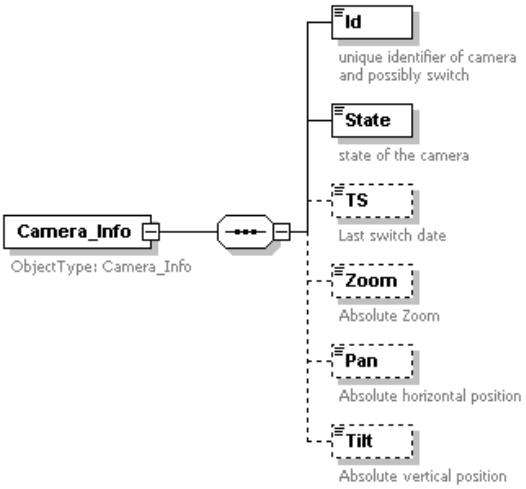
The use of the primary key is not obligatory.

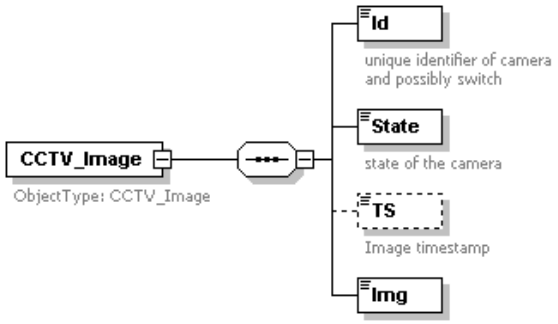
Available object types:

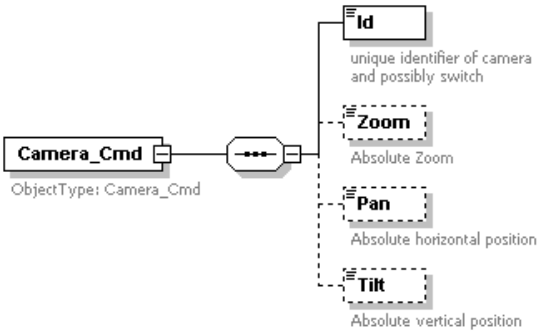
Object type	Schema file	Short description
CCTV_Description	cctv.xsd	Description of the camera data
Camera_Info		Current status of the camera (without image)
CCTV_Image		Current image on a camera

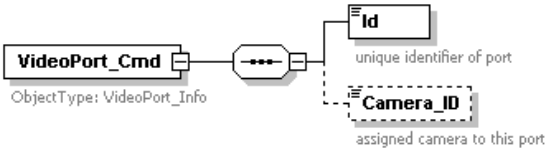
Object type	Schema file	Short description
Camera_Cmd		Command / adjusting command to a camera
VideoPort_Info		Current status of the monitor matrix
VideoPort_Cmd		Command / adjusting command for the monitor matrix

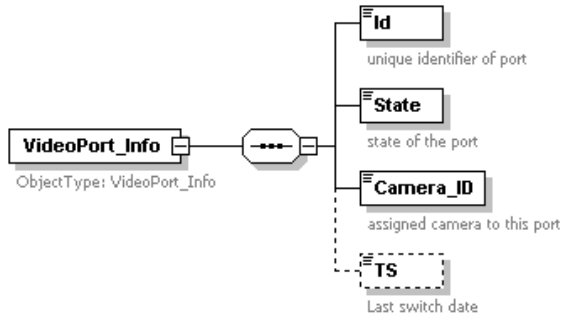












3.6 Situations and strategies

This communication module is used to transmit the descriptions of traffic situations or control strategies and input for strategic traffic control measures.

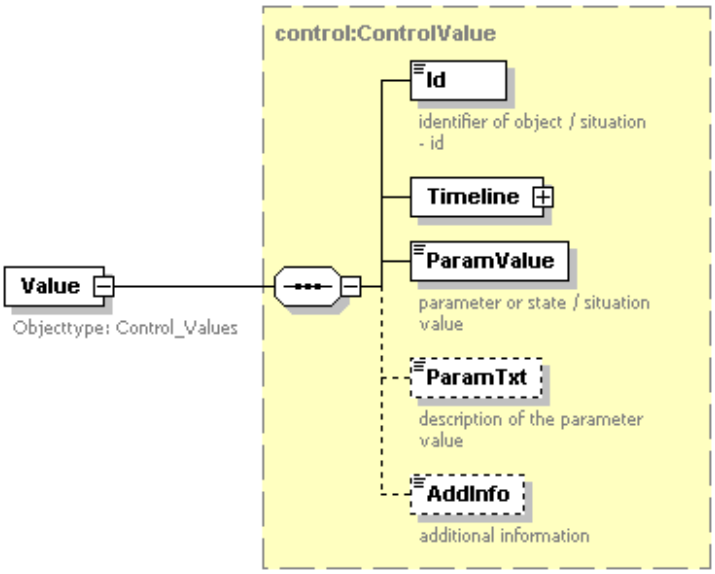
Situations or strategies are identified using pre-defined IDs.

Recommended identification:

Sit<primary key>_<secondary_key>

Available object types:

Object type	Schema file	Short description
Control_Description	Control.xsd	Data describing the situation or strategy
Control_Values		Current strategy/situation



AddInfo can be used for additional information relating to the project.

Application examples:

- Strategy
  The ID identifies the strategy, the variant of the strategy can be modelled using ParamValue.

- Situation.
  The ID identifies the situation (e.g. the status of a set of doors), the status of this set of doors can be modelled using ParamValue (e.g. door open or door closed).

- Digital contact
  The ID identifies the digital contact, the status of this contact can be modelled using ParamValue (e.g. high level or low level).

- Groups
  The ID identifies the group (e.g. the TSS group), the status of this group can be modelled using ParamValue (e.g. group off or group in SP1 or activate group with specific figure).

3.7 Operating messages

This communication module is used to transmit operating messages from traffic equipment, such as from traffic signal systems.

Available object types:

Object type	Schema file	Short description
OperatingMessages_Operatingmessage	operating_messages.xsd	Operating message

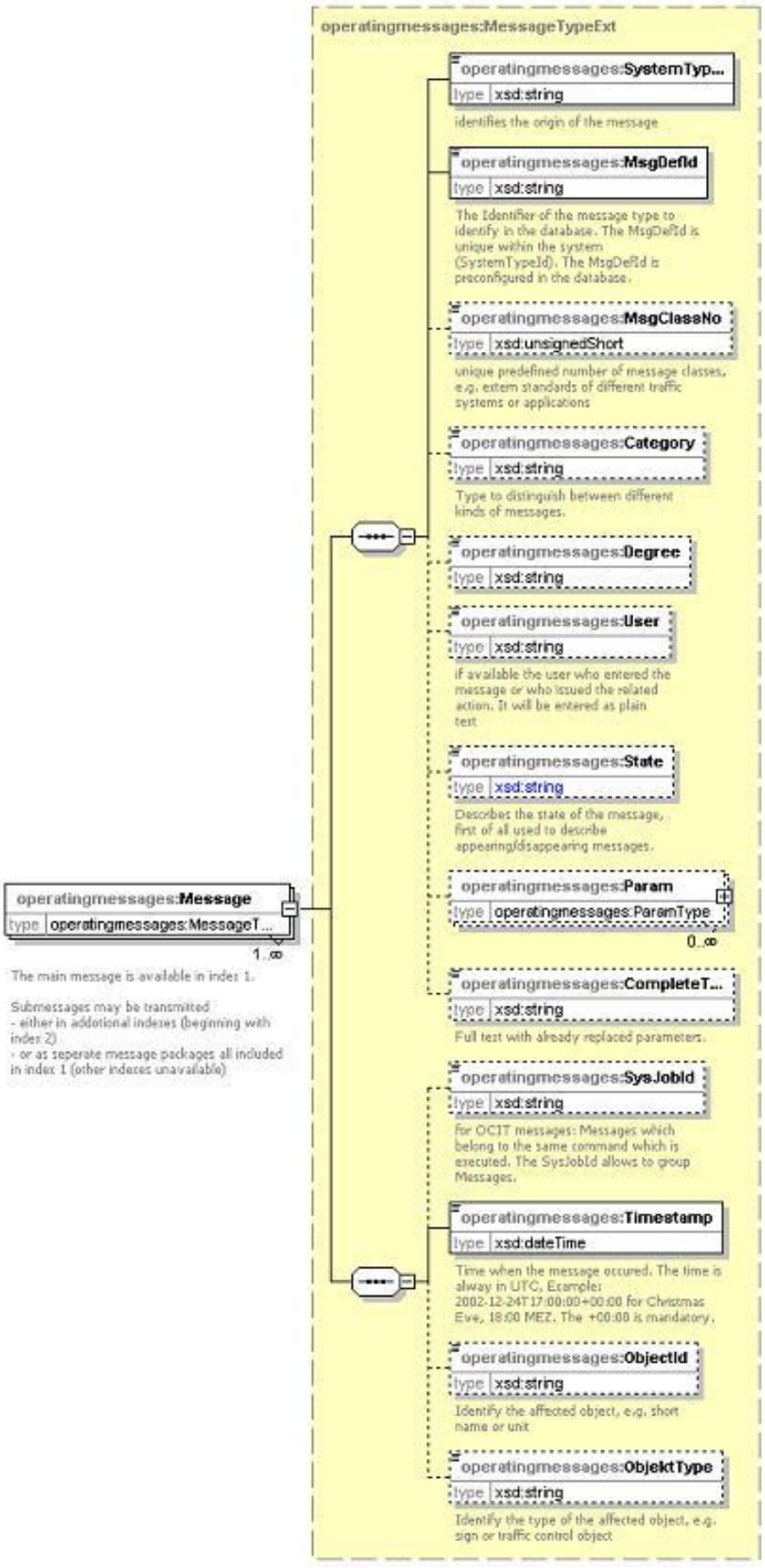
Operating messages have a main message and optional secondary messages. The main message ranks first. Then the secondary messages follow. The elements of a main or a secondary message will be described below.

Identification is done by the object identifier of the object to which this message belongs. A query therefore identifies an object or device (not a message) and can return several different message types (MsgDefId) in the response.

Examples of LSA-related messages (see also: 2.3.1): "J1_12_22555_1"

Example of messages about a sign: "Schild_Bahnhofstr._Nord"

The object identifier of the message must be identical to the object identifier that is also used to address the dynamic data of the object.


operatingmessages:MessageTypeExt
operatingmessages:SystemTyp...
type xsd:string
identifies the origin of the message
operatingmessages:MsgDefId
type xsd:string
The Identifier of the message type to identify in the database. The MsgDefId is unique within the system (SystemTypeId). The MsgDefId is preconfigured in the database.
operatingmessages:MsgClassNo
type xsd:unsignedShort
unique predefined number of message classes, e.g. extern standards of different traffic systems or applications
operatingmessages:Category
type xsd:string
Type to distinguish between different kinds of messages.
operatingmessages:Degree
type xsd:string
operatingmessages:User
type xsd:string
if available the user who entered the message or who issued the related action. It will be entered as plain text
operatingmessages:State
type xsd:string
Describes the state of the message, first of all used to describe appearing/disappearing messages.
operatingmessages:Param
type operatingmessages:ParamType
0..∞
operatingmessages:CompleteT...
type xsd:string
Full text with already replaced parameters.
operatingmessages:SysJobId
type xsd:string
for OCIT messages: Messages which belong to the same command which is executed. The SysJobId allows to group Messages.
operatingmessages:Timestamp
type xsd:dateTime
Time when the message occured. The time is alway in UTC, Example: 2002-12-24T17:00:00+00:00 for Christmas Eve, 18:00 MEZ. The +00:00 is mandatory.
operatingmessages:ObjectId
type xsd:string
Identify the affected object, e.g. short name or unit
operatingmessages:ObjektType
type xsd:string
Identify the type of the affected object, e.g. sign or traffic control object
operatingmessages:Message
type operatingmessages:MessageT...
1..∞
The main message is available in index 1.
Submessages may be transmitted
- either in addotional indexes (beginning with index 2)
- or as seperate message packages all included in index 1 (other indexes unavailable)

Day	Description
SystemTypeID	Origin of the operating message (in OCIT-O, this corresponds to the member number)
MsgDefID	Message type for identification of the operating message. The message type is pre-configured uniquely within the system (SystemTypeField) and within the database (in OCIT-O, this corresponds to the otype).
MsgClassNo	Message class for differentiating between message groups from a manufacturer
Category	Characteristic for determining between various types of messages
Degree	Severity of the message
User	If available, enter the user who entered the message or performed the corresponding action.
State	Describes the status of the message, especially the depiction of the messages.
Param	Parameter for specifying the operating message. The parameter types are pre-defined within the database.
CompleteText	Complete text of the operating message including parameter
SysJobId	If messages are the cause of a certain action, then the action as well as the messages carry the same SysJobId. The SysJobId groups the messages by specific causes / triggers.
Timestamp	Time stamp of when the message arose
ObjectId	Identification of the object in question, e.g.abbreviation
Objekttyp	Identification of the object type in question e.g. sign or traffic control object

You can specify switched and array parameters by filling the corresponding ParamId: "x1", "x2", etc., "y1a", "y1b", "y2a", etc. To do this, a rule must be defined that states how class or field names and array indices shall be defined accordingly, in order to generate a unique ParamId.

