CEN/TC 278

Date: 2017-10

TC 278 TS 16614-1

CEN/TC 278

Secretariat: NEN

Public transport — Network and Timetable Exchange (NeTEx) — Part 1: Public transport network topology exchange format

NeTEx — Haupt-Element — Teil 1: Teil-Titel

Transport Public — Echanges des informations planifiées (NeTEx) — Partie 1 : Topologie du réseau

ICS:

Descriptors:

Final DRAFT for NeTEx v1.1

2017.08.08	CD – DRAFT V1 with embedded diagrams	V38
2017.10.10	NJSK –Revised Draft Updated UML Diagrams	V38-nk1
2017.12.01	NJSK –Revised Draft Apply CR changes and correct from CD anomaly list.	V38_nk02
2018.07	Final update and cleaning for TC Inquiry	Review Final1
2018.12	Management of TC Inquiry comments	TCI_Editorial
2019.03.17	CD Updates for review CRS.	Final_1
2019.04.18	NJSK Remaining CRs and comments and update UML. Extensive Corrections.	Final_2
2019.04.24	NJSK Further minor corrections	Final_4

Document type: Technical Specification
Document subtype:
Document stage: Formal Vote
Document language: E
C:\Users\nick.knowles\Google Drive\PROJECTSDRIVE\A_NetEx\2019.02.18-NX -Review\docs\prCEN TS 16614-1 FV (E)-2019-Final-Draft.docx STD Version 2.5a

Contents

Page

Foreword

This document (TC 278 TS 16614-1) has been prepared by

Technical Committee CEN/TC 278 “Road transport and traffic telematics”, the secretariat of which is held by NEN.

This document is currently submitted to the Formal Vote.

This document presents Part 1 of the European Technical Specification known as “NeTEx”. NeTEx provides a framework for specifying communications and data exchange protocols for organisations wishing to exchange scheduled Information relating to public transport operations.

This technical specification is made up of three parts defining a single European Standard series, which provides a complete exchange format for public transport networks, timetable description and fare information.

— Part 1 is the description of the public transport network topology exchange format. It also contains use cases shared with part 2, and modelling rules and the description of a framework shared by all parts.

— Part 2 is the description of the scheduled timetables exchange format.

— Part 3 is the description of the fare information exchange format.[1]

Part 1 is fully standalone, and part 2 and 3 rely on part 1.

The XML schema can be downloaded from http://netex-cen.eu, along with available guidance on its use, example XML files, and case studies of national and local deployments.

NOTE This document is highly technical, and a special care has been taken on keeping the text readable. This has been done through a set of editorial rules enhancing usual CEN writing rules:

— To avoid confusion with usual wording, Transmodel terms are in capital letters (JOURNEY PATTERN for example).

— To avoid confusion with usual wording, attributes names are in bold/italic style and use camelcase style with no spaces (***JourneyPattern*** for example).

— To avoid confusion with usual wording, attributes types are in italic style and use camelcase style with no spaces (*TypeOfEntity* for example).

This document describes the NeTEx 1.1 schema which includes corrrections and enancements to the original 1.0 schema.

[1] Currently under development

Introduction

Public transport services rely increasingly on information systems to ensure reliable, efficient operation and widely accessible, accurate passenger information. These systems are used for a range of specific purposes: setting schedules and timetables; managing vehicle fleets; issuing tickets and receipts; providing real-time information on service running, and so on.

This European Technical Specification specifies a Network and Timetable Exchange (NeTEx) standard for Public Transport. It is intended to be used to exchange data relating to scheduled public transport between the systems of PT organisations. It can also be seen as complementary to the SIRI (Service Interface for Real-time Information) standard, as SIRI needs a prior exchange of reference data from NeTEx's scope to provide the necessary context for the subsequent exchange of real-time data.

Well-defined and open interfaces have a crucial role in improving the economic and technical viability of Public Transport Information Systems of all kinds. Using standardised interfaces, systems can be implemented as discrete pluggable modules that can be chosen from a wide variety of suppliers in a competitive market, rather than as monolithic proprietary systems from a single supplier. Interfaces also allow the systematic automated testing of each functional module, vital for managing the complexity of increasing large and dynamic systems. Furthermore, individual functional modules can be replaced or evolved, without unexpected breakages of obscurely dependent function.

This standard will improve a number of features of public transport information and service management: Interoperability – the standard will facilitate interoperability between information processing systems of the transport operators by: (i) introducing common architectures for message exchange; (ii) introducing a modular set of compatible information services for real-time vehicle information; (iii) using common data models and schemas for the messages exchanged for each service; and (iv) introducing a consistent approach to data management.

Technical advantages include the following: a modular reusing of a common communication layer shared with SIRI for all the various technical services enables cost-effective implementations, and makes the standard readily extensible in future.

1 Scope

1.1 General

NeTEx is dedicated to the exchange of scheduled data (network, timetable and fare information). It is based on Transmodel V6 (EN 12896) and SIRI (CEN/TS 15531-4/5 and EN 15531-1/2/3) and supports the exchange of information of relevance for passenger information about public transport services and also for running Automated Vehicle Monitoring Systems (AVMS).

