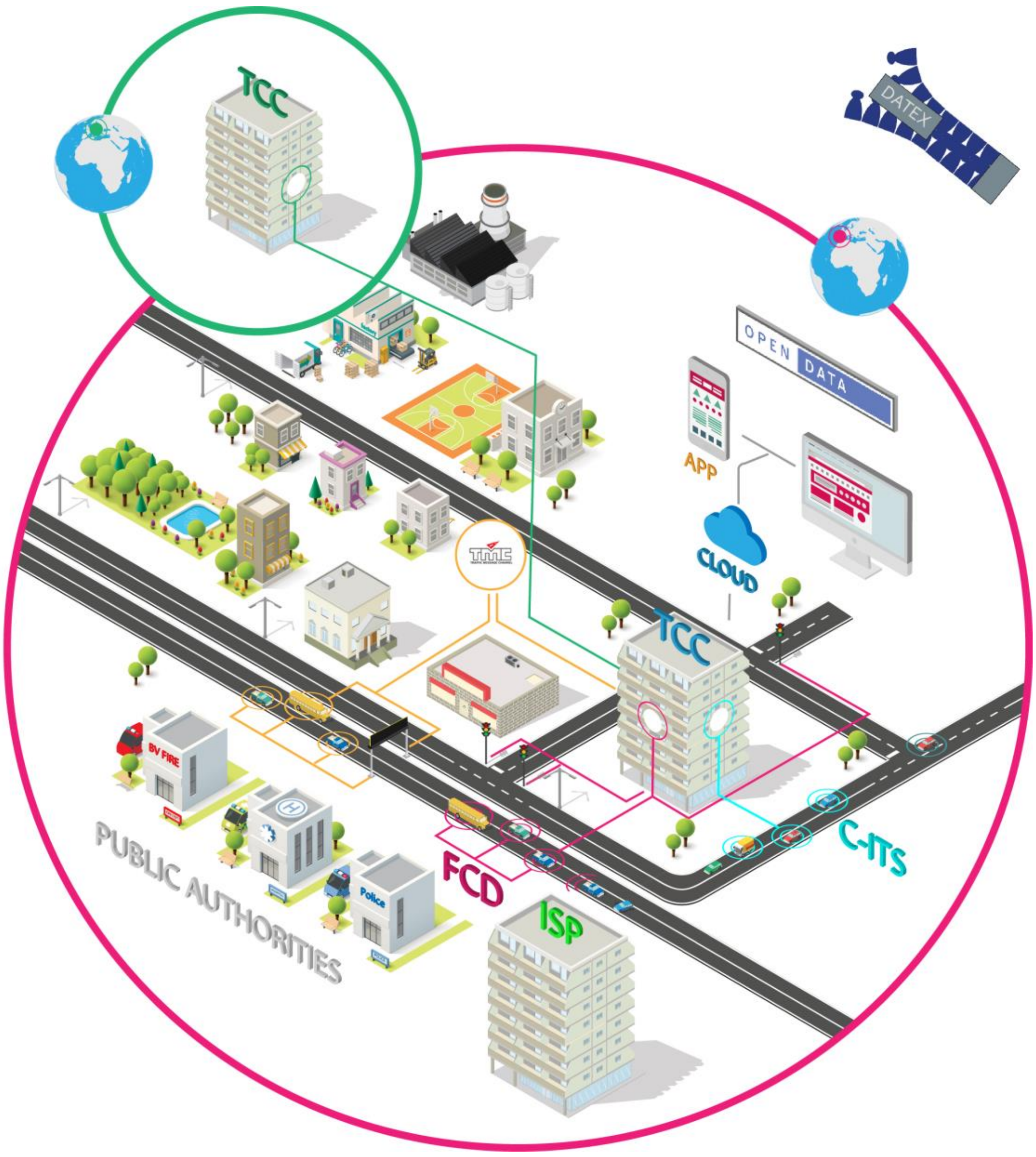
D2Light

Report 2018-xx-xx



Index

Introduction

In many countries the need for a detailed information model is recognised and properly reflected by DATEX II. However, this level of detail is perceived as complex and heavy for relative simple information services that are supported by app developers. There is a strong demand from road operators to open up their data to the app developers in order to achieve maximum usability of their data in de Open Data domain. To support the ability to meet this demand, a task in Activity 6 for DATEX Light was already set up in the General Agreement.

To make things light the title is changed to D2Light. This report covers D2Light and its principles and should be regarded as a proposal of best practices for introducing an API directed to app developers which provides real-time road traffic information covered by DATEX II.