3.8 Signs

This communication module is used to transmit data for controlling segment and full matrix signs:

Recommended Identification:

IP<SystemNr>_<SubsystemNr>_<UnitNr>_<ObjektNr>

Available object types:

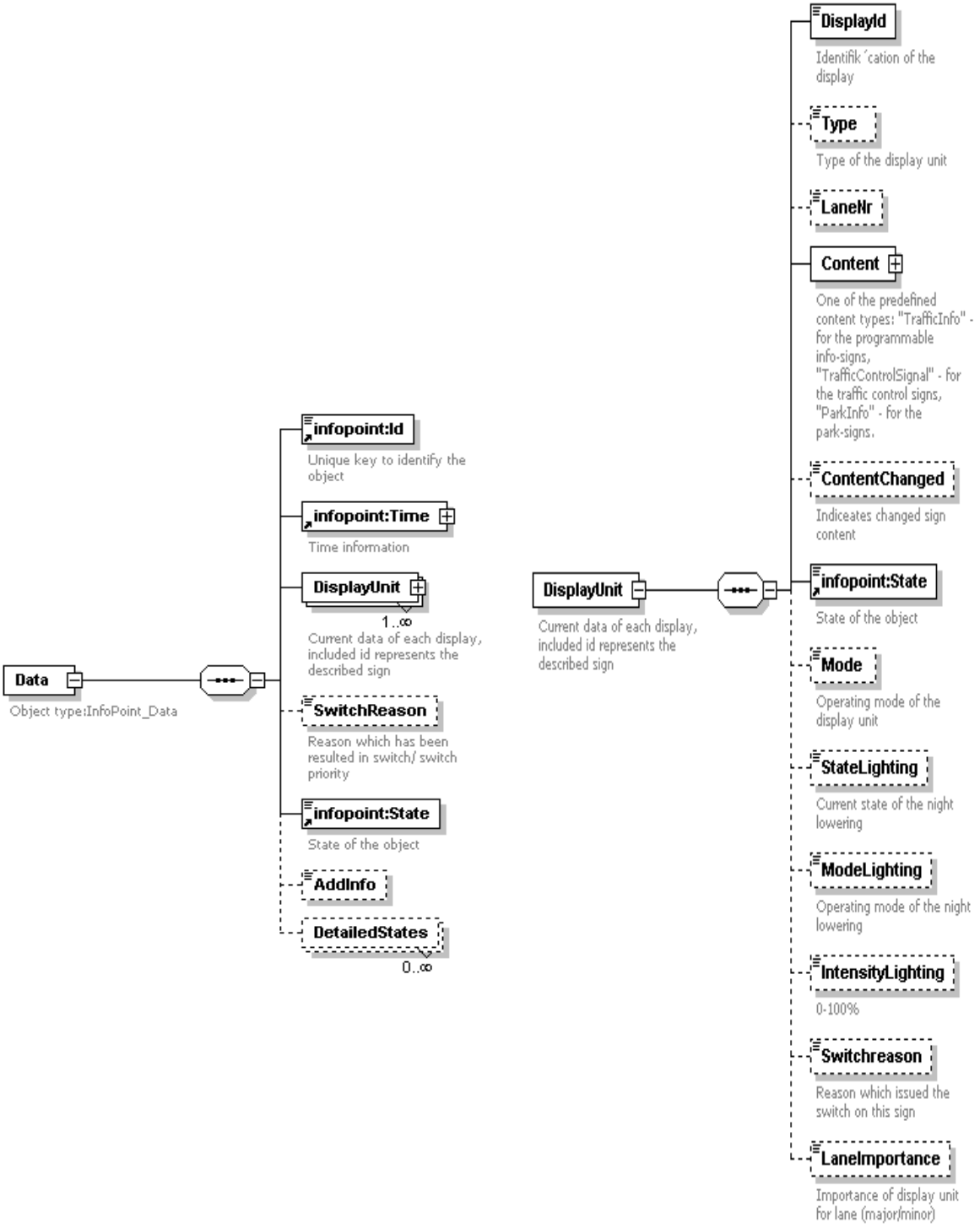
Object type	Schema file	Short description
Infopoint_Description	infopoint_data.xsd	Data describing signs
InfoPoint_Data	(importiert sign.xsd)	Sign contents (current status)
InfoPoint_Data		Sign contents (target status)
Infopoint_Cmd		Alternative adjusting command for signs

The *sign.xsd* is applied in the case of complex matrix signs.

3.8.1 Data model

The data model displayed below is used for exchanging sign contents as well as for activating signs.

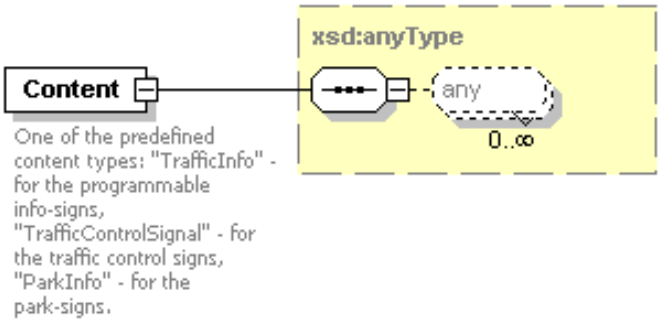
The "state" XML element is only used for exchanging sign contents. This element does not play any role in activating signs.


Data
Object type:InfoPoint_Data
infopoint:Id
Unique key to identify the object
infopoint:Time
Time information
DisplayUnit
1..∞
Current data of each display, included id represents the described sign
SwitchReason
Reason which has been resulted in switch/ switch priority
infopoint:State
State of the object
AddInfo
DetailedStates
0..∞
DisplayUnit
Current data of each display, included id represents the described sign
DisplayId
Identifik´cation of the display
Type
Type of the display unit
LaneNr
Content
One of the predefined content types: "TrafficInfo" - for the programmable info-signs, "TrafficControlSignal" - for the traffic control signs, "ParkInfo" - for the park-signs.
ContentChanged
Indiceates changed sign content
infopoint:State
State of the object
Mode
Operating mode of the display unit
StateLighting
Current state of the night lowering
ModeLighting
Operating mode of the night lowering
IntensityLighting
0-100%
Switchreason
Reason which issued the switch on this sign
LaneImportance
Importance of display unit for lane (major/minor)

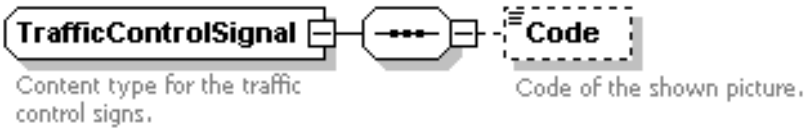
Day	Contents	Description
Id		Identification of the sign
Time	Timestamp	In UTC
DisplayUnit	sub elements	Only occurs - once on full-matrix displays - multiple times on segment displays
DisplayId	-	Labelled segment displays
Type	-	Unused
Content	As defined in sign.xsd	See sign.xsd
ContentChanged	-	Unused
State	-	Unused here
Mode	-	Determination of whether the part of the sign will be controlled centrally or locally
StateLighting		Unused
SwitchReason	-	Unused
State	o.k. n.o.k. busy	Status information from the sign server
	switch	Command display from the sign to the sign server
AddInfo		Additional information (optional).
On_off	on off	

3.8.2 Complex sign contents

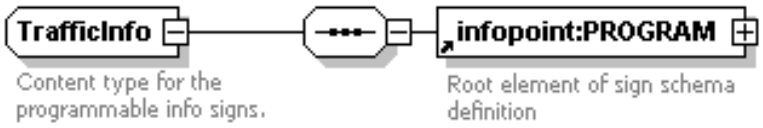
The sign.xsd models complex sign contents. The connection to the infopoint_data.xsd schema is realised using the anyType:



The "any" field can identify the sign content using simple predefined code selection



- a programmable matrix definition or base 64 encoded picture stream

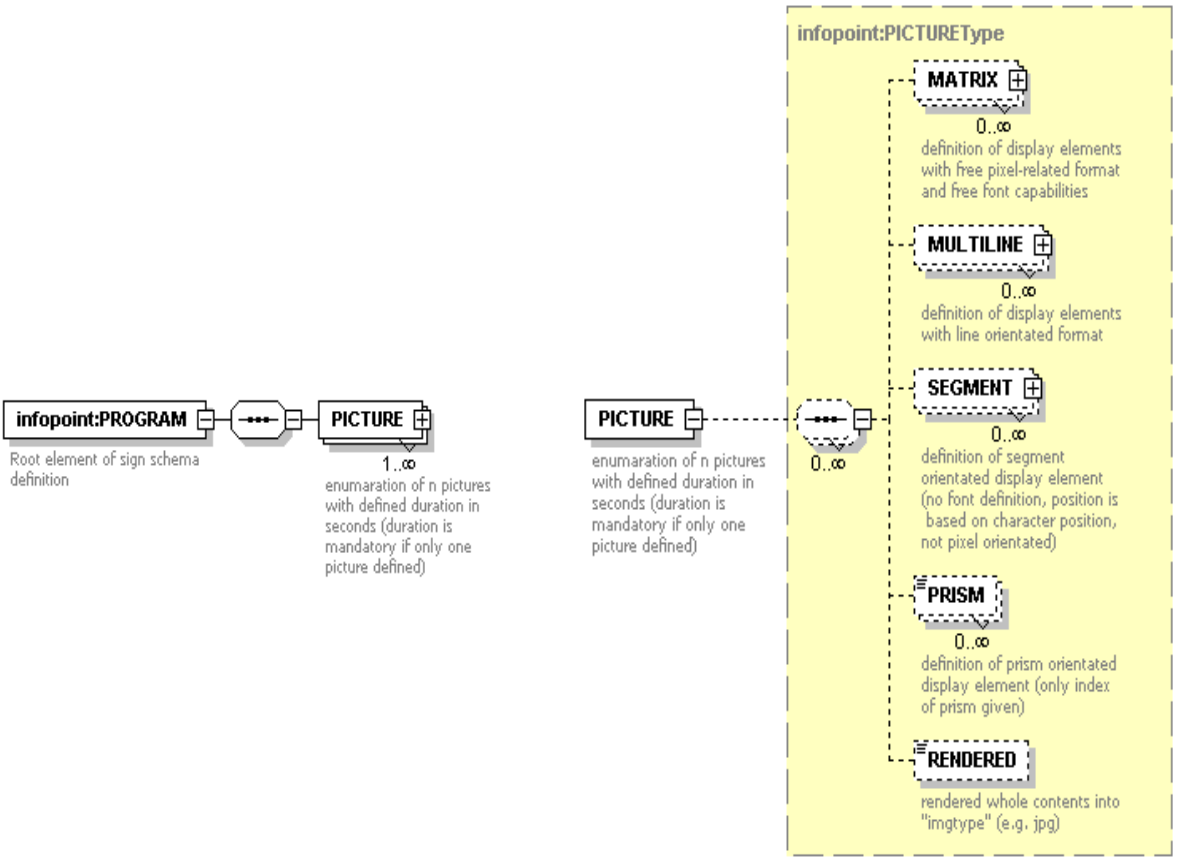


- describe parking information.

The content is a chain of characters e.g. a text, a place holder value or the face of a prism. The ContentType attribute is able to describe the use of the chain of characters in greater detail.

- Programme Definition

  allows the definition of frame sequences:



3.9 Public transport passenger information

This communication module is used to transmit road, line, or public transport stop related data of the public transport network.

Publictransport_data are referenced using pre-defined IDs.

Recommended identification:

PT<primary key>_<secondary_key> for junctions

The use of the primary key is not absolutely required.