NOTE Many NeTEx concepts are taken directly from Transmodel; the definitions and explanation of these concepts are extracted directly from the respective standard and reused in NeTEx, sometimes with adaptions in order to fit the NeTEx context.

Although the data exchanges targeted by NeTEx are predominantly oriented towards provisioning passenger information systems and AVMS with data from transit scheduling systems, it is not restricted to this purpose and NeTEx can also provide an effective solution to many other use cases for transport data exchange.

1.2 Transport modes

All mass public transport modes are taken into account by NeTEx, including train, bus, coach, metro, tramway, ferry, and their submodes. It is possible to describe airports and air journeys, but there has not been any specific consideration of any additional requirements that apply specifically to air transport.

1.3 Compatibility with existing standards and recommendations

Concepts covered in NeTEx that relate in particular to long-distance train travel include; rail operators and related organizations; stations and related equipment; journey coupling and journey parts; train composition and facilities; planned passing times; timetable versions and validity conditions.

In the case of long distance train the NeTEx takes into account the requirements formulated by the ERA (European Rail Agency) – TAP/TSI (Telematics Applications for Passenger/ Technical Specification for Interoperability, entered into force on 13 May 2011 as the Commission Regulation (EU) No 454/2011), based on UIC directives.

As regards the other exchange protocols, a formal compatibility is ensured with TransXChange (UK), VDV 452 (Germany), NEPTUNE (France), UIC Leaflet, BISON (The Netherlands) and NOPTIS (Nordic Public Transport Interface Standard).

The data exchange is possible either through dedicated web services, through data file exchanges, or using the SIRI exchange protocol as described in part 2 of the SIRI documentation.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

EN 15531-1, *Public transport - Service interface for real-time information relating to public transport operations - Part 1: Context and framework*

EN 15531-2, *Public transport - Service interface for real-time information relating to public transport operations - Part 2: Communications infrastructure*

EN 15531-3, *Public transport - Service interface for real-time information relating to public transport operations - Part 3: Functional service interfaces*

CEN/TS 15531-4, *Public transport - Service interface for real-time information relating to public transport operations - Part 4: Functional service interfaces: Facility Monitoring*

CEN/TS 15531-5, *Public transport - Service interface for real-time information relating to public transport operations - Part 5: Functional service interfaces - Situation Exchange*

EN 12896 series, Road transport and traffic telematics - Public transport - Reference data model

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

NOTE A lot of definitions are shared with Transmodel (EN 12896): special attention was paid to the consistency of definitions, keeping exactly the same wording. The italic bracket name at the beginning of the definition is a package name that will help the reader to find the related concept in the UML data model.

3.1
access
(Generic Place MODEL)
The physical (spatial) possibility for a passenger to access or leave the public transport system. This link may be used during a trip for the walking movement of a passenger from a PLACE (origin of the trip) to a STOP POINT (origin of the PT TRIP), or for the walking movement from a STOP POINT (destination of the PT TRIP) to a PLACE (destination of the trip).

3.2
access end
(Generic Place MODEL)
Origin or destination end of an ACCESS link. May indicate a MODE, POINT and PLACE.

3.3
access mode
(Reusable Transport Mode MODEL)
A characterisation of the passenger movement according to the means of transport different from public transport (e.g. walk, bicycle, etc).

3.4
access space
(Stop Place MODEL)
A passenger area within a STOP PLACE such as a concourse or booking hall, immigration hall or security area that is accessible by passengers, but without a direct access to vehicles. Direct access to a VEHICLE is always from a QUAY and/or BOARDING POSITION. An ACCESS SPACE may be a Room, Hall, Concourse, Corridor, or bounded open space within a STOP PLACE.

3.5
access zone
(Site MODEL)
A ZONE for which the duration to cover any ACCESS LINK to a particular STOP POINT is the same.

3.6
accessibility assessment
(Accessibility MODEL)
The accessibility characteristics of an entity used by passengers such as a STOP PLACE or a STOP PLACE COMPONENT. Described by ACCESSIBILITY LIMITATIONs, and/or a set of SUITABILITies.

3.7
accessibility limitation
(Accessibility MODEL)
A categorisation of the accessibility characteristics of a SITE, e.g. a STOP PLACE or a STOP PLACE COMPONENT to indicate its usability by passengers with specific needs, for example, those needing wheelchair access, step-free access or wanting to avoid confined spaces such as lifts. A small number of well-defined

categories are used that are chosen to allow the consistent capture of data and the efficient computation of routes for different classes of user.

3.8
accommodation
(Facility MODEL)
A combination of accommodation characteristics available on a service, e.g. First Class Couchette with shower and 2 bunks.

3.9
activated equipment
(Activation MODEL)
An equipment activated by the passage of a vehicle at an ACTIVATION POINT or on an ACTIVATION LINK.

3.10
activation assignment
(Activation MODEL)
An assignment of an ACTIVATION POINT/LINK to an ACTIVATED EQUIPMENT related on its turn to a TRAFFIC CONTROL POINT. The considered ACTIVATION POINT/LINK will be used to influence the control process for that TRAFFIC CONTROL POINT (e.g. to fix priorities as regards the processing of competing requests from different ACTIVATION POINTs/LINKs).