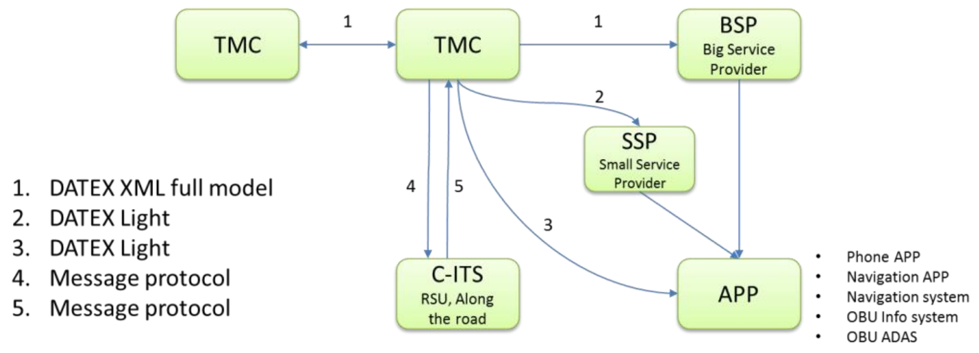
As a basis the existing principles and concept of DATEX II is applied where possible. The proposal is based on publication and uses the tool to produce a schema which defines the content.

D2Light is not aiming for formal standardisation.

UseCase

To be revised.

Positioning of D2Light and DATEX II original full model. In the figure, the Functional roles are depicted, in which DATEX II is used. DATEX II Full model is interface of type 1 and all other interfaces are not officially standardized usages of (parts of) DATEX II.



D2Light is aimed at the exchange between a Traffic management Centre and a Small Service Provider as well as the exchange with end-user apps directly. It covers both the content and the exchange mechanism.

The functional chain characteristics are:

- Aimed at displaying the provided information
  - Information display in readable format
  - Most often shown on a map
  - Support of standardized iconset
- Limited processing of data to display information
- No integration of this data in advanced mobility services
- No traffic information engineering knowledge available

For Service Providers who have additional sources and will add extra value the XML full model is more appropriate with its more details.

Governance

To be revised.

In order to keep the simplified approach simple there must be …………………

Requirements

Very specific and detailed requirements were not made from start, apart from: It should be JSON and it should be simple. A team started the work and followed the approach:

- Determine the operational environment of DATEX II Light
- Determine design principles and prerequisites
- Review and harmonising the existing API's in Sweden and Netherlands.
- Base it on these harmonized API
- Create a simplified model
- Use the DATEX II Conversion tool to generate JSON Schema's

With this basic principles a draft of D2Light is developed. The draft has been reviewed inside the TMG and comments have been taking in consideration.

Design principles

The principles for D2Light are

- Keep it simple!
    - Don’t make the development take too long, don’t overcomplicate, and do not add extra complex components. If there’s alternatives, chose the simple one
- Ease the processing at the client
    - The client should just need to do the rendering of the information in order to make it useful.
- Flattened exchange model
    - Not as deep, no need for traversal, very easy to extract the information
- Self-contained, all information needed is in the response
    - It is not necessary to download additional data from other services.
    - No references to other publications are made
- DATEX II V3 based
    - Don’t invent and insert something new which is already specified in DATEX II V3
    - Reuse objects, attributes names etc.
- The response is language dependent
    - No multilingual strings in order to use ordinary string types
    - Language text etc. should be specified in the request. The response will only contain translations of types (if exists) and free text fields in the requested language.
    - For the server side it means some extra work for Types or coded values which should be translated to the language specified in the request and Multilingual Strings have to be replaced by simple strings.
- No complex automated creation of simplified model.
    - As foreseen D2Light will not address many different publications. At this stage, do not introduce a complex configuration and process to the model creation for a few instances.
    - The naming schemes with an automatic generation would be another complexity. If the namespaces are to be intact, it would cause very long names.
- The simplified model as an ordinary new DATEX II publication
    - Keep familiar concepts and the process of schemas
- Only return active objects/situationrecords.
    - Based on the assumption that the results is only of interest for events active here and now.
    - Will simplify decoding/filtering in the client.
    - An exception will be if a Schedule is specified.
- Always provide geographic information as use is often for a map view
    - Use methods that easily can be displayed on maps e.g. with simple coordinates and LineString
    - As a minimum coordinates should be provided
- Situations as initial publication

    - Selected with all events and the related information limited to safety related messages. Will support priority actions C. Forthcoming steps can add other required situation types.
    - Regarded as one type of information suitable for apps.
- Incorporate JSON in the tool in the same way XML schema's are produced
    - Keep the same concept in tool, makes it more generic and maintainable
- Use JSON schema draft V4
    - Use a late version for the encoding
- Simple and widely used exchange model
    - For an app this will mean support for a RESTful implementation and in practice HTTP request with JSON response.

Tool support

The tool is modified in order to support the schemas for D2Light. As a result we now have the opportunity to produce several varieties of schemas. In essence the tool modification allows to produce for any package (or profile) a XML or a JSON schema.

The tool is designed and intended to generate a XML Schema for the full model and a JSON Schema for the D2Light publication.