Available object types:

Object type	Schema file	Short description
PT_Description	publictransport_data.xsd	
PT_Link		Passenger information relating to the road: Number, speed and delay
PT_LinePerLinkLink		Passenger information relating to the line: Number, speed and delay

3.10 Traffic light signal systems

This communication module is used to transmit data as input for controlling traffic signal systems.

Intersection_data are referenced using pre-defined IDs.

Recommended identification:

J<primary key>_<secondary_key> for junctions

The use of the primary key is not absolutely required.

Available object types:

Object type	Schema file	Short description
IntersectionDescription	intersection_data.xsd	Data describing TSS
IntersectionStatus		Current TSS data (actual status)
PredefCommand		Pre-defined TSS switching operations
IntersectionFeature		Ability of a TSS to perform a switching operation
IntersectionCommand		TSS switching operation (modelled out as a replacement of the PredefCommand)
ControlMethodParameters		Dynamic parameters for traffic-related processes
SGTimes		Summarised red and green times for a cycle.

3.10.1 **Pre-defined commands** - obsolete

With the introduction of modelled commands, pre-defined commands have become superfluous. They will still be used during the transition period for compatibility reasons.

The following pre-defined commands are available:

Command	**Configuration of the XML element**
Knoten ein/aus	on off local
SPx (Knoten wird implizit eingeschaltet) x ist die Programmnummer	plan1 plan2 .... (planx) local plan
IV ein/aus	it_on it_off it_local
ÖV ein/aus	pt_on pt_off pt_local
Lokale VA ein/aus	localta_on localta_off localta_local
Teilknoten ein/aus	Subintersection0_on; subintersection0_off; subintersection0_local subintersection1_on; subintersection1_off; subintersection1_local subintersection2_on; subintersection2_off; subintersection2_local subintersection3_on; subintersection3_off; subintersection3_local
Sonderkommandos	specialcommand<number> localspecialcommand<number>
Projektspezifische Modifikation	projectspecificcommand_<name or number> localprojectspecificcommand_<name or number>
Freigabe	release

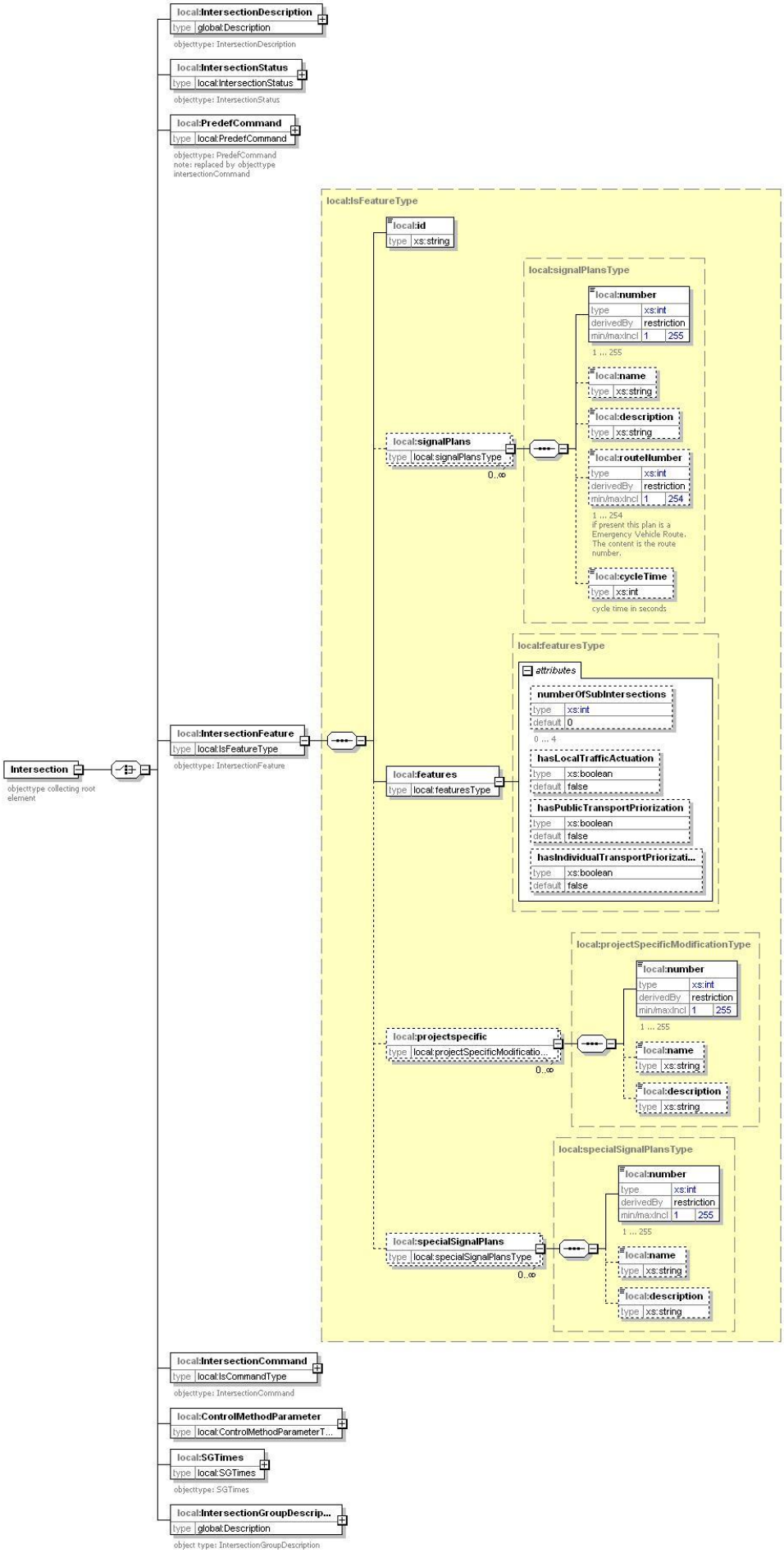
Further commands relating to the project can be added as part of the project.

Should multiple pre-defined commands be executed simultaneously at the same point in time, these must be executed within a put (within the put-List under the same object ID). The server interprets these such that these can only lead to one switchover in the controller.

3.10.2 Intersection Features

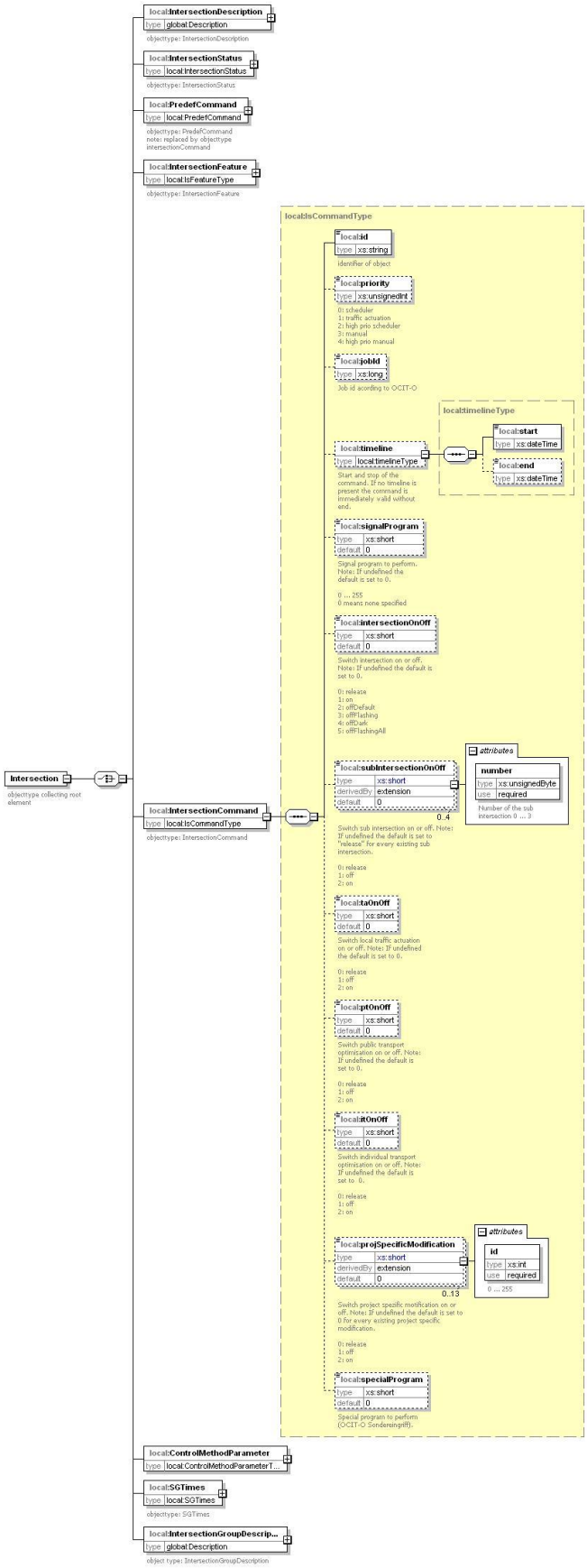
The IntersectionFeatures object type allows you to see the switching options of a TSS (e.g. which plans can be switched to).

This makes it possible to limit the selection of switching operations, before they are sent via the Intersection-Command and in this regard declined.


Intersection
objecttype collecting root element
local:IntersectionDescription
type global:Description
objecttype: IntersectionDescription
local:IntersectionStatus
type local:IntersectionStatus
objecttype: IntersectionStatus
local:PredefCommand
type local:PredefCommand
objecttype: PredefCommand
note: replaced by objecttype intersectionCommand
local:IntersectionFeature
type local:IsFeatureType
objecttype: IntersectionFeature
local:IsFeatureType
local:id
type xs:string
local:signalPlans
type local:signalPlansType
0..∞
local:signalPlansType
local:number
type xs:int
derivedBy restriction
min/maxIncl 1 255
1 ... 255
local:name
type xs:string
local:description
type xs:string
local:routeNumber
type xs:int
derivedBy restriction
min/maxIncl 1 254
1 ... 254
if present this plan is a Emergency Vehicle Route. The content is the route number.
local:cycleTime
type xs:int
cycle time in seconds
local:features
type local:featuresType
local:featuresType
attributes
numberOfSubIntersections
type xs:int
default 0
0 ... 4
hasLocalTrafficActuation
type xs:boolean
default false
hasPublicTransportPriorization
type xs:boolean
default false
hasIndividualTransportPriorizati...
type xs:boolean
default false
local:projectspecific
type local:projectSpecificModificatio...
0..∞
local:projectSpecificModificationType
local:number
type xs:int
derivedBy restriction
min/maxIncl 1 255
1 ... 255
local:name
type xs:string
local:description
type xs:string
local:specialSignalPlans
type local:specialSignalPlansType
0..∞
local:specialSignalPlansType
local:number
type xs:int
derivedBy restriction
min/maxIncl 1 255
1 ... 255
local:name
type xs:string
local:description
type xs:string
local:IntersectionCommand
type local:IsCommandType
objecttype: IntersectionCommand
local:ControlMethodParameter
type local:ControlMethodParameterT...
local:SGTimes
type local:SGTimes
objecttype: SGTimes
local:IntersectionGroupDescrip...
type global:Description
object type: IntersectionGroupDescription

3.10.3 IntersectionCommand (modelled switching operations)

The IntersectionCommand object type allows you to trigger switching operations.



3.10.4 Control parameter (**ControlMethodParameter**)

The recipient of the dynamic parameters is a traffic-procedure within the traffic signal system.

The content is therefore completely determined by the device function, e.g. the VA procedure, and must be coordinated between the device manufacturer and the sending OCIT-C client.

Identification: J <SystemNo> <SubsystemNo> <UnitNo> <ObjectNo>

Example: "J1_12_22555_1" (identifies a device, not an ApWert)

See: 2.3.1

Parameter	**Procedure**
path	Name of the parameter set for transparent distribution to the device, e.g. via OCIT-O, under which the device expects the APWertBlock (BLOB). Examples: "61,200", "VDK", "DPV1", "VS-PLUS.NETZSTEUERUNG "
data	Binary content of the ApWert block, transparently forwarded to the device, e.g. via OCIT-O.
id	Optional designation of the device
methodname	Optional name of the control method to receive the data, e.g. VS-PLUS
interfacename	Optional name of the interface in the control unit, e.g. NETZSTEUERUNG

Feedback can be give via suitable operating messages, that can be agreed upon on a project or process-specific basis.