3.11
activation link
(Activation MODEL)
A LINK where a control process is activated when a vehicle passes it.

3.12
activation point
(Activation MODEL)
A POINT where a control process is activated when a vehicle passes it. Equipment may be needed for the activation.

3.13
actual vehicle equipment
(Actual Vehicle Equipment MODEL)
An item of equipment of a particular type in an individual VEHICLE.

3.14
address
(Topographic MODEL)
An address of a PLACE.

3.15
administrative zone
(Generic Organisation MODEL)
The area of a district, a region, a city, a municipality, or other area with which an ORGANIZATION has a RESPONSIBILITY ROLE.

3.16
allowed line direction
(Route MODEL)
An allowed DIRECTION that can be used on a given ROUTE. This can be used to validate the selection of allowed values.

3.17
alternative name
(Site MODEL)
Alternative name for the entity.

3.18
assistance service
(Local Service Equipment MODEL)
Specialisation of LOCAL SERVICE for ASSISTANCE providing information like language, accessibility trained staff, etc.

3.19
authority
(Transport Organisations MODEL)
The organisation under which the responsibility of organising the transport service in a certain area is placed.

3.20
availability condition
(Reusable Availability MODEL)
A VALIDITY CONDITION expressed in terms of temporal parameters and referring to DAY TYPEs.

3.21
beacon point
(Activation MODEL)
A POINT where a beacon or similar device to support the automatic detection of vehicles passing by is located.

3.22
block
(Vehicle Service MODEL)
The work of a vehicle from the time it leaves a PARKING POINT after parking until its next return to park at a PARKING POINT. Any subsequent departure from a PARKING POINT after parking marks the start of a new BLOCK. The period of a BLOCK has to be covered by DUTies.

3.23
block part
(Vehicle Service MODEL)
Part of a BLOCK corresponding to the different JOURNEY PARTs of the VEHICLE JOURNEYs in a BLOCK.

3.24
boarding position
(Stop Place MODEL)
A location within a QUAY from which passengers may directly board, or onto which passengers may directly alight from a VEHICLE.

3.25
booking arrangements
(ServiceRestrictions MODEL)
Booking arrangements for FLEXIBLE LINE.

3.26
check constraint
(Check Constraint MODEL)
Characteristics of a process that takes place at a SITE COMPONENT, such as check-in, security screening, ticket control or immigration, that may potentially incur a time penalty that should be allowed for when journey planning.

3.27
check constraint delay
(Check Constraint MODEL)
Time penalty associated with a CHECK CONSTRAINT.

3.28
check constraint throughput
(Check Constraint MODEL)

Throughput of a CHECK CONSTRAINT: the number of passengers who can pass through it in a specified interval.

3.29
class in frame
(Generic Version MODEL)
The different CLASSEes IN REPOSITORY which can be relevant for corresponding VERSION FRAMEs.

3.30
class in repository
(Generic Entity MODEL)
Any ENTITY name belonging to the repository. E.g. DAY TYPE, PROPERTY OF DAY, TIME BAND, VEHICLE TYPE, DUTY, etc. are relevant instances of CLASS IN REPOSITORY in the context of Version Management.

3.31
class of use
(Fare-Related Service Restriction MODEL)
A classification of fare and other service classes by category of user entitled to use them.

3.32
common section
(Generic Point & Link MODEL)
A part of a public transport network where the ROUTEs of several JOURNEY PATTERNs are going in parallel and where the synchronisation of SERVICE JOURNEYs may be planned and controlled with respect to commonly used LINKs and STOP POINTs. COMMON SECTIONs are defined arbitrarily and need not cover the total lengths of topologically bundled sections.

3.33
communication service
(Local Commercial Service MODEL)
Specialisation of LOCAL SERVICE dedicated to communication services.

3.34
complaints service
(Local Service Equipment MODEL)
Specialisation of CUSTOMER SERVICE for COMPLAINTs.

3.35
complex feature
(Generic Zone and Feature MODEL)
An aggregate of SIMPLE FEATUREs and/or other COMPLEX FEATUREs; e.g. a STOP AREA: combination of STOP POINTS; a train station: combination of SIMPLE FEATUREs (POINTs, LINKs) and COMPLEX FEATUREs (STOP AREAs).

3.36
complex feature projection
(Generic Projection MODEL)
An oriented correspondence: from one COMPLEX FEATURE in the source layer, onto an entity in a target layer: e.g. POINT, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

3.37
compound block
(Vehicle Service MODEL)
The work of a vehicle during the time it is coupled to another vehicle.

3.38
compound train
(Train MODEL)
A VEHICLE TYPE composed of a sequence of more than one vehicles of the type TRAIN.

3.39
connection
(Service Pattern MODEL)
The physical (spatial) possibility for a passenger to change from one public transport vehicle to another to continue the trip, determined by two SCHEDULED STOP POINTs. Different times may be necessary to cover the link between these points, depending on the kind of passenger.

3.40
connection end
(Service Pattern MODEL)
One end of a CONNECTION.

3.41
contact details
(Generic Organisation MODEL)
Contact details for ORGANISATION for public use.