	XML	JSON
Full model	**As currently**	Usability unknown
D2Light	Usability unknown	**New JSON Schema**

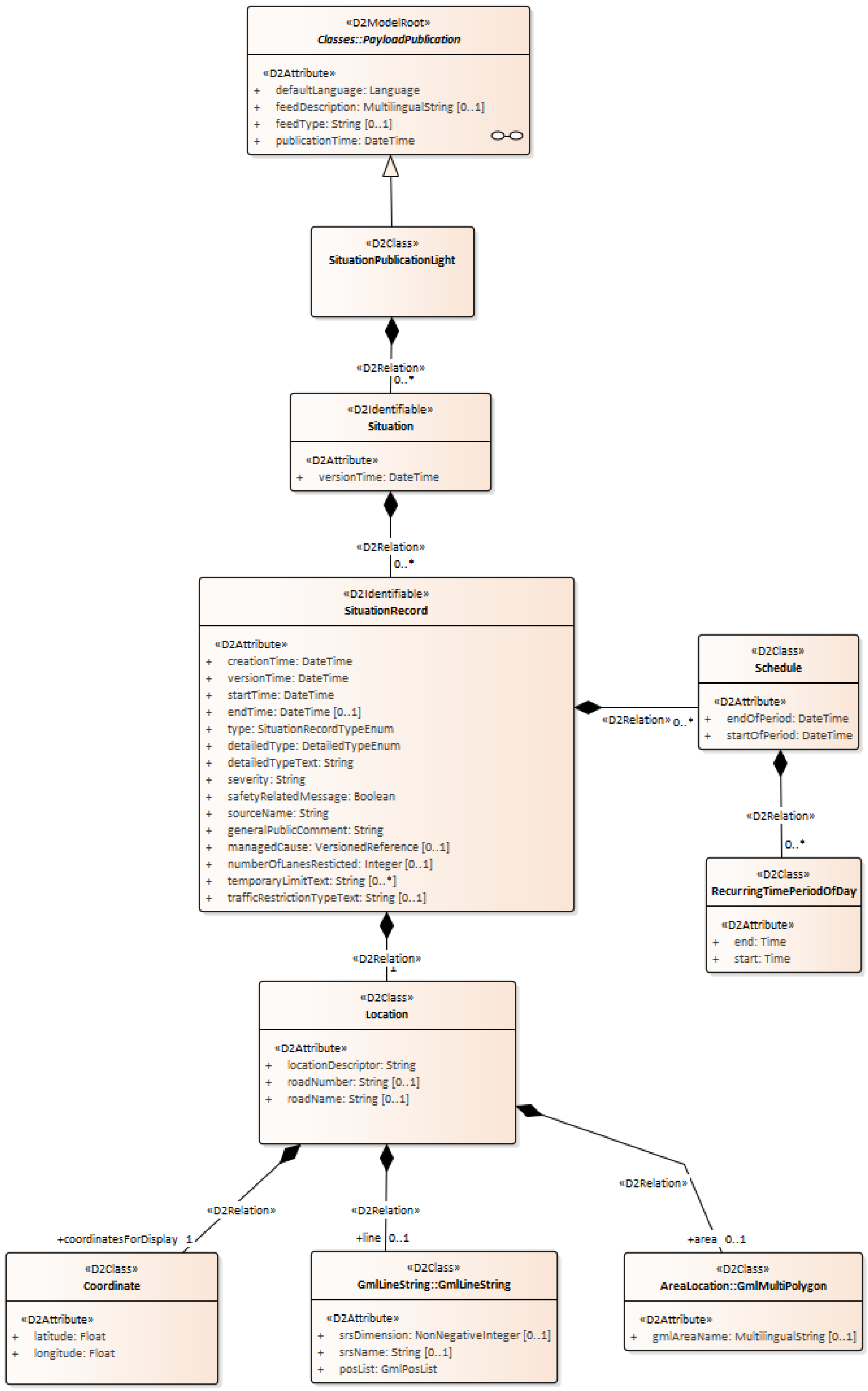
Implementation pre-requisite

It's a presumption that the information generated to fill the elements in D2Light publication is based on the DATEX II full model in the relevant parts. This simplified model is an exchange model where we assume that the DATEX II PIM information model exists and thus will guarantee consistency in messages. The simplified model and publication can be thought of as a new view into the full model and information.

The common way of receiving JSON encoded information is not to use a Schema. However for the sender the use of the JSON Schema will give a proper way to assure consistency with expected format.

Model

Relevant part from the full model are compiled into this new Publication.