3.10.5 TSS status (IntersectionStatus)

Detailed information about the status of the TSS can be found under "faultStatus", if supported by the device:

TextNo: A number which labels the content

Text: Optional text entry

Params: Parameters for further details

Default: Value used by the client if the TextNr was not sent by the device.

TSS State			
TextNo	**Text**		**Description**
1001	Time source		Specifies the current time source of the device time. Default: 5
	Params		
	0	Line voltage	
	1	Quartz clock	
	2	LAN	
	3	DCF	
	4	GPS	
	5	Unknown	
1002	EmergencyOff		The EMERGENCYOFF-switch of the device has been activated.
1003	DoorOpen		The door is open.
1004	LineVoltageOK		The mains voltage in the device is OK.
1005	PersistenceStorageOK		Information about whether the persistent storage is in order
1006	Lamp fault		The specified lamps are faulty.
	Params	<Relknoten><Signalgruppe><Signalgeber><Kammer> e.g. 1-8-2-3	
1007	DetectorFault		The specified detectors are disturbed.
	Params	ChannelNumber One parameter per faulty detector e.g. "5"	
1008	CollectiveFault		Collective fault from OCIT-O is listed under faultState.
	Params	The values: • NoFault, • Fault, • FaultWithShutoff, • FaultWithPartialShutoff, • InternalFault	
1009	ConnectionLoss		Information that the connection between the control panel and the traffic light is interrupted
1010	PTReception		Indicates whether the public transport receiver is working.
	Params		

	0	PTReception configurated and OK	Default: 255
	1	Too many CRC errors in received telegrams	
	2	Receiver unit malfunctioned or no communication	
	3	Radio receiver unit disturbed or no noise	
	4	Have not received valid public transport telegrams for a long time (time-out time can be configured on a device-specific basis)	
	254	No PTReception configured	
	255	Zustand unbekannt oder nicht lieferbar	
1011	ReferenceTime		Specifies whether the device is in sync with the reference time. Default: 255
	Params		
	0	Device TX runs in sync with reference time	
	1	Device is out of sync with reference time	
	2	Device does not run in synchronized mode	
	254	No recalculation method configured, i.e. device does not have to run time-synchronized	
	255	Condition unknown or out of stock	
1012	CentralCommands		Specifies whether the device would currently respond to control panel switching commands. Default: 255
	Params		
	0	Device accepts switching requests from the control panel	
	1	Device does not accept switching requests from the control panel (e.g. operating mode locally fixed)	
	255	Condition unknown or out of stock	
1013	UPS		Indicates whether the UPS is installed in the unit and is working. Default: 255
	Params		
	0	UPS is OK, battery has enough voltage	
	1	UPS is defective or battery is empty or defective	
	254	No UPS configured in the unit	
	255	Condition unknown or out of stock	

1014	Time Source / Time Synchronization		Specifies whether at least one of the configured time sources is available for time synchronization. Default: 255
	Params		
	0	At least one of the defined time sources for time synchronization is accessible and works	
	1	None of the configured time sources are reachable or working	
	254	No external time sources configured in the device	
	255	Condition unknown or out of stock	
1015	RSUCondition		Specifies the state of the RSU. Default: 255
	Params		
	0	RSU configured and OK	
	1	Communication unit disrupted or no communication	
	2	Certification system disrupted or no valid certificate chain	
	3	Internal malfunction, e.g. supply problem	
	4	Internal communication malfunction between RSU and ECU/OCIT attachment	
	254	No RSU configured	
	255	Condition unknown or out of stock	
1016	MaintenanceStatus		Indicates whether the device is in a maintenance state Default: 255
	Params		
	0	Device is not in maintenance state	
	1	Device is in a maintenance state	
	254	Maintenance status is not supported by the device	
	255	Maintenance status unknown or unavailable	

3.10.6 SGTimes

Red and green times for signal groups are summarised relating to the cycle in the SGTimes object type. The addressing is done at the TSS level, i.e. it is not possible to activate signal groups individually within this object type.

3.10.7 IntersectionGroupDescription

The TSS group assignment is defined in the object type IntersectionGroupDescription.

Recommended identifications:

JG<SystemNr>_<SubsystemNr>_<GroupNr>

JG<GroupNr>

Use of the element **type**

- Description of TSS group type: e.g. group, district

Use element **Rel_Ids**

- Reference to TSS
- Reference to TSS group

3.11 TSS raw data

This communication module is used to efficiently transmit a large amount of confidential values from traffic signal light systems.

The data to be transferred are grouped by statuses. Each status assumed is listed once and assigned to the point in time at which it was assumed. All time stamps from a certain time interval contain a start time (xsd:element name="timeline") for the interval and the offset until the start time. The offset can be depicted as a product made up of a number and a time unit (xsd:element name="intervalLength"). This number is saved as an unsigned 16 bit value. On the one hand, this results in a maximum interval length, on the other hand, it is possible to list all 16 bit values successively without separate XML elements. The XML representation for the field of 16 bit values is then its Base64 coding (xsd:element name="Events").

Recommended identification:

- Object identification in accordance with section 2.3.1

Available object types:

Object type	Schema files	Short description
RawTrafficDataBlock_ Detectoredge	intersection_raw_data.xsd	Detector edges
DigOut_Raw_Values		Digital outputs
RawTrafficDataBlock_ Signalgroupvalue		Signal group values
NamedValue_Raw_Values		AP values
NamedValuesOrderList	intersection_raw_data.xsd	AP values Order
PublicTransportTelegrammExtended	Public_ Transport_ExtendedTelegram.xsd (importiert Public_Transport_Telegram.xsd)	Advanced PT telegram
DetectorExtValueType	detector_ext.xsd	Individual detector data

3.11.1 Detector edges

The detectorEdges object (object type RawTrafficDataBlock_Detectoredge) has been defined for transferring the detector edges.

The value (xsd:string) is set with '0' for falling edges and '1' for rising edges.

For example:

Detector had the "occupied" status at the following times (rising):

1. 2011-03-23T14:20:00.100+01:00,
2. 2011-03-23T14:20:01.200+01:00,
3. 2011-03-23T14:20:02.000+01:00.

Detector had the "not occupied" status at the following times:

4. 2011-03-23T14:20:00.300+01:00,
5. 2011-03-23T14:20:01.800+01:00,
6. 2011-03-23T14:20:02.300+01:00.

The data are to be entered as follows:

ID of the detector: `id: Det_1`

Starting time: `timeline/Timestamp: 2011-03-23T14:20:00+01:00`

Time unit in milliseconds: `intervalLength: 100`

Rising edges: `data/Value: 1`

Events:

Calculation rule: (Event time - start time) / time unit == number

`(2011-03-23T14:20:00.100 – 2011-03-23T14:20:00.000) / 100 == 1`

Number is saved as an unsigned 16 bit value.

`1: 100 /100 == 0x0001`

`2: 1200 / 100 == 0x000c`

`3: 2000 / 100 == 0x0014`

All 16 bit values should be entered successively.

Byte1 == 0x00

Byte2 == 0x01

Byte3 == 0x00

Byte4 == 0x0c

Byte5 == 0x00

Byte6 == 0x14

data/Events: AAEADAAU (0001000c0014 coded in Base64)

Falling edges: data/Value: 0

Events:

4: 300 / 100 == 0x0003

5: 1800 / 100 == 0x0012

6: 2300 / 100 == 0x0017

`data/Events: AAMAEgAX (000300120017 coded in Base64)`

3.11.2 Signal group states

The sgValues object (object type RawTrafficDataBlock_Signalgroupvalue) has been defined for transferring the signal group statuses.

The value (xsd:string) is defined as an integer. The value should be used identically to OCIT-O object signal aspects 1:611 (e.g. 3 for red, 48 for green).

If the value is set to "3", the point in time at which the signal group switches to "red" is entered.

If the value is set to "48", the point in time at which the signal group switches to "green" is entered.

For example:

Signal group had the "red" status at:

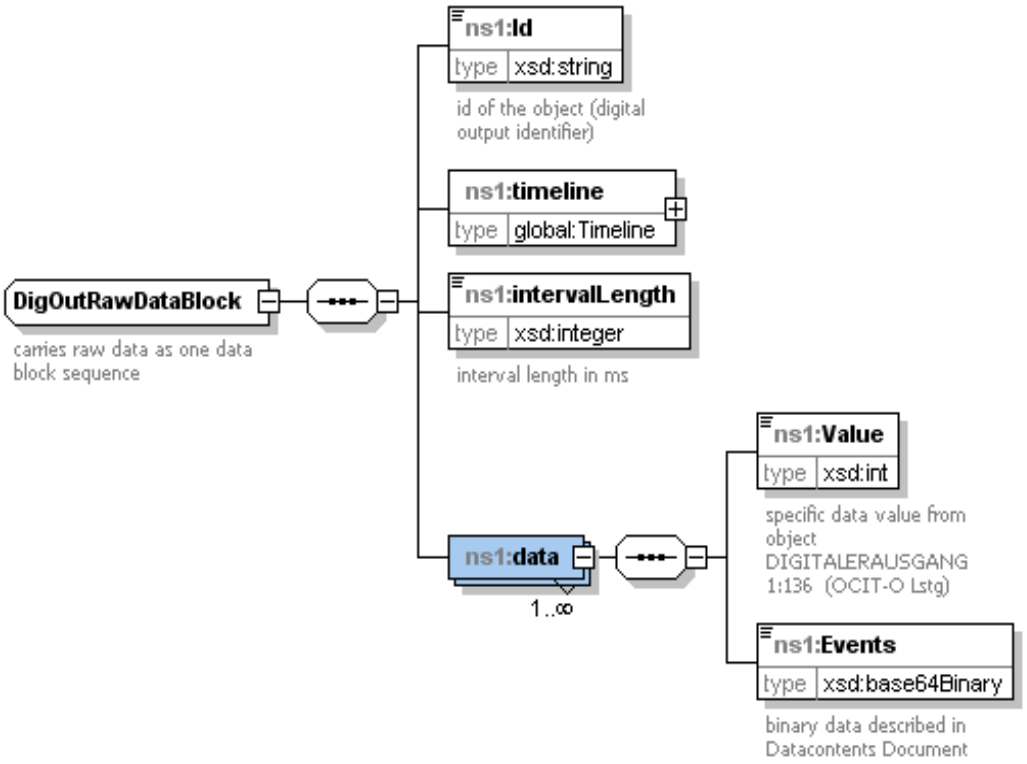
1. 2011-03-23T14:20:10.000+01:00,
2. 2011-03-23T14:21:10.000+01:00,
3. 2011-03-23T14:22:10.000+01:00.

The data are to be entered as follows:

```
id: Sg_1

timeline/Timestamp:  2011-03-23T14:20:00+01:00

intervalLength: 1000

data/Value: 3

Events:

1: 10000 / 1000 == 0x000a

2: 70000 / 1000 == 0x0046

3: 130000 / 1000 == 0x0082

data/Events: AAoARgCC (000a00460082 coded in Base64)
```

3.11.3 Digital outputs in the traffic signal controller

The object DigOutRawType (object type DigOut_Raw_Values) has been defined for transferring the statuses of digital outputs.



The status has been defined as an integer type. The value has been adopted from the OCIT-O object DIGITALERAUSGANG 1:136, e.g. 3 for one, 1 for flashing start off.

If the value is set to "3", the point in time at which the digital output switches to the "on" state is entered.

If the value is set to "1", the events for which the digital output switches to the "Flashing start off" state is entered.