3.42
control centre
(Transport Organisations MODEL)
An ORGANISATION PART for an operational team who is responsible for issuing commands to control the services.

3.43
country
(Topographic MODEL)
A jurisdictional geographic boundary. A COUNTRY normally has a two character IANA identifier.

3.44
coupled journey
(Coupled Journey MODEL)
A complete journey operated by a coupled train, composed of two or more VEHICLE JOURNEYs remaining coupled together all along a JOURNEY PATTERN. A COUPLED JOURNEY may be viewed as a single VEHICLE JOURNEY.

3.45
course of journeys
(Vehicle Service MODEL)
A part of a BLOCK composed of consecutive VEHICLE JOURNEYs defined for the same DAY TYPE, all operated on the same LINE.

3.46
crew base
(Vehicle & Crew Point MODEL)
A place where operating EMPLOYEEs (e.g. drivers) report on and register their work.

3.47
crossing equipment
(Access Equipment)
Specialisation of PLACE ACCESS EQUIPMENT for CROSSING EQUIPMENTs (zebra, pedestrian lights, acoustic device sensors, tactile guide strips, etc.).

3.48
customer service
(Local Service Equipment MODEL)
Generic specialisation of LOCAL SERVICE for CUSTOMER SERVICEs (lost properties, meeting point, complaints, etc.).

3.49
cycle parking equipment
(Parking Equipment MODEL)
A specialisation of PLACE EQUIPMENT describing CYCLE PARKING EQUIPMENT.

3.50
data source
(Generic Responsibility MODEL)
The DATA SOURCE identifies the system which has produced the data. References to a data source are useful in an interoperated computer system.

3.51
dated passing time
(Passing Times MODEL)
A PASSING TIME on a particular OPERATING DAY.

3.52
dated vehicle journey
(Dated Journey MODEL)
A particular journey of a vehicle on a particular OPERATING DAY including all modifications possibly decided by the control staff.

3.53
day of week
(Service Calendar MODEL)
A particular week day (from Monday to Sunday).

3.54
day type
(Service Calendar MODEL)
A type of day characterised by one or more properties which affect public transport operation. For example: weekday in school holidays.

3.55
day type assignment
(Service Calendar MODEL)
The assignment of operational characteristics, expressed by DAY TYPEs, to particular OPERATING DAYs within a SERVICE CALENDAR.

3.56
dead run
(Vehicle Journey MODEL)
A non-service VEHICLE JOURNEY.

3.57
dead run pattern
(Journey Pattern MODEL)
A JOURNEY PATTERN to be used for DEAD RUNs.

3.58
default connection
(Service Connection MODEL)
The physical (spatial) possibility for a passenger to change from one public transport vehicle to another to continue the trip.

It specifies default times to be used to change from one mode of transport to another at an area or national level as specified by a TOPOGRAPHIC PLACE, STOP AREA or SITE ELEMENT. It may be restricted to a specific MODE or OPERATOR or only apply in a particular direction of transfer, e.g. bus to rail may have a different time than rail to bus.

3.59
default connection end
(Service Connection MODEL)
One end of a DEFAULT CONNECTION.

3.60
default dead run run time
(Time Demand Times MODEL)
The time taken to traverse a TIMING LINK during a DEAD RUN, for a specified TIME DEMAND TYPE. This time may be superseded by the JOURNEY PATTERN RUN TIME or VEHICLE JOURNEY RUN TIME if these exist.

3.61
default interchange
(Interchange MODEL)
A quality parameter fixing the acceptable duration (standard and maximum) for an INTERCHANGE to be planned between two SCHEDULED STOP POINTs. This parameter will be used to control whether any two VEHICLE JOURNEYs serving those points may be in connection.

3.62
default service journey run time
(Time Demand Times MODEL)
The default time taken by a vehicle to traverse a TIMING LINK during a SERVICE JOURNEY, for a specified TIME DEMAND TYPE. This time may be superseded by the JOURNEY PATTERN RUN TIME or VEHICLE JOURNEY RUN TIME if these exist.

3.63
delivery variant
(Notice MODEL)
A variant text of a NOTICE for use in a specific media or delivery channel (voice, printed material, etc).

3.64
department
(Generic Organisation MODEL)
An ORGANIZATION PART specific to a purpose and/or organisational structure.

3.65
destination display
(Route MODEL)
An advertised destination of a specific JOURNEY PATTERN, usually displayed on a headsign or at other on-board locations.

3.66
destination display variant
(Route MODEL)
A variant, depending on the delivery type, of an advertised destination of a specific JOURNEY PATTERN, usually displayed on a headsign or at other on-board locations.

3.67
direction
(Route MODEL)
A classification for the general orientation of ROUTEs.

3.68
display assignment
(Passenger Information Equipment MODEL)
The assignment of one STOP POINT and one JOURNEY PATTERN to a PASSENGER INFORMATION EQUIPMENT specifying that information on the STOP POINT and the JOURNEY PATTERN will be provided (e.g. displayed, printed).

3.69
dynamic stop assignment
(Stop Assignment MODEL)
The dynamic association of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) with the next available STOP PLACE, QUAY or BOARDING POSITION within a STOP PLACE.