«D2ModelRoot»
Classes::PayloadPublication
«D2Attribute»
+ defaultLanguage: Language
+ feedDescription: MultilingualString [0..1]
+ feedType: String [0..1]
+ publicationTime: DateTime
«D2Class»
SituationPublicationLight
«D2Relation»
0..*
«D2Identifiable»
Situation
«D2Attribute»
+ versionTime: DateTime
«D2Relation»
0..*
«D2Identifiable»
SituationRecord
«D2Attribute»
+ creationTime: DateTime
+ versionTime: DateTime
+ startTime: DateTime
+ endTime: DateTime [0..1]
+ type: SituationRecordTypeEnum
+ detailedType: DetailedTypeEnum
+ detailedTypeText: String
+ severity: String
+ safetyRelatedMessage: Boolean
+ sourceName: String
+ generalPublicComment: String
+ managedCause: VersionedReference [0..1]
+ numberOfLanesResticted: Integer [0..1]
+ temporaryLimitText: String [0..*]
+ trafficRestrictionTypeText: String [0..1]
«D2Relation» 0..*
«D2Class»
Schedule
«D2Attribute»
+ endOfPeriod: DateTime
+ startOfPeriod: DateTime
«D2Relation»
0..*
«D2Class»
RecurringTimePeriodOfDay
«D2Attribute»
+ end: Time
+ start: Time
«D2Relation»
1
«D2Class»
Location
«D2Attribute»
+ locationDescriptor: String
+ roadNumber: String [0..1]
+ roadName: String [0..1]
«D2Relation»
+coordinatesForDisplay 1
«D2Relation»
+line 0..1
«D2Relation»
+area 0..1
«D2Class»
Coordinate
«D2Attribute»
+ latitude: Float
+ longitude: Float
«D2Class»
GmlLineString::GmlLineString
«D2Attribute»
+ srsDimension: NonNegativeInteger [0..1]
+ srsName: String [0..1]
+ posList: GmlPosList
«D2Class»
AreaLocation::GmlMultiPolygon
«D2Attribute»
+ gmlAreaName: MultilingualString [0..1]

The **SituationPublicationLight** is derived from **PayloadPublication**. The **SituationPublicationLight** have zero or many **Situation.**

The **Situation** is stereotyped as **D2Identifiable** and has a **versionTime** attribute. The Situation as zero or many **SituationRecord.**

Elementname	Purpose	Relation to full PIM
creationTime	Time when the situation was first created by sender	Identical
versionTime	Time when the information in the situationrecord was updated	Identical
startTime	Time when the event in the situationrecord was supposed to have started	Identical
endTime	Time when the event in the situationrecord is supposed to end	Identical
situationRecordType	Defines the kind of information in the record, enabling the presentation of icons	Based on the possible types. Can be used to display icons (GDD reference?)
detailedType	Provides the type of event, enabling the presentation of icons	Could be all enumliterals from all types. Questionable whether this is usefull
detailedTypeText	Textual representation of the event in requested language.	Based from the transformation of the information in the eventtype-enums. In case a cause (which is not a record, or situation) is known, it should be mentioned here as well (so *lane closed, due to an accident*) In some can be combined with supplementarylocation information elements (e.g. *use of hardshoulder allowed*, instead of *use of identified lane allowed* and then mention in location the hardshoulder)
Severity	Tells the severity of the event	Identical
SafetyRelatedMessage	Tells if this is a safety related message or not	Identical
SourceName	Gives the name of the source	Identical
GeneralPublicComment	Gives the public comment for the event	Identical

ManageCause	Makes it possible to relate **situationRecords t**o each other and tell the cause	
numberOfLanesRestricted	Tells the number of lanes restricted.	Identical
temporaryLimitText	Tells a textual description of temporaryLimits e.g. "SpeedLimit 50 km/h"	
trafficRestrictionTypeText	Gives a textual description of restrictions e.g. "one lane blocked."	

SituationRecord has the following aggregations **Schedule** and **Location**.