For example:

Digital output has the "on" status at:

1. 2011-03-23T14:20:00.100+01:00,
2. 2011-03-23T14:20:01.200+01:00,
3. 2011-03-23T14:20:02.000+01:00

The data are to be entered as follows:

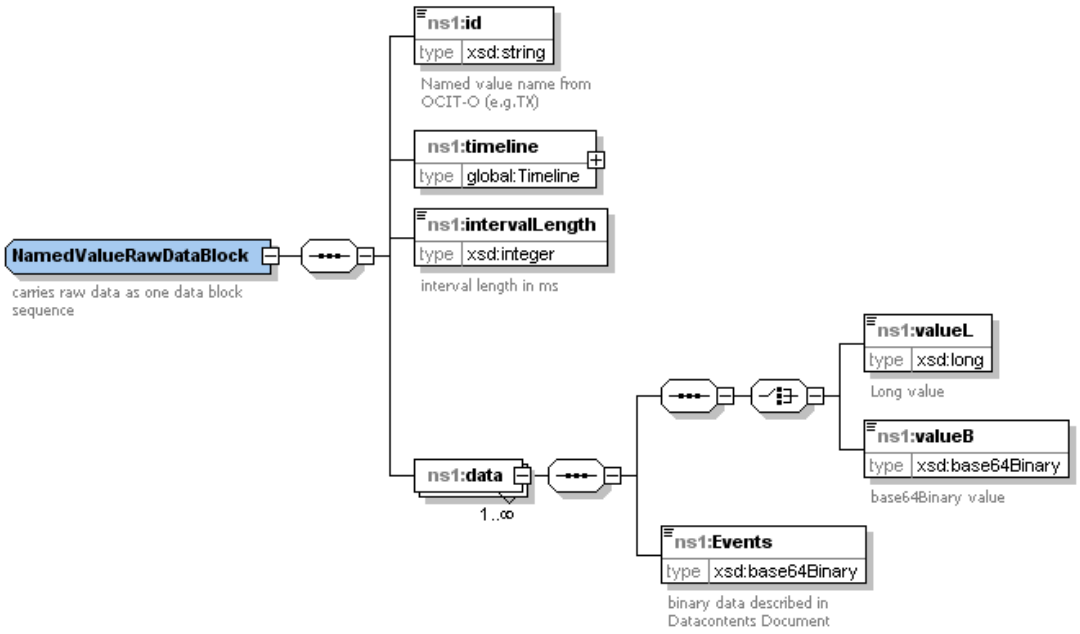
```
id: Dout_1

timeline/Timestamp:  2011-03-23T14:20:00+01:00

intervalLength: 100

data/Value: 3

Events:
```

```
1: 100 / 100 == 0x0001

2: 1200 / 100 == 0x000c

3: 2000 / 100 == 0x0014

data/Events: AAEADAAU (0001000c0014 coded in Base64)
```

3.11.4 User program values (AP-values)

The NamedValueRawType object has been defined for transferring the status of AP values.

AP values are identified via an identification string. The string is defined in the XML schema intersection_config_data_ap_values.xsd (see OCIT-C_LSA_Versorgungsdaten) under OITDdef/OITDdef/Verfahren/OITD/Bezeichner/KurzBez.



The state has been defined as a selection between the long and the base64Binary types. If AP values of the types:

long, integer, short, byte should be transferred, valueL is set.

If a BLOB type AP value should be transferred (cf. OCIT-O object APValueBlock 1:508), valueB is set. The binary data from the BLOB type are transferred (BYTE data[]).

OCIT-C defines the following standardised AP values:

OITD IP notation	OITD4-No.	OITD2-No.	Name	Description
41.94	2687070	42078	TX	**The cycle second of the running signal program**

				is counted beginning with second 0 in 100ms increments. 65535: Value undefined
41.96	2687072	42080	PH	**Current stage in the signal program** 0: Stage not active or not defined in the process 1 - 65534 current stage number
41.97	2687073	42081	UE	**Requested stage** 0: No stage transition active 1 - 65534: Transition active from stage PH to stage UE
41.98	2687073	42082	DPV1	Dynamic Parameters for a Traffic Engineering Procedure

Other names for the AP values are to be agreed upon on a project-specific basis.

Examples:

- Cycle second (TX) has the state "10" at:
  1. 2011-03-23T14:20:10.000+01:00,
  2. 2011-03-23T14:21:10.000+01:00,
  3. 2011-03-23T14:22:10.000+01:00.

The data are to be entered as follows:

```
id: TX
timeline/Timestamp:  2011-03-23T14:20:00+01:00
intervalLength: 1000
data/Value: 1
Events:
1: 10000 / 1000 == 0x000a
2: 70000 / 1000 == 0x0046
3: 130000 / 1000 == 0x0082
data/Events: AAoARgCC (000a00460082 coded in Base64)
```

- AP value block (ABValueB) has the state "BYTE data[01, 05, 0c, a2]" at:
  1. 2011-03-23T14:20:10.000+01:00,
  2. 2011-03-23T14:21:10.000+01:00,
  3. 2011-03-23T14:22:10.000+01:00.

The data are to be entered as follows:

```
id:  APWertB_1

timeline/Timestamp:  2011-03-23T14:20:00+01:00

intervalLength: 1000

data/Value: AQUMog == (01050ca2 in Base64 kodiert)

Events:

1: 10000 / 1000 == 0x000a

2: 70000 / 1000 == 0x0046

3: 130000 / 1000 == 0x0082

data/Events: AAoARgCC (000a00460082 coded in Base64)
```

3.11.4.1 Query Commissioned AP Values

The AP values previously applied to a node are returned:

Object Type: "NamedValue_Raw_Values"

If only certain AP values should be delivered, (e.g. only TX), in the protocol's "get" method, the "data" element with the data type "NamedValueFilterType" needs to be set . Otherwise all AP values which the server has collected for this system are delivered.

There are three filter options available for a query:

Variant 1 (node-related filtering):

"filterList" (without NamedValueFilterType)Restricts the delivered AP values to the specified AP values (default behavior of the filterList). Node and (optionally) OITD numbers must not be abbreviated.
Example 1: J1_12_4711_41.94_1 only this AP value of this nodeExample 2: J1_12_4711_61.111 only this AP value array of this node
Example 3: J1_12_4711 all AP values of this node

Variant 2 (AP value-related filtering):

"data" (data type NamedValueFilterType)Restriction to a list of AP values with the OITD numbers in IP notation (as in the case of commissioning) without node reference.Example 1: 41.94_1 defines a single AP value (only one instance)Example 2: 61.111 defines all instances of an AP value listExample 3: 61.111_213 defines an AP value instance of a list

Variant 3 (combined node and AP value filtering):

Enables orthogonal filtering through independent combination of node and AP value filtering.

"filterList" Restricts the delivered AP values to the specified nodes. Here, only the specification of nodes is allowed.
For example, J1_12_4711 all AP values of this node

"data" (data type NamedValueFilterType)Restricts the AP values pre-filtered to nodes by "filterlist" to the specified AP values. See variant 2.

Example of a combined XML request:

```
<ocitc:data xmlns:ns20="http://odg_und_partner/external/intersection_rawData"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="ns20:NamedValue-
FilterType">
   <ns20:NamedValueId>41.94</ns20:NamedValueId>
   <ns20:NamedValueId>41.96</ns20:NamedValueId>
   <ns20:NamedValueId>41.97</ns20:NamedValueId>
</ocitc:data>
<ocitc:filterList>
   <ocitc:filter>
      <ocitc:identifier>
         <ocitc:ident>J1_13_466</ocitc:ident>
      </ocitc:identifier>
   </ocitc:filter>
   <ocitc:filter>
      <ocitc:identifier>
         <ocitc:ident>J1_13_8161</ocitc:ident>
      </ocitc:identifier>
   </ocitc:filter>
</ocitc:filterList>
```

3.11.4.2 Setting of requestable AP values

To start externally requestable AP values, an object type has been introduced which makes a list of "requestable AP values" through precisely one interface partner possible. This means, the specification is made in the customer project as to whether the configuration of this list can even be set, and if yes, by which communication partners.

External commissioning is optional, but project-specific coordination is required in any case.

External commissioning is only possible to a limited extent, as the maximum number of total commissionable AP_Werten is limited by the device (device type, VA, detectors, ...).The request of the AP values by multiple communication partners is however possible, even if the configuration of the list can only be set by one partner.

Object type: "NamedValuesOrderList"

The list consists of the complete identifiers including object number of one or more AP values for each node. AP values, including list items, must be listed individually.

The following spellings are possible:

J1_12_4711_41.49_1 defines a single AP value (only one instance)

J1_12_4711_61.111_213 defines a single AP value instance of a list

The function depends on the method used:

put, delete:All AP values that have not yet been applied are applied or removed. By using the putList, AP values can be applied or removed from multiple nodes in one step. Due to

the direct influence of the control units, this potentially critical function must be agreed on a project-specific basis.

get:Query the currently requested and deliverable AP values from all nodes (without filterList) or only the nodes specified in filterList.

3.11.5 Public transport telegram

3.11.5.1 R09-Telegram

The object PTTelegramType (object type PublicTransportTelegram) has been defined for the transmission of the R09 telegram for public transport data. For the transmission in the protocol, the rawDC object is used as a container, supplementing the timestamp.

3.11.5.2 -Extended public transport telegram

The object PTExtendedTelegramType (object typePublicTransportTelegrammExtended) has been defined for transferring extended public transport telegrams. For the transfer in the protocol, the rawDC object is used together with the time stamp as the container.

3.11.6 Individual detector data

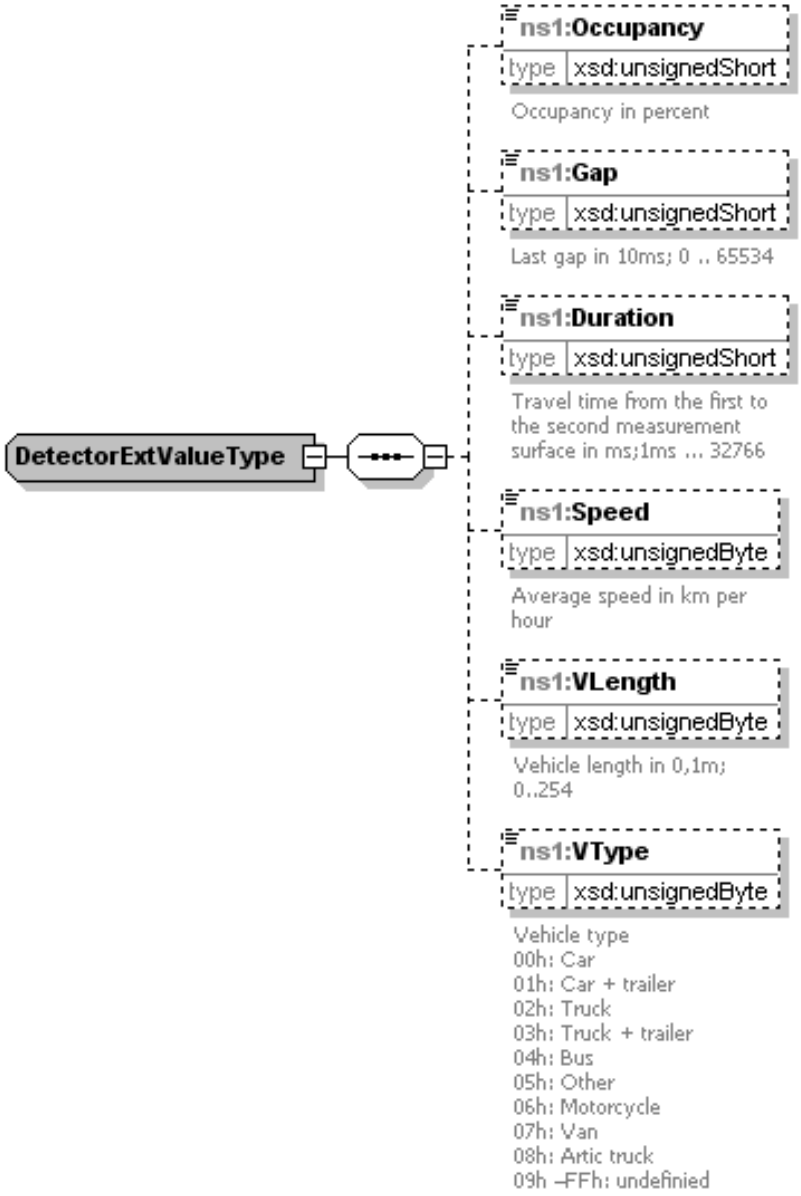
The DetectorExtValueType (detector_ext.xsd) object has been defined for transferringadditional information from detectors. The measurement refers to the detector event of an individual vehicle.

Note: Since large data quantities are involved with the transfer of individual values, the data should only be available for a few detectors.

Recommended identification:

DExt<primary key>_<secondary key>

<primary key> is optional.



The extended detector measurements include the following elements:

Occupancy - occupancy duration of the measuring point in 10ms

Gap - last gap time before vehicle in 10ms

Duration - journey time from the first to the second measuring point in ms

Speed - Measured speed in km/h

VLength - Length of the vehicle

VType - type of vehicle (cf. OCIT-O object VehicleType 1:3128)

3.11.7 Object type: Signalgroup_Description

The data structure is clarified in Chapter 2.4.1

Use element **Rel_Ids**

- Reference to TSS

3.12 Car-2-X Communication

This section provides a brief overview which describes all the Car-2-X communication objects for the OCIT-O V2.0 interface between a control centers.