3.70
encumbrance need
(Accessibility MODEL)
A specific USER NEED, i.e. a requirement of a passenger travelling with luggage, animal or any other object requiring special arrangements to access public transport.

3.71
entity
(Generic Entity MODEL)
Any data instance to be managed in an operational Version Management System. When several data sources coexist (multimodality and/or interoperability), an ENTITY has to be related to a given DATA SOURCE in which it is defined.

3.72
entity in version
(Generic Version MODEL)
The ENTITY associated to a given VERSION.

3.73
entrance
(Site MODEL)
A physical entrance or exit to/from a SITE. May be a door, barrier, gate or other recognizable point of access.

3.74
entrance equipment
(Access Equipment)
Specialisation of PLACE ACCESS EQUIPMENT for ENTRANCEs (door, barrier, revolving door, etc.).

3.75
equipment
(Generic Equipment MODEL)
An item of equipment installed either fixed (PLACE EQUIPMENT) or on-board vehicles (VEHICLE EQUIPMENT). A service (LOCAL SERVICE such as LEFT LUGGAGE, TICKETING SERVICE) is considered as immaterial equipment as well.

3.76
equipment place
(Generic Equipment MODEL)
A SITE COMPONENT containing EQUIPMENT

3.77
equipment position
(Generic Equipment MODEL)
The precise position within an EQUIPMENT PLACE where particular equipment is placed.

3.78
escalator equipment
(Stair Equipment)
Specialisation of STAIR EQUIPMENT for ESCALATORs.

3.79
facility
(Facility MODEL)
A named amenity available to the public at a SITE or on a SERVICE. A facility has no further properties other than a name. An EQUIPMENT or LOCAL SERVICE is used to describe the further properties provided as part of particular facility.

3.80
facility requirement
(Vehicle Type MODEL)
A classification of public transport vehicles according to the facilities available on the vehicle.

3.81
facility set
(Facility MODEL)
A set of FACILITIES that may be associated with an ENTITY and subject to a specific VALIDITY CONDITION.

3.82
flexible area
(Flexible Stop Place MODEL)
Specialisation of a FLEXIBLE QUAY (which is abstract) to identify what is the catchment area for a flexible service (so that a stop finder can find the nearest available types of transport). It is a named zone visited by a particular mode of transport. It is part of the SITE data set rather than the service data set, since it can be defined and exists independently of an actual service.

3.83
flexible line
(Flexible Network MODEL)
Specialisation of LINE for flexible service. As all the service on a LINE may not all be flexible, flexibility itself is described at JOURNEY PATTERN level (meaning that a separate JOURNEY PATTERN is needed for each type of flexibility available for the line).

Types of flexible services are :

- Virtual line service

- Flexible service with main route

- Corridor service

- Fixed stop area-wide flexible service

- Free area-wide flexible service

- Mixed types of flexible service

- Mixed type of flexible and regular services.

3.84
flexible link properties
(Flexible Network MODEL)
Set of properties describing the flexible characteristics of a LINK.

A composition is used with LINK in order to avoid multiple inheritance and a type explosion of link subtypes.

3.85
flexible point properties
(Flexible Network MODEL)

Set of characteristics describing the possible flexibility of POINTs. A composition is used with POINT in order to avoid multiple inheritance.

3.86
flexible quay
(Flexible Stop Place MODEL)
A physical ZONE such as a section of a road where a flexible service is available on demand. The existence of the zone makes the services visible to journey planners looking for available services for an area.

3.87
flexible route
(Flexible Network MODEL)
Specialisation of ROUTE for flexible service. May include both point and zonal areas and ordered and unordered sections.

3.88
flexible service properties
(Flexible Service MODEL)
Additional characteristics of flexible service. A service may be partly fixed, partly flexible.

3.89
flexible stop assignment
(Flexible Stop Place MODEL)
The allocation of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) to a specific FLEXIBLE STOP PLACE, and also possibly a FLEXIBLE AREA or HAIL AND RIDE AREA. May be subject to a VALIDITY CONDITION.

3.90
flexible stop place
(Flexible Stop Place MODEL)
A specialisation of the STOP PLACE describing a stop of a FLEXIBLE SERVICE. It may be composed of FLEXIBLE AREAs or HAIL AND RIDE AREAs identifying the catchment areas for flexible services (when they use areas or flexible quays). Some FLEXIBLE SERVICE also use regular STOP PLACEs for their stops. When assigned to a SCHEDULED STOP POINT the corresponding SCHEDULED STOP POINT is supposed to be a ZONE (the centroid point of the ZONE being the SCHEDULED STOP POINT).

3.91
garage
(Vehicle & Crew Point MODEL)
A facility used for parking and maintaining vehicles. PARKING POINTs in a GARAGE are called GARAGE POINTs.

3.92
garage point
(Vehicle & Crew Point MODEL)
A subtype of PARKING POINT located in a GARAGE.

3.93
group of entities
(Generic Grouping MODEL)
A set of ENTITies grouped together according to a PURPOSE OF GROUPING, e.g. grouping of stops known to the public by a common name.

3.94
group of lines
(Route MODEL)
A grouping of lines which will be commonly referenced for a specific purpose.