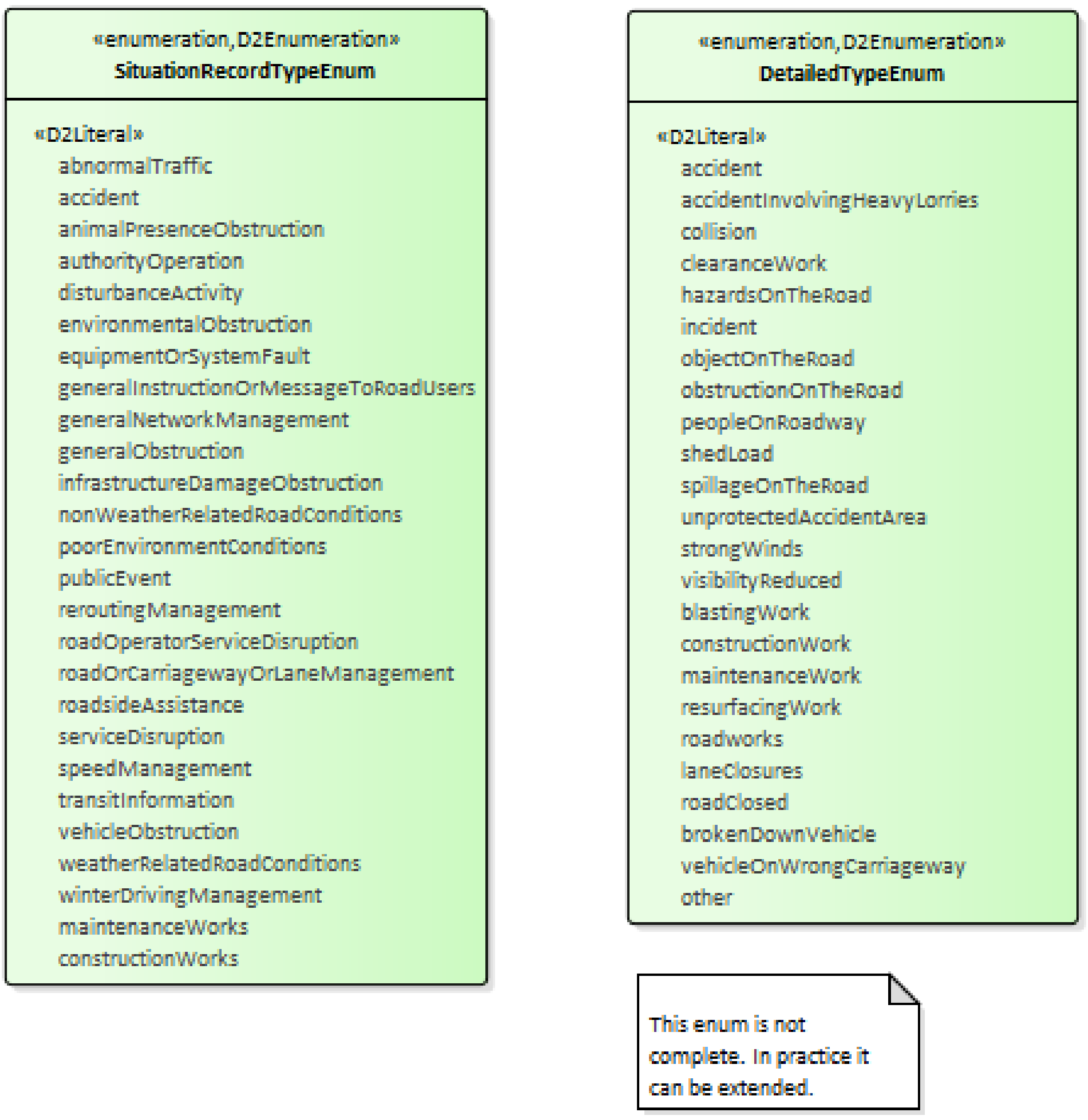
Schedule give the possibility to in more detail specify when an element is supposed to be active or not in time.

In most countries the **location** information is either based on the TMC location referencing or openLR. Interpreting this information requires substantial effort. Hence in D2Light the texts that are required to present the location information has three attributes

Elementname	Purpose	Relation to full PIM
locationDescriptor	The full description of the location. Could be something like: *On the A12 Utrecht-Arnhem, between Bunnik and Maarn on the parallel carriageway*	None
roadName	The name of the road in case it does not have a number or has a name as well (e.g. *Route du soleil* or *Brüderstrasse*)	only in case tpeg or linearreferencing is used
roadNumber	The roadnumber (e.g. M1 or E19) in order to facilitate lists of roads with incidents.	only in case tpeg or linearreferencing is used

Location has one aggregation: coordinate in the role of **locationForDisplay**.

Location can optionally have one **GmlLineString** for representing linear locations by coordinates or one **AreaLocation** representing an area as a **GmlMultiPolygon**.



The **SituationRecordTypeEnum** is an enum representing all situation record types in the complete model. Each derived class from the **SituationRecord** Class has an enumeration literal in the enumeration.

The **DetailedTypeEnum** is an enumeration containing all the types that the API supports. It's a merge of all "types" enums in the full model. As the note says, in the diagram below, the enumeration can be extended to include the desired enumeration literals. There is a concern about this, that you need to extend and change schema, just to add a type. The solution for this is to make the DetailedType's datatype to a string instead of an enumeration.

Encoding

The UML model is mapped JSON mapping principles. For more detailed information, see the accompanying "JSON Schema Definition mapping" document.

JSON Schema

Below is the schema for SituationPublicationLight.