3.12.1 Status information

The data from the C2X – communication should be used for a quality analysis and quality assurance of the intersections.

The vehicles send the status information from several sensors to the RSU via the CAM messages. Many data, such as the outside temperature, status of the brake booster or the angular position of the steering wheel are of no interest here. The data which are of interest for the traffic engineers are as follows:

- Time and date
- Position
- Speed
- Vehicle direction
- Vehicle type

The positions of the vehicles in a lane or even a signal group can be assigned using the intersection topology from the MAP message. The following values are provided for central analysis of the vehicle data (each in the area of the RSU):

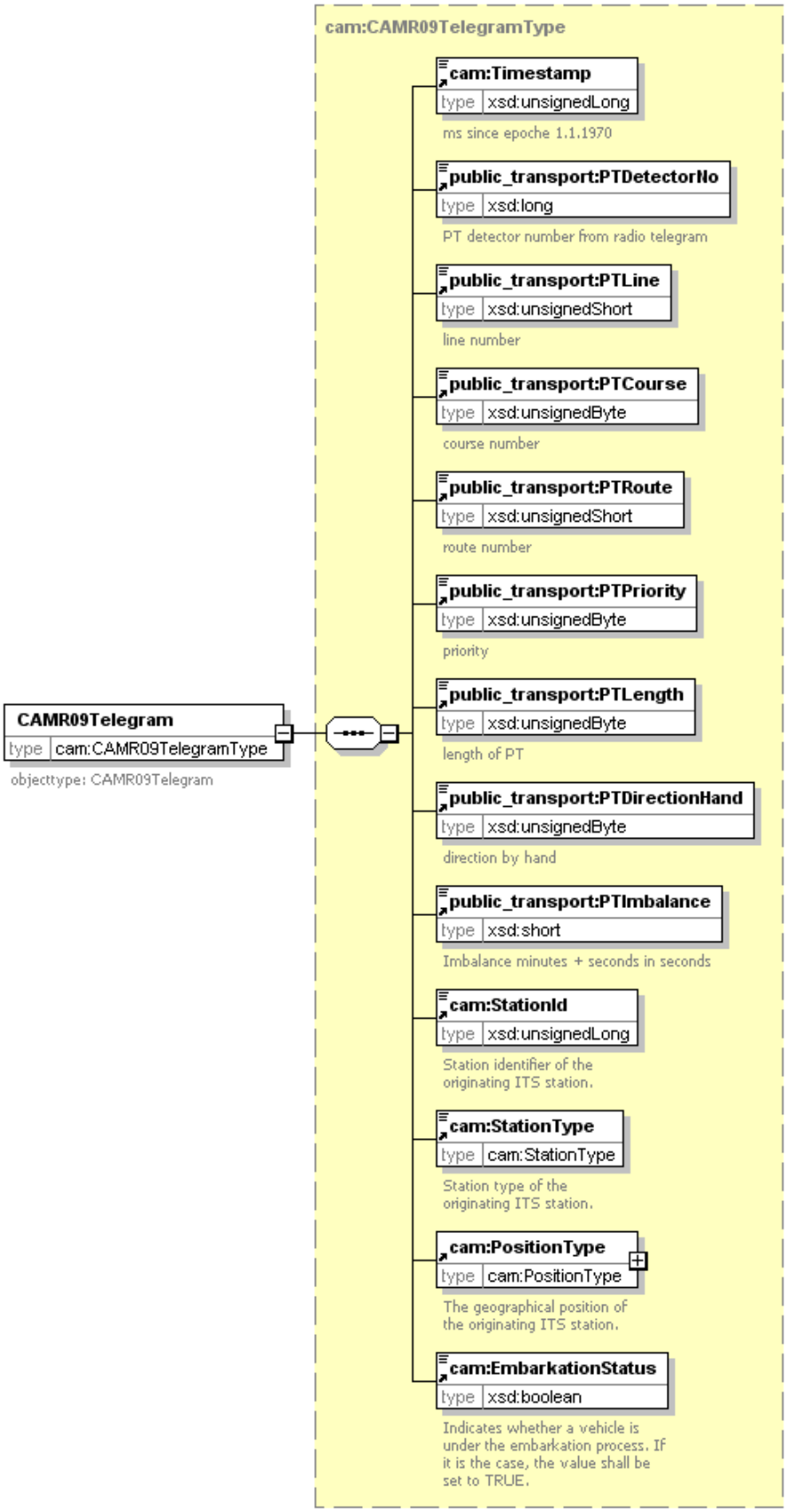
- Average speed
- Waiting time
- Number of stops
- connection-related average values

3.12.2 Prioritization

Public transport vehicles and special response vehicles are able to request a prioritization using the Car2X communication CAM telegrams. These request telegrams can be registered With OCIT-C Car.

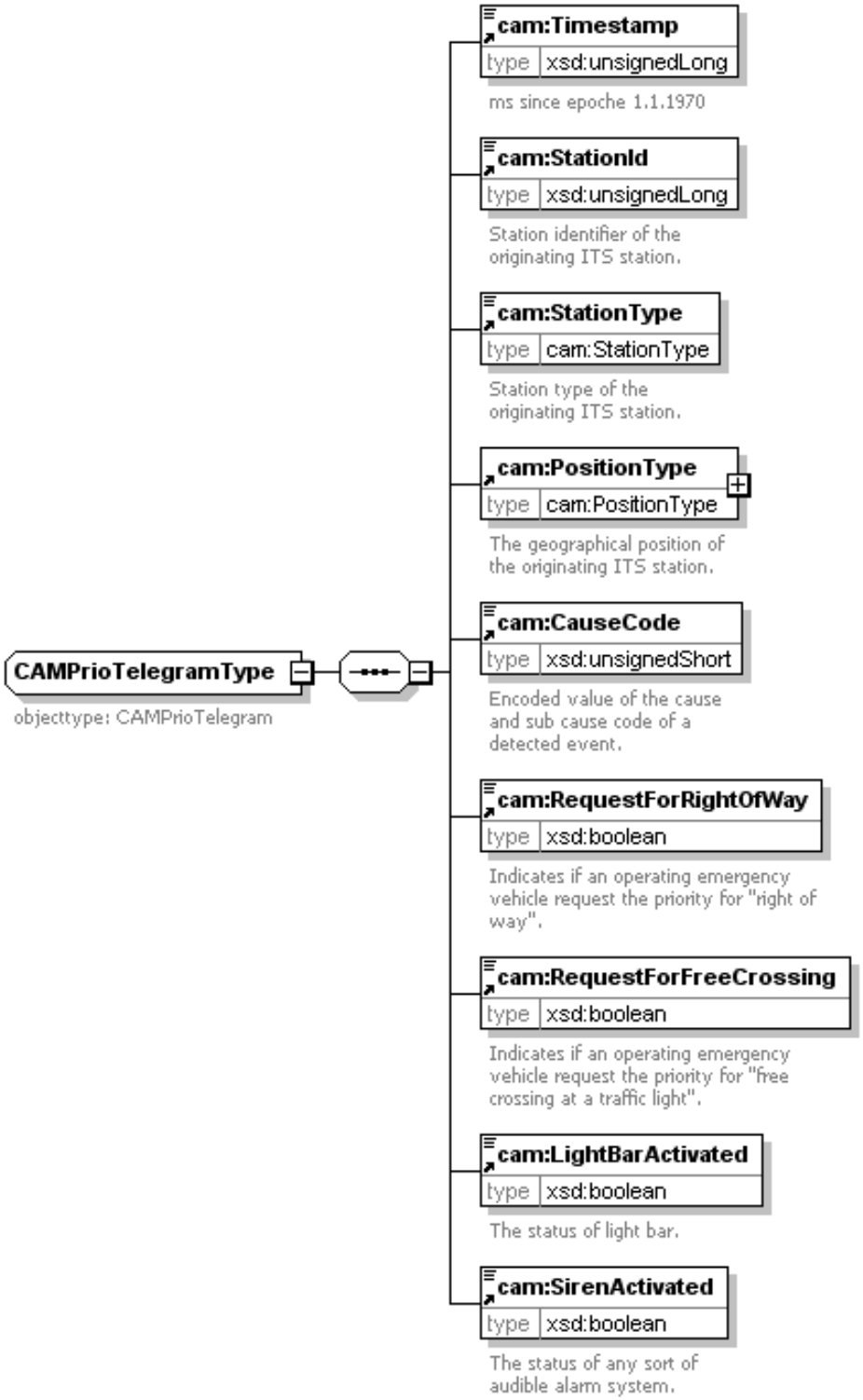
3.12.2.1 CAM-R09 telegram

The object CAMR09TelegramType (object type CAMR09Telegram) has been defined for transmitting CAM messages with an R09 container.


CAMR09Telegram
type cam:CAMR09TelegramType
objecttype: CAMR09Telegram
cam:CAMR09TelegramType
cam:Timestamp
type xsd:unsignedLong
ms since epoche 1.1.1970
public_transport:PTDetectorNo
type xsd:long
PT detector number from radio telegram
public_transport:PTLine
type xsd:unsignedShort
line number
public_transport:PTCourse
type xsd:unsignedByte
course number
public_transport:PTRoute
type xsd:unsignedShort
route number
public_transport:PTPriority
type xsd:unsignedByte
priority
public_transport:PTLength
type xsd:unsignedByte
length of PT
public_transport:PTDirectionHand
type xsd:unsignedByte
direction by hand
public_transport:PTImbalance
type xsd:short
Imbalance minutes + seconds in seconds
cam:StationId
type xsd:unsignedLong
Station identifier of the originating ITS station.
cam:StationType
type cam:StationType
Station type of the originating ITS station.
cam:PositionType
type cam:PositionType
The geographical position of the originating ITS station.
cam:EmbarkationStatus
type xsd:boolean
Indicates whether a vehicle is under the embarkation process. If it is the case, the value shall be set to TRUE.

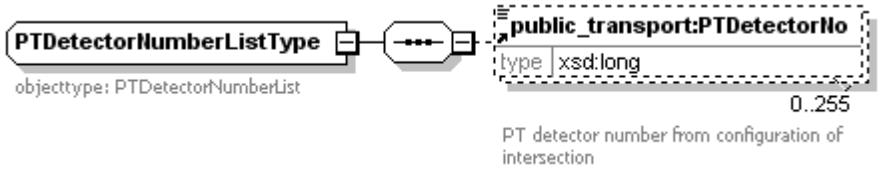
3.12.2.2 CAM prio telegram

The object CAMPrioTelegramType (object type CAMPrioTelegram) has been defined for transmitting CAM Meldungen with a Rescue, Emergency or SafetyCar container.



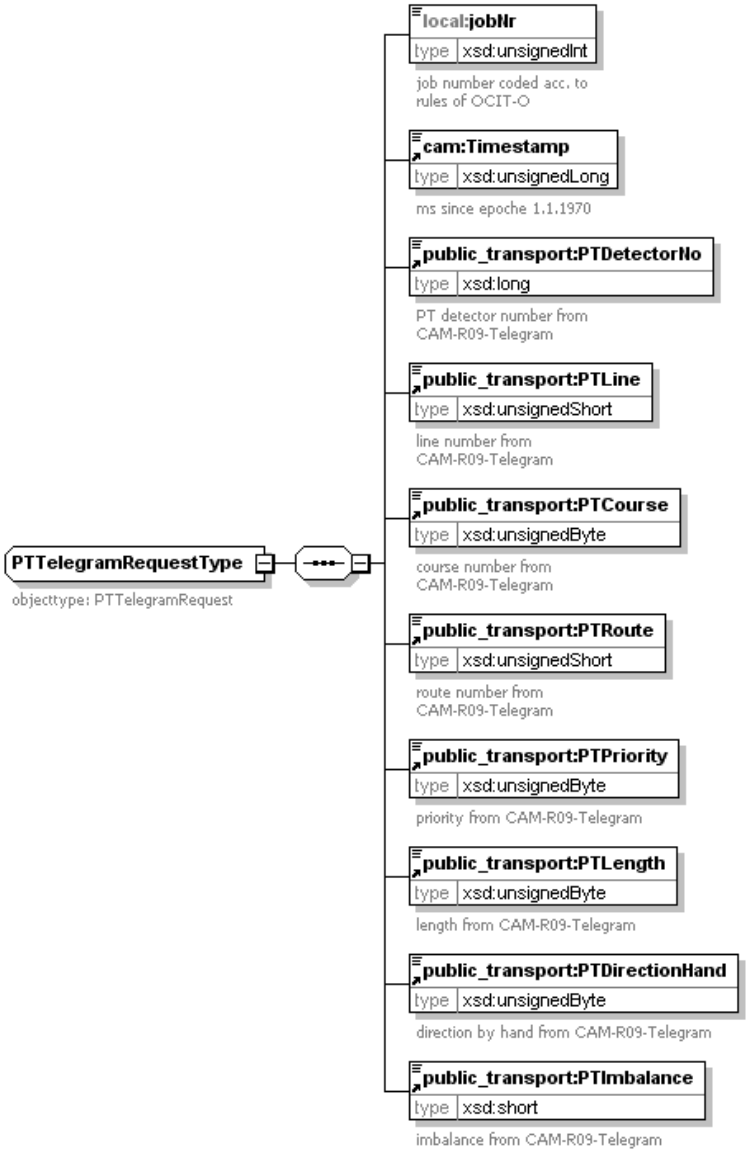
3.12.2.3 PT reporting point list

The object PTDetectorNumberListType (object type PTDetectorNumberList) has been defined for transmitting PT reporting points configured in a TSS. This way, you can request the TSS's PT reporting point list.