3.95
group of link sequences
(Generic Point & Link Sequence MODEL)
A grouping of LINK SEQUENCEs.

3.96
group of links
(Generic Point & Link MODEL)
A grouping of LINKs. E.g. one GROUP OF LINKs may be managed by a same AUTHORITY.

3.97
group of operators
(Transport Organisations MODEL)
A group of OPERATORs having for instance common schemes for fare collection or passenger information.

3.98
group of points
(Generic Point & Link MODEL)
A grouping of POINTs. The STOP AREA represents one of the most significant GROUPs OF POINTS.

3.99
group of services
(Service Journey MODEL)
A group of SERVICEs, often known to its users by a name or a number.

3.100
group of timebands
(Service Calendar MODEL)
A grouping of TIME BANDs.

3.101
group of timing links
(Time Demand Type MODEL)
A set of TIMING LINKs grouped together according to the similarity of TIME BANDs which are relevant to them. There may be a GROUP OF TIMING LINKS which covers all TIMING LINKs, for use when different GROUPs OF TIMING LINKS are not needed.

3.102
hail and ride area
(Flexible Stop Place MODEL)
Specialisation of a FLEXIBLE QUAY to identify what is the catchment zone for a hail and ride service (so that a stop finder can find the nearest available types of transport). It is a named zone visited by a particular mode of transport and may be designated by a start point and end point on the road.

NOTE 1 to entry: It is part of the site data set rather than the service data set, since it can be defined and exists independently of an actual service.

3.103
heading sign
(Sign Equipment)
Specialisation of SIGN EQUIPMENT for headings providing information like direction name, line name, etc.

3.104
headway interval
(Vehicle Journey Times MODEL)
A time interval or a duration defining a headway period and characterizing HEADWAY JOURNEY GROUP (e.g. every 10 min, every 4-6 min).

3.105
headway journey group
(Vehicle Journey Times MODEL)
A group of VEHICLE JOURNEYs following the same JOURNEY PATTERN having the same HEADWAY INTERVAL between a specified start and end time (for example, every 10 min). This is especially useful for passenger information.

3.106
hire service
(Local Commercial Service MODEL)
Specialisation of LOCAL SERVICE dedicated to hire services (e.g. cycle hire, car hire).

3.107
infrastructure link
(Infrastructure Network MODEL)
A super-type including all LINKs of the physical network (e.g. RAILWAY ELEMENT).

3.108
infrastructure point
(Infrastructure Network MODEL)
A super-type including all POINTs of the physical network (e.g. RAILWAY JUNCTION).

3.109
installed equipment
(Generic Equipment MODEL)
An item of equipment either fixed (PLACE EQUIPMENT) or on board i.e. associated with vehicles. This equipment is materialised as opposed to a service (LOCAL SERVICE) considered as an immaterial equipment.

3.110
interchange
(Interchange MODEL)
The scheduled possibility for transfer of passengers between two SERVICE JOURNEYs at the same or different STOP POINTs.

3.111
interchange rule
(Interchange Rule MODEL)
Conditions for considering JOURNEYs to meet or not to meet, specified indirectly: by a particular MODE, DIRECTION or LINE. Such conditions may alternatively be specified directly, indicating the corresponding services. In this case they are either a SERVICE JOURNEY PATTERN INTERCHANGE or a SERVICE JOURNEY INTERCHANGE.

3.112
interchange rule parameter
(Interchange Rule MODEL)
Assignment of parameters characterising an INTERCHANGE RULE.

3.113
interchange rule timing
(Interchange Rule MODEL)
Timings for an INTERCHANGE RULE for a given TIME DEMAND TYPE or TIME BAND.

3.114
journey
(Vehicle Journey MODEL)
Common properties of VEHICLE JOURNEYs and SPECIAL SERVICEs, e.g. their link to accounting characteristics.

3.115
journey accounting
(Journey Accounting MODEL)
Parameters characterizing VEHICLE JOURNEYs or SPECIAL SERVICEs used for accounting purposes in particular in contracts between ORGANISATIONs.

3.116
journey frequency group
(Vehicle Journey Times MODEL)
A group of JOURNEYs defined in order to describe special behaviour like frequency based services or rhythmical services (runs all xxh10, xxh25 and xxh45, for example; this is especially useful for passenger information).

3.117
journey headway
(Journey Timing MODEL)
Headway interval information that is available for all the VEHICLE JOURNEYs running on the JOURNEY PATTERN for a given TIME DEMAND TYPE, at a given TIMING POINT. This is a default value that can be superseded by VEHICLE JOURNEY HEADWAY. This information shall be consistent with HEADWAY JOURNEY GROUP if available (HEADWAY JOURNEY GROUP being a more detailed way of describing headway services).

3.118
journey layover
(Journey Timing MODEL)
Time allowance at the end of each journey on a specified JOURNEY PATTERN, to allow for delays and for other purposes. This layover supersedes any global layover and may be superseded by a specific VEHICLE JOURNEY LAYOVER.

3.119
journey meeting
(Interchange MODEL)
A time constraint for one or several SERVICE JOURNEYs fixing interchanges between them and/or an external event (e.g. arrival or departure of a feeder line, opening time of the theatre, etc.).