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "definitions": {
                "_DetailedTypeEnum": {
                        "title": "_DetailedTypeEnum",
                        "additionalItems": false,
                        "properties": {
                                "value": {
                                        "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/DetailedTypeEnum"
                                },
                                "_extendedValue": {
                                        "type": "string"
                                }
                        },
                        "required": [
                                "value"
                        ]
                },
                "_SituationRecordTypeEnum": {
                        "title": "_SituationRecordTypeEnum",
                        "additionalItems": false,
                        "properties": {
                                "value": {
                                        "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/SituationRecordTypeEnum"
                                },
                                "_extendedValue": {
                                        "type": "string"
                                }
                        },
                        "required": [
                                "value"
                        ]
                },
                "Coordinate": {
                        "title": "Coordinate",
                        "type": "object",
                        "properties": {
                                "latitude": {
                                        "title": "latitude",
                                        "$ref": "DATEXII_3_Common.json#/definitions/Float"
                                },
                                "longitude": {
                                        "title": "longitude",
                                        "$ref": "DATEXII_3_Common.json#/definitions/Float"
                                },
                                "_coordinateExtension": {
                                        "title": "_coordinateExtension",
                                        "$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
                                        "type": "object",
                                        "additionalItems": false
                                }
                        },
                        "required": [
                                "latitude",
                                "longitude"
                        ]
                },
                "DetailedTypeEnum": {
                        "title": "DetailedTypeEnum",
                        "type": "string",
                        "enum": [
                                "accident",
                                "accidentInvolvingHeavyLorries",
                                "collision",
                                "clearanceWork",
                                "hazardsOnTheRoad",
                                "incident",
                                "objectOnTheRoad",
                                "obstructionOnTheRoad",
                                "peopleOnRoadway",
                                "shedLoad",
                                "spillageOnTheRoad",
                                "unprotectedAccidentArea",
                                "strongWinds",
                                "visibilityReduced",
                                "blastingWork",
                                "constructionWork",
                                "maintenanceWork",
                                "resurfacingWork",
                                "roadworks",
                                "laneClosures",
                                "roadClosed",
                                "brokenDownVehicle",
                                "vehicleOnWrongCarriageway",
                                "other",
                                "_extended"
                        ]
                },
                "Location": {
                        "title": "Location",
                        "type": "object",
                        "properties": {
                                "locationDescriptor": {
                                        "title": "locationDescriptor",
                                        "$ref": "DATEXII_3_Common.json#/definitions/String"
                                },
                                "roadNumber": {
                                        "title": "roadNumber",
                                        "$ref": "DATEXII_3_Common.json#/definitions/String"
                                },
                                "roadName": {
                                        "title": "roadName",
                                        "$ref": "DATEXII_3_Common.json#/definitions/String"
                                },
                                "coordinatesForDisplay": {
                                        "title": "coordinatesForDisplay",
                                        "$ref": "DATEXII_3_SituationPublicationLight.json#/definitions/Coordinate"
                                },
                                "line": {
                                        "title": "line",
                                        "properties": {
                                                "gmlLinearRing": {
                                                        "$ref":
"DATEXII_3_LocationReferencing.json#/definitions/GmlLinearRing"
                                                }
                                        }
                                },
                                "area": {
                                        "title": "area",
                                        "$ref": "DATEXII_3_LocationReferencing.json#/definitions/GmlMultiPolygon"
                                },
                                "_locationExtension": {
                                        "title": "_locationExtension",
                                        "$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
                                        "type": "object",
                                        "additionalItems": false
                                }
                        },
                        "required": [
                                "locationDescriptor"
                        ]
                },
                "RecurringTimePeriodOfDay": {
                        "title": "RecurringTimePeriodOfDay",
                        "type": "object",
                        "properties": {
                                "start": {
                                        "title": "start",
                                        "$ref": "DATEXII_3_Common.json#/definitions/Time"
                                },
                                "end": {
                                        "title": "end",
                                        "$ref": "DATEXII_3_Common.json#/definitions/Time"
                                },
                                "_recurringTimePeriodOfDayExtension": {
                                        "title": "_recurringTimePeriodOfDayExtension",
                                        "$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
                                        "type": "object",
                                        "additionalItems": false
                                }
                        },
                        "required": [
                                "start",
                                "end"
                        ]
                },
                "Schedule": {
                        "title": "Schedule",
                        "type": "object",
                        "properties": {
                                "startOfPeriod": {
                                        "title": "startOfPeriod",
                                        "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                                },
                                "endOfPeriod": {
                                        "title": "endOfPeriod",
                                        "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                                },
                                "recurringTimePeriodOfDay": {
                                        "title": "recurringTimePeriodOfDay",
                                        "type": "array",
                                        "items": {
                                                "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/RecurringTimePeriodOfDay"
                                        },
                                        "minItems": 0
                                },
                                "_scheduleExtension": {
                                        "title": "_scheduleExtension",
                                        "$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
                                        "type": "object",
                                        "additionalItems": false
                                }
                        },
                        "required": [
                                "startOfPeriod",
                                "endOfPeriod"
                        ]
                },
                "Situation": {
                        "title": "Situation",
                        "type": "object",
                        "properties": {
                                "id": {
                                        "type": "string"
                                },
                                "versionTime": {
                                        "title": "versionTime",
                                        "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                                },
                                "situationRecord": {
                                        "title": "situationRecord",
                                        "type": "array",
                                        "items": {
                                                "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/SituationRecord"
                                        },
                                        "minItems": 0
                                },
                                "_situationExtension": {
                                        "title": "_situationExtension",
                                        "$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
                                        "type": "object",
                                        "additionalItems": false
                                }
                        },
                        "required": [
                                "id",
                                "versionTime"
                        ]
                },
                "SituationPublicationLight": {
                        "title": "SituationPublicationLight",
                        "type": "object",
                        "properties": {
                                "situation": {
```