3.12.2.4 PT request

The object PTTelegramRequestType (object type PTTelegramRequest) has been defined for forwarding PT request telegrams (R09 telegrams). The OCIT-C server adds missing optional values if these are needed for communication with the TSS. What values can be transmitted to the TSS depends on the protocol between the OCIT-C server and the TSS.



3.12.3 Risk messages

Available object types:

Object type	Schema file	Short description
DENMessage	Denm.xsd	Dencentralized Environmental Notification Message

Risk messages can be triggered at the control center level, as well as at the field level, and can be distributed across the entire system.

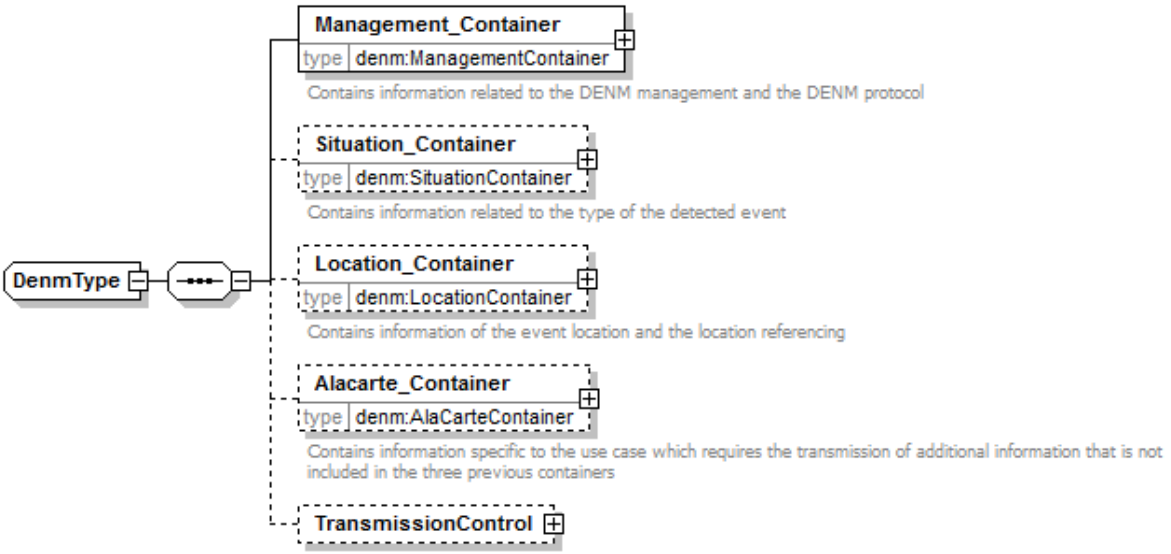
Risk messages cover the following areas:

- Roadworks
- Incident and incident messages
- Traffic situation
- Road condition
- Meteorological data (wind, rain, visibility)

Risk messages are transmitted with the DENMessage object.

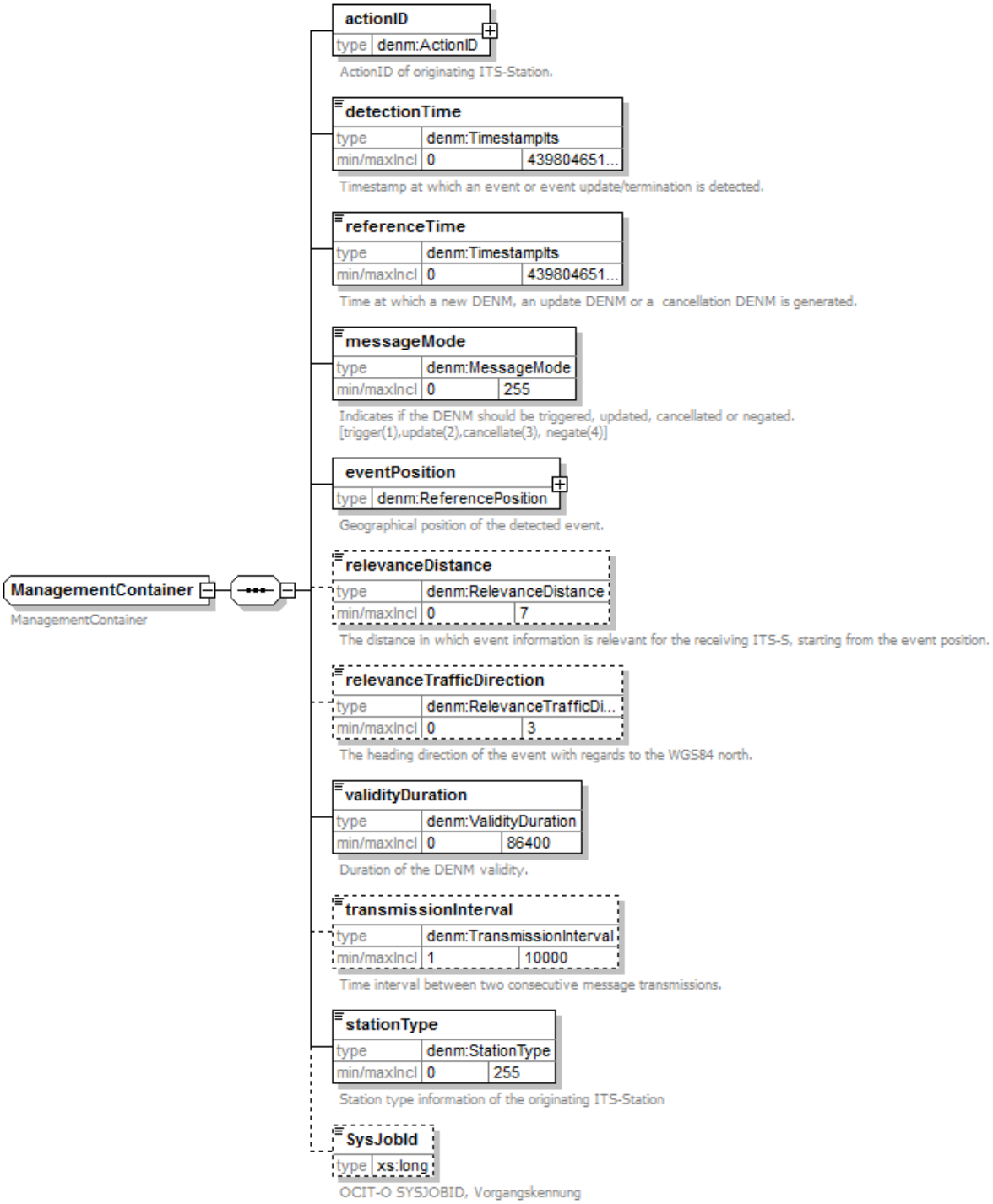
A DENMessage object is composed of 5 blocks:

- Management_Container
- Situation_Container
- Location_Container
- Alacarte_Container
- TransmissionControl



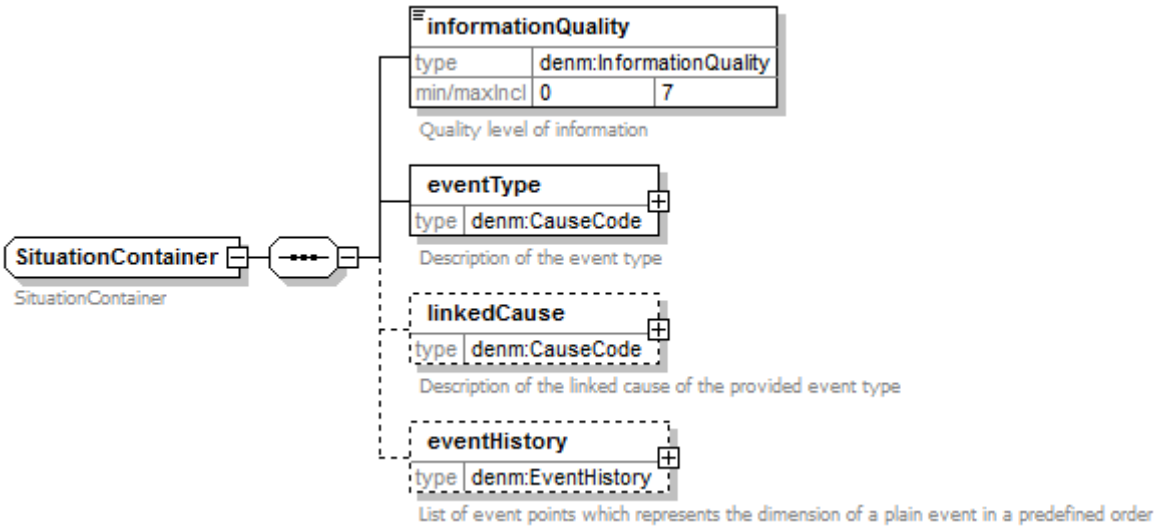
3.12.3.1 Management_Container

Contains information related to the DENM management and the DENM protocol.



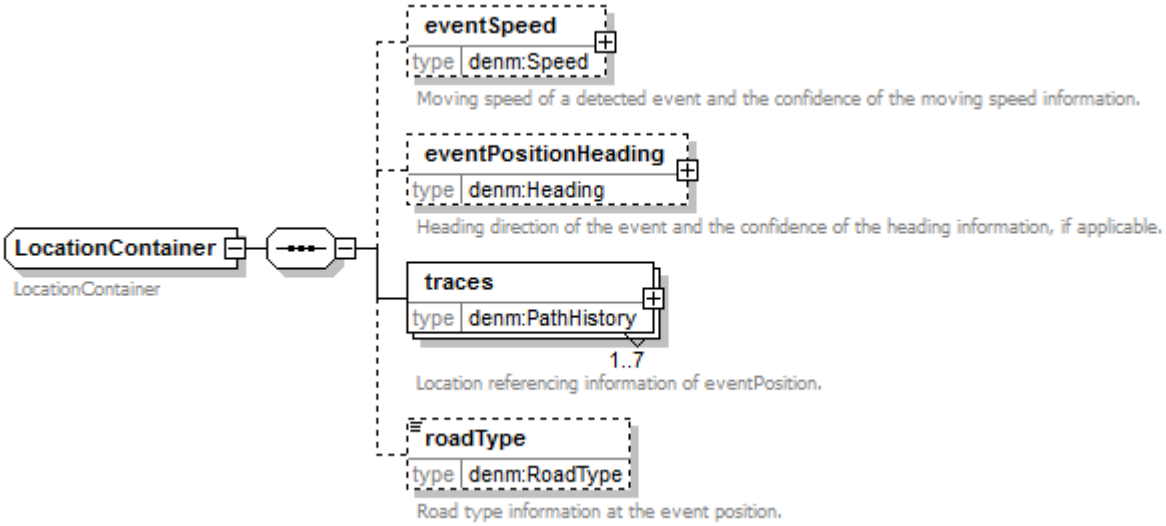
3.12.3.2 Situation_Container

Contains information related to the type of the detected event.