3.120
journey part
(Coupled Journey MODEL)
A part of a VEHICLE JOURNEY created according to a specific functional purpose, for instance in situations when vehicle coupling or separating occurs.

3.121
journey part couple
(Coupled Journey MODEL)
Two JOURNEY PARTs of different VEHICLE JOURNEYs served simultaneously by a train set up by coupling their single vehicles.

3.122
journey pattern
(Journey Pattern MODEL)
An ordered list of SCHEDULED STOP POINTs and TIMING POINTs on a single ROUTE, describing the pattern of working for public transport vehicles. A JOURNEY PATTERN may pass through the same POINT more than once. The first point of a JOURNEY PATTERN is the origin. The last point is the destination.

3.123
journey pattern headway
(Journey Pattern Times MODEL)
Headway interval information that is available for all the VEHICLE JOURNEYs running on the JOURNEY PATTERN. This is a default value that can be superseded by the VEHICLE JOURNEY HEADWAY on a specific

journey. This information shall be consistent with HEADWAY JOURNEY GROUP if available (HEADWAY JOURNEY GROUP being a more detailed way of describing headway services).

3.124
journey pattern layover
(Journey Pattern Times MODEL)
Time allowance at the end of each journey on a specified JOURNEY PATTERN, to allow for delays and for other purposes. This layover supersedes any global layover and may be superseded by a specific VEHICLE JOURNEY LAYOVER.

3.125
journey pattern run time
(Journey Pattern Times MODEL)
The time taken to traverse a TIMING LINK in a particular JOURNEY PATTERN, for a specified TIME DEMAND TYPE. If it exists, it will override the DEFAULT SERVICE JOURNEY RUN TIME and DEFAULT DEAD RUN RUN TIME.

3.126
journey pattern wait time
(Journey Pattern Times MODEL)
The time a vehicle has to wait at a specific TIMING POINT IN JOURNEY PATTERN, for a specified TIME DEMAND TYPE. This wait time can be superseded by a VEHICLE JOURNEY WAIT TIME.

3.127
journey run time
(Journey Timing MODEL)
The time taken to traverse a TIMING LINK in a particular JOURNEY PATTERN, for a specified TIME DEMAND TYPE. If it exists, it will override the DEFAULT SERVICE JOURNEY RUN TIME and DEFAULT DEAD RUN RUN TIME.

3.128
journey timing
(Journey Timing MODEL)
A time-related information referring to journey timing whose value depends on the time of use and so can be associated with a TIME DEMAND TYPE, TIME BAND or OPERATIONAL CONTEXT.

3.129
journey wait time
(Journey Timing MODEL)
The time a vehicle has to wait at a specific TIMING POINT IN JOURNEY PATTERN, for a specified TIME DEMAND TYPE. This wait time can be superseded by a VEHICLE JOURNEY WAIT TIME.

3.130
left luggage service
(Local Service Equipment MODEL)
Specialisation of CUSTOMER SERVICE for left luggage (provides left luggage information like self-service locker, locker free, etc.).

3.131
level
(Site MODEL)
An identified storey (ground, first, basement, mezzanine, etc) within an interchange building or SITE on which SITE COMPONENTs reside. A PATH LINK may connect components on different levels.

3.132
lift equipment
(Stair Equipment)
Specialisation of PLACE ACCESS EQUIPMENT for LIFTs (provides lift characteristics like depth, maximum load, etc.).

3.133
line
(Route MODEL)
A group of ROUTEs which is generally known to the public by a similar name or number.

3.134
line network
(Line Network MODEL)
The topological structure of a NETWORK as a graph of LINE SECTIONs. This allows the branches and loops of a LINE to be described as a whole.

3.135
line section
(Line Network MODEL)
A part of a NETWORK comprising an edge between two nodes. Not directional.

3.136
line shape
(Generic Projection MODEL)
The graphical shape of a LINK obtained from a formula or other means, using the LOCATION of its limiting POINTs and depending on the LOCATING SYSTEM used for the graphical representation.

3.137
link
(Generic Point & Link MODEL)
An oriented spatial object of dimension 1 with view to the overall description of a network, describing a connection between two POINTs.

3.138
link in link sequence
(Generic Point & Link Sequence MODEL)
The order of a LINK in a LINK SEQUENCE to which it belongs.

3.139
link projection
(Generic Projection MODEL)
An oriented correspondence from one LINK of a source layer, onto an entity in a target layer: e.g. LINK SEQUENCE, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

3.140
link sequence
(Generic Point & Link Sequence MODEL)
An ordered sequence either of POINTs or of LINKs, defining a path through the network.

3.141
local service
(Generic Equipment MODEL)
A named service relating to the use of the SITE or transport services at a particular location, for example porterage, assistance for disabled users, booking offices etc. The service may have a VALIDITY CONDITION associated with it. A LOCAL SERVICE is treated as a form of non-material EQUIPMENT.

3.142
locating system
(Location MODEL)
The system used as reference for location and graphical representation of the network and other spatial objects.

3.143
location
(Location MODEL)

The position of a POINT with a reference to a given LOCATING SYSTEM (e. g. coordinates).