```
                    "title": "situation",
                    "type": "array",
                    "items": {
                        "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/Situation"
                    },
                    "minItems": 0
                },
                "_situationPublicationLightExtension": {
                    "title": "_situationPublicationLightExtension",
                    "$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
                    "type": "object",
                    "additionalItems": false
                }
            },
            "oneOf": [
                {
                    "$ref": "DATEXII_3_Common.json#/definitions/PayloadPublication"
                }
            ]
        },
        "SituationRecord": {
            "title": "SituationRecord",
            "type": "object",
            "properties": {
                "id": {
                    "type": "string"
                },
                "creationTime": {
                    "title": "creationTime",
                    "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                },
                "versionTime": {
                    "title": "versionTime",
                    "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                },
                "startTime": {
                    "title": "startTime",
                    "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                },
                "endTime": {
                    "title": "endTime",
                    "$ref": "DATEXII_3_Common.json#/definitions/DateTime"
                },
                "type": {
                    "title": "type",
                    "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/_SituationRecordTypeEnum"
                },
                "detailedType": {
                    "title": "detailedType",
                    "$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/_DetailedTypeEnum"
                },
                "detailedTypeText": {
                    "title": "detailedTypeText",
                    "$ref": "DATEXII_3_Common.json#/definitions/String"
                },
                "severity": {
                    "title": "severity",
                    "$ref": "DATEXII_3_Common.json#/definitions/String"
                },
                "safetyRelatedMessage": {
                    "title": "safetyRelatedMessage",
                    "$ref": "DATEXII_3_Common.json#/definitions/Boolean"
                },
                "sourceName": {
                    "title": "sourceName",
                    "$ref": "DATEXII_3_Common.json#/definitions/String"
                },
                "generalPublicComment": {
                    "title": "generalPublicComment",
                    "$ref": "DATEXII_3_Common.json#/definitions/String"
                },
                "managedCause": {
                    "title": "managedCause",
                    "$ref": "DATEXII_3_Common.json#/definitions/VersionedReference"
                },
                "numberOfLanesResticted": {
```

```
					"title": "numberOfLanesResticted",
					"$ref": "DATEXII_3_Common.json#/definitions/Integer"
				},
				"temporaryLimitText": {
					"title": "temporaryLimitText",
					"type": "array",
					"items": {
						"$ref": "DATEXII_3_Common.json#/definitions/String"
					},
					"minItems": 0
				},
				"trafficRestrictionTypeText": {
					"title": "trafficRestrictionTypeText",
					"$ref": "DATEXII_3_Common.json#/definitions/String"
				},
				"schedule": {
					"title": "schedule",
					"type": "array",
					"items": {
						"$ref":
"DATEXII_3_SituationPublicationLight.json#/definitions/Schedule"
					},
					"minItems": 0
				},
				"location": {
					"title": "location",
					"$ref": "DATEXII_3_SituationPublicationLight.json#/definitions/Location"
				},
				"_situationRecordExtension": {
					"title": "_situationRecordExtension",
					"$ref": "DATEXII_3_Common.json#/definitions/_ExtensionType",
					"type": "object",
					"additionalItems": false
				}
			},
			"required": [
				"id",
				"creationTime",
				"versionTime",
				"startTime",
				"type",
				"detailedType",
				"detailedTypeText",
				"severity",
				"safetyRelatedMessage",
				"sourceName",
				"generalPublicComment"
			]
		},
		"SituationRecordTypeEnum": {
			"title": "SituationRecordTypeEnum",
			"type": "string",
			"enum": [
				"abnormalTraffic",
				"accident",
				"animalPresenceObstruction",
				"authorityOperation",
				"disturbanceActivity",
				"environmentalObstruction",
				"equipmentOrSystemFault",
				"generalInstructionOrMessageToRoadUsers",
				"generalNetworkManagement",
				"generalObstruction",
				"infrastructureDamageObstruction",
				"nonWeatherRelatedRoadConditions",
				"poorEnvironmentConditions",
				"publicEvent",
				"reroutingManagement",
				"roadOperatorServiceDisruption",
				"roadOrCarriagewayOrLaneManagement",
				"roadsideAssistance",
				"serviceDisruption",
				"speedManagement",
				"transitInformation",
				"vehicleObstruction",
				"weatherRelatedRoadConditions",
				"winterDrivingManagement",
				"maintenanceWorks",
				"constructionWorks",
                                "_extended"
                        ]
                }
        }
}
```

JSON Example

The following is an example message containing one accident.