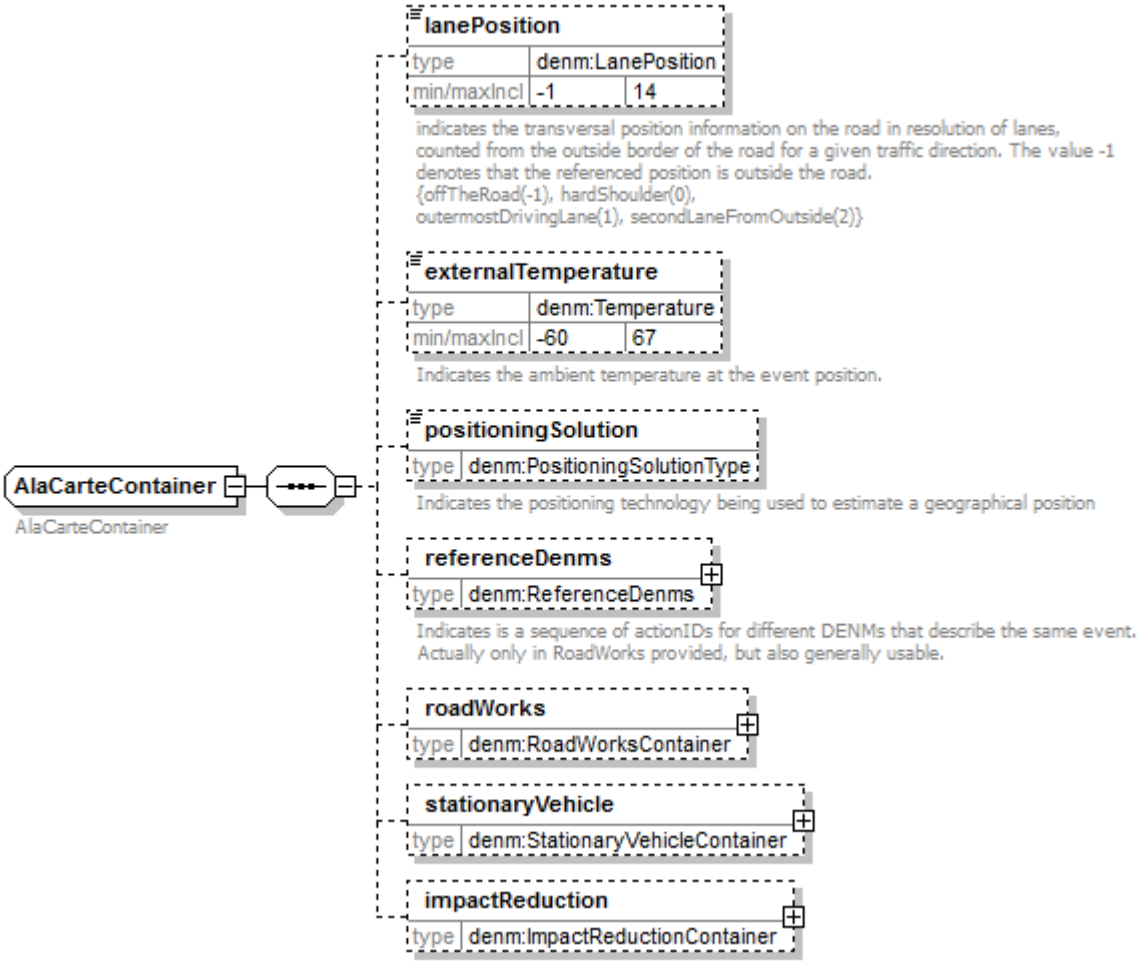
3.12.3.3 Location_Container

Contains information of the event location and the location referencing.



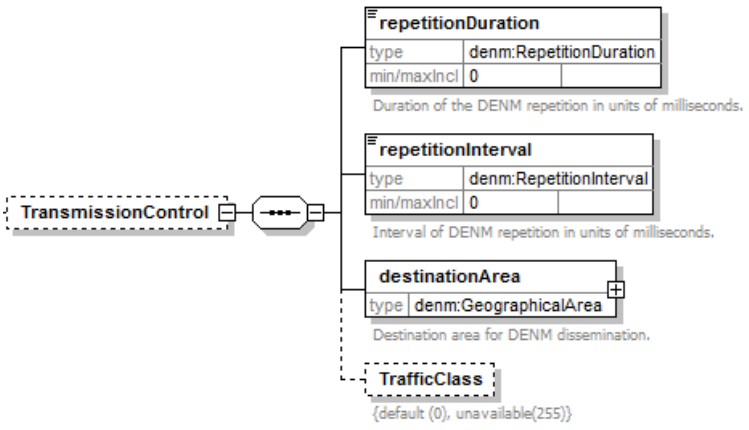
3.12.3.4 Alacarte_Container

Contains information specific to the use case which requires the transmission of additional information that is not included in the three previous containers.



3.12.3.5 TransmissionControl

Contains information about transmission features.



3.12.4 Spat

The Spat object offers the option of requesting the predicted duration for the current status of a signal group or to also exchange this between control centers. Transmission is required because it is expected that in the future there will be centralized prediction processes that create predictions based on historical data.

The object type presented in the following can be used for the individual request or the transmission of the forecast signal status of signal groups.

3.12.4.1 Object type PredictedSpat

The object type PredictedSpat can be addressed once per traffic signal system. Identification is performed based on the addressing of traffic signal systems used in OCIT-C.

ObjectTypePredictedSpat			
Id			Identification of the traffic signal system
predictionSource			Name of the prediction process (origin)
timeline			Contains UTC time stamp from global.xsd which the values below relate to.
sg (Array)			Number of the following datasets is at most equal to the number of signal groups or the number of signal groups for which there is a forecast.
	nr		Number of the signal group
	forecast (Array)		Number of field entries is equal to the number of signal group color changes available for this signal group.
		offset	Time offset to the time stamp in 100 ms units Future > 0 History (if required): <0
		quality	Quality statement of the predicted value for the signal group in question in %
		trafficState	Current or future status of the signal group (0=GREEN, 1=RED)
		color	Current or future color status of the signal group (OCIT color code)
		duration	Predicted remaining time for the current or future signal status (in 100 ms units)
		min	Minimum remaining time for the current or future signal status
		max	Maximum remaining time for the current or future signal status (For fixed time, ProgDuration, MinDuration and MaxDuration are identical.)

		transition	Duration of the transition between the described and the following status (0 = no transition present)

An unavailable Spat is identified using the corresponding ErrorCodes of the OCIT-C.

Note: The individual request always returns the current and optionally the next n following signal end states. The time stamp in conjunction with the time offset calculates the start of the current status, i.e. depending on the time of the request it may also be in the past. The offset is then 0 and the status is already present, however it is not possible to say how long the status has already been present. The transferred duration values refer to the time, timestamp + offset.

3.12.5 MAP topology data

Currently, there is still insufficient specific experience from practical applications. This is why the MAP data are applied in the same format as in the specifications of the ETSI (ASN.1 Format). The data is transmitted from the traffic engineer worksite to the controller.

Possible modifications driven by experiences from practical application are expected.

Because the data are only passed through as a data block, no modified or detailed checksum management for the block of MAP data takes place in OCIT-C. A checksum is only calculated using block 5 with the usual algorithm.

3.13 TSS config data

This communication module is used for supplying data to traffic signal systems. Fault messages for the planning tool

- Asynchronous feedback if there is a change of supply
- List of the AP values which can be delivered for a TA process
- Request of the field device information

For a detailed description, see document "OCIT-C_TSS_Supply data".

3.14 Binary container

This communication module is used to transmit binary encoded data. The data are assigned to a point in time (or time period) and an object. The name type identifies the type of data in the container and thus makes it possible for the recipient to process.

Binary data are identified using pre-defined IDs.

For example: The so-called "Compromised messages" from a TSS should be transferred to the container, the name type in this case would be "KPM", the object ID would be specified

as the ID of the TSS and the data container would contain a complete compromised message, as well as the one transferred from the TSS.

Recommended identification:

BC<primary key>_<secondary key>

The use of the primary key is not obligatory.

Available object types:

Object type	Schema file	Short description
BinaryContainer	binary_container.xsd	Binary container, the contents of which can be specified on a project-specific basis.

3.15 Project-specific extensions

Project-specific extensions of communication modules are possible. The following must be taken into consideration:

- It must be possible to disengage project-specific data structures using uniquely identifiable project or customer-specific object types.

  These newly introduced object types must be stored in separate schema definitions and relate to the standard XML data types.

  To prevent ambiguousness, separate manufacturer-specific prefixes are placed before the object types used. In addition, the groups to which the data type belong are indicated. An underscore follows the prefix ("_")

  For example:

  Manufacturer "abc"
  New object type "xyz"
  New resulting identifying object type: "abc_xyz".

- Extensions as part of protocol functions must be avoided under any circumstances.

- Extensions to existing data types must be avoided. Should it not be possible to avoid this, it is necessary to take into account the compatibility rules set out in the next chapter.

3.15.1 Compatibility rules

Extension rules for public data models (schema definitions) and the assigned object types:

- Elements and attributes may be added to existing schema definitions,

- New attributes and elements must be optional in order to ensure backwards compatibility.

- The consuming interface must respond flexibly to the availability of the newly entered elements/attributes.

- The consuming interface must be respond flexibly to any unknown elements / attributes and skip these upon encountering without any further assessment. This means unknown elements may not influence the behaviour of the decoding of the received XML structure.

- It is not permitted to delete attributes or elements from schema definitions.

- It is not permitted to modify the semantics of existing attributes or elements.

4 Terms and abbreviations

Term / abbreviation	Description
AP	User program
CAM	Cooperative Awareness Message
Car2X	Communication between vehicles as well as vehicles with their surroundings.
Client	A program which wishes to use services offered by other (servers) and actively opens them to do so.
DATEX II	Specifications of Technical Committee 278 of the European Committee for Standardization (CEN) for the exchange of traffic-related data between traffic control centers.
DENM	Dencentralized Environmental Notification Message
FTP	File Transfer Protocol, a network protocol for transferring files
http	HyperText Transfer Protocol, a protocol for transferring data over a network.
TSS	Traffic signal light system
MAP	Topology Info for Intersection and Road Segments
Method	The algorithms assigned to a class of objects. Also used as a synonym for function, procedure, command, action.
PT	Public Transport
OCIT	Open Communication Interface for Road Traffic Control Systems.
OCIT-C	Open Communication Interface for Road Traffic Control Systems - Center to Center. OCIT-C covers the functions for communicating between the central traffic control and traffic guidance systems.
OCIT-O	OCIT Outstations Interface between traffic control centres and traffic signal controllers for controlling and supplying the traffic signal controllers.
ODG	OCIT Developer Group
OSI	Open Systems Interconnection Reference Model, a communication model of the International Organization for Standardization (ISO) for communication protocols in computer networks.
OTS 2	Open Traffic Systems, Version 2

Server	A program that offers certain services and passively waits on incoming calls (from clients) to do so.
SOAP	SOAP (Simple Object Access Protocol), a protocol which enables data to be exchanged between systems. SOAP uses the "Remote Procedure Call", through which it enables the functions in other computers to be called. See http://www.w3.org/TR/SOAP
SSL	Secure Socket Layer.
Soap-Server-Interface	Soap and Protocolmanager on the server side
Soap-Client-Interface	Soap and Protocolmanager on the client side
Protocolmanager	Protocol layer used for implementing commands in the buffer
TLS	Technical delivery terms for roadway stations. The TLS are a standard for the structure of traffic control systems on major German Federal highways. Issued by: German Federal Highway Research Institute
TCP / IP	Transmission Control Protocol / Internet Protocol, a family of network protocols for the Internet.
VDV	Association of German Transportation Companies
WSDL	Web Services Description Language, a platform / programme language and protocol-independent description language for network services (web services) for exchanging messages based on XML.
XML	Extensible Markup Language, a markup language for presenting structured data in the form of text. XML is used among other things for a platform and implementation-independent exchange of data between computer systems. An XML document is made up of text characters, in the most basic case in ASCII coding, and is therefore machine-readable. It does not contain binary data. The XML specification is published by the World Wide Web Consortium (W3C) as a recommendation.
XSD	XML schema, a recommendation of the World Wide Web Consortium (W3C) for defining structures for XML documents. The structure is described in the form of an XML document. Furthermore, it supports a large number of data types. The XSD schema language describes data types, individual XML schema instances (documents) and groups of such instances. A specific XML schema is called an XSD (XML Schema Defintion) and the file usually has the ending ".xsd".

Further explanations about the technical terms and abbreviations used in this document can be found in “OCIT – O Glossary V3.0”.

OCIT-C_Data_V2.0_A02