```
{
        "payload": {"modelBaseVersion":"3",
                "situationPublicationLight": {
                        "publicationTime": "2017-12-24T10:00:00+02:00",
                        "publicationCreator": {
                                "country": "se",
                                "nationalIdentifier": "STA"
                        },
                        "situation": [
                                {
                                        "id": "1234","versionTime": "2017-12-24T10:00:00Z",

                                        "headerInformation": {
                                                "informationStatus": "real",
                                                "confidentiality": "noRestriction"
                                        },
                                        "situationRecord": [
                                                {
                                                        "creationTime": "2017-12-24T10:00:00Z",
                                                        "versionTime": "2017-12-24T10:00:00Z",
                                                        "startTime": "2017-12-24T10:00:00Z",
                                                        "endTime": "2017-12-29T12:00:00Z",
                                                        "type": {"value": "accident" },
                                                        "detailedType": {"value": "unprotectedAccidentArea"},
                                                        "detailedTypeText": "Olycka",
                                                        "id": "1234566",
                                                        "severity": "High",
                                                        "safetyRelatedMessage": true,
                                                        "generalPublicComment": "Detta är en Olycka",
                                                        "sourceName": "Trafikverket",
                                                        "location": {"locationDescriptor": "Väg 70, Borlängt till
Falun vid Källviksbacken",
                                                                "roadName": "Väg 70", "roadNumber":
"70","locationForDisplay": {"pointCoordinates": {"latitude": 59.2342344, "longitude": 16.32455}}
                                                        },
                                                        "temporaryLimitText": ["Hastighet: 50 km/h"],

                                                        "trafficRestrictionTypeText": "Ett körfält blockerat",
                                                        "numberOfLanesResticted": "1"



                                                }
                                        ]
                                }
                        ]
                }
        }
}
```

Exchange

Exchange in an API is usually exposed as a simple http REST endpoint. Either as HTTP GET or HTTP POST.

- HTTP GET can be used if parameters could be specified be parameters. Eg. http://datex.org/getapi?param1&param2..
- HTTP POST is used if some more complex parameters has to be specified. With POST a document containing parameters can be send to the endpoint.

Commonly the format of the response can be specified in the HTTP header e.g.

- `Accept: application/json for JSON`
- `Accept: application/xml for XML`

An alternative way is to specify the format as part of the URL e.g.

http://datex.org/getapi.xml

Filtering

To get fast response times the amount of data received should be limited. By supporting a query or filter feature, the client can tailor and limit the response to what is really needed. The client should be able to specify conditions and what fields to receive in the response.

To query or filter of the content of an endpoint it is commonly done as

- Publishing different content on different URL e.g. http://datex.org/getAccidents to get accidents and http://datex.org/getX to get something related to X.
- Filtering, by specifying parameters either as query string or as a document/message.

Filtering usually limits the amount of data to receive. E.g. of parameters

- Severity = High
- Latitude and longitude in combination with a radius would limit the result to elements that is inside the area specified by the coordinate and radius.
-

Version management

One way to handle different version is to have different versions published on different URL's e.g http://datex.org/v1.1/getX has version v1.1 published.

Example from API in Sweden

Request:

```
<REQUEST>
  <LOGIN authenticationkey="SomeAuthenticationKey" />
  <QUERY objecttype="SomeObjectType" limit="10">
  <FILTER>
    <EQ name="SomeDataField" value="2" />
  </FILTER>
  </QUERY>
</REQUEST>
```

Request is send via HTTP POST.

In the URL version and format can be specified e.g.

http://api.trafikinfo.trafikverket.se/v/*[version]*/data.*[format]*

format can be either xml or json.

To access the API a authentication key is required.

For FILTER there are several options e.g.

`<EQ/>`	”Equals”
`<EXISTS/>`	”Exists”
`<GT/>`	”Greater Than”,

The table above is not complete, more options exists.

Filter by geometry can also be done e.g.

```
<WITHIN name="Geometry.SWEREF99TM" shape="box" value="276192 6576098, 28096
7 6579969"/>
```

Response:

Example XML:

```
<RESPONSE>
  <RESULT>
```

```
    <SomeObjectType>
      <Id>123<Id>
      <Name>Adam<Name>
    </SomeObjectType>
    <SomeObjectType>
      <Id>345<Id>
      <Name>Bertil<Name>
    </SomeObjectType>
  </RESULT>
</RESPONSE>
```

Example JSON:

```
{
   "RESPONSE":{
      "RESULT":[
         {
            "SomeObjectType":[
               {
                  "Id":"123",
                  "Name":"Adam"
               },
               {
                  "Id":"456",
                  "Name":"Bertil"
               }
            ]
         }
      ]
   }
}
```

Publications

Additional future publications should be those that support the needs of an app user.

In consideration would for example be

- Parking
- Charging/fueling
- Traveltimes
- VMS
- Simplified DATEX II METR publication when ready

Observations and considerations

- WKT (Well known text)
  - is used in the Swedish API and WKT seems to be a simple and commonly used way to transport a geometry as a string. It's used by most databases with spatial support. But is a new location representing method and we decided to reuse the ones we have in DATEX II.
- The detailedType,
  - that we now have as an enum, could also be just a simple string. We will lose the validation but the schema would be simpler and no need to extend it to use whatever "detailedType" you want.

Appendix A