3.144
logical display
(Passenger Information Equipment MODEL)
A set of data that can be assembled for assignment to a physical PASSENGER INFORMATION EQUIPMENT or to a logical channel such as web or media. It is independent of any physical embodiment.

A LOGICAL DISPLAY may have a set of DISPLAY ASSIGNMENTS each of which associates a JOURNEY PATTERN whose journeys are to be shown at the LOGICAL DISPLAY. It may also be associated with a SCHEDULED STOP POINT. A LOGICAL DISPLAY corresponds to a SIRI STOP MONITORING point.

3.145
lost property service
(Local Service Equipment MODEL)
Specialisation of CUSTOMER SERVICE for lost properties.

3.146
luggage service
(Local Service Equipment MODEL)
Specialisation of CUSTOMER SERVICE for luggage services (provides luggage service facilities and characteristics like luggage trolley, free to use, etc.).

3.147
luggage locker equipment
(Site Equipment MODEL)
Specialisation of STOP PLACE EQUIPMENT for luggage lockers.

3.148
management agent
(Additional Organisation MODEL)
Specialisation of ORGANISATION for MANAGEMENT AGENTs.

3.149
manoeuvre
(Network Restriction MODEL)
A specification of impossible move for a certain type of vehicle. It specifies from which INFRASTRUCTURE LINK to which other (adjacent) INFRASTRUCTURE LINK a certain VEHICLE TYPE cannot proceed, due to physical restrictions.

3.150
manoeuvring requirement
(Vehicle Type MODEL)
A classification of requirements for a public transport VEHICLE according to the manoeuvring capabilities of the vehicle.

3.151
medical need
(Accessibility MODEL)
A specific USER NEED, i.e. a requirement of a passenger as regards medical constraint (e.g. allergy) to access public transport.

3.152
meeting point service
(Local Service Equipment MODEL)
Specialisation of CUSTOMER SERVICE for meeting points (provides characteristics like description, label, etc.).

3.153
meeting restriction
(Network Restriction MODEL)
A pair of INFRASTRUCTURE LINKs where vehicles of specified VEHICLE TYPEs are not allowed to meet.

3.154
mobility need
(Accessibility MODEL)
A specific USER NEED, i.e. a constraint of a passenger as regards his mobility, e.g. wheelchair, assisted wheelchair, etc.

3.155
mode
(Reusable Transport Mode MODEL)
Any means of transport.

3.156
money service
(Local Commercial Service MODEL)
Specialisation of LOCAL SERVICE dedicated to money services.

3.157
navigation path
(Path & Navigation Path MODEL)
A designated path between two places. May include an ordered sequence of PATH LINKs.

3.158
navigation path assignment
(Path Assignment MODEL)
The allocation of a NAVIGATION PATH to a specific STOP POINT ASSIGNMENT, for example to indicate the path to be taken to make a CONNECTION.

3.159
network
(Route MODEL)
A named grouping of LINEs under which a transport network is known.

3.160
normal dated block
(Vehicle Service MODEL)
A DATED BLOCK identical to a long-terms planned BLOCK, possibly updated according to short-term modifications of the PRODUCTION PLAN decided by the control staff.

3.161
normal dated vehicle journey
(Dated Journey MODEL)
A DATED VEHICLE JOURNEY identical to a long-term planned VEHICLE JOURNEY, possibly updated according to short-term modifications of the PRODUCTION PLAN decided by the control staff.

3.162
notice
(Notice MODEL)
A text for informational purposes on exceptions in a LINE, a JOURNEY PATTERN, etc. The information may be usable for passenger or driver information.

3.163
notice assignment
(Notice Assignment MODEL)

The assignment of a NOTICE showing an exception in a JOURNEY PATTERN, a COMMON SECTION, or a VEHICLE JOURNEY, possibly specifying at which POINT IN JOURNEY PATTERN the validity of the NOTICE starts and ends respectively.

3.164
onboard stay
(Facility MODEL)
Permission to board early before the journey or stay on board after the journey.

3.165
operating day
(Service Calendar MODEL)
A day of public transport operation of which the characteristics are defined within a specific SERVICE CALENDAR. An OPERATING DAY may last more than 24 hours.

3.166
operating department
(Transport Organisations MODEL)
A specific DEPARTMENT which administers certain LINEs.

3.167
operating period
(Service Calendar MODEL)
A continuous interval of time between two OPERATING DAYs which will be used to define validities.

3.168
operational context
(Transport Organisations MODEL)
Characterization of a set of operational objects, such as timing or links determined either by a DEPARTMENT or by an ORGANISATIONAL UNIT.

3.169
operator
(Transport Organisations MODEL)
A company providing public transport services.

3.170
organisation
(Generic Organisation MODEL)
A legally incorporated body associated with any aspect of the transport system.

3.171
organisation day type
(Additional Organisation MODEL)
DAY TYPE that is defined in terms of operation or not operation of a referenced SERVICED ORGANISATION.

3.172
organisation part
(Generic Organisation MODEL)
A part of an ORGANISATION to which specific responsibilities upon the data and data management may be assigned.

3.173
organisational unit
(Generic Organisation MODEL)
An ORGANISATION PART to which a set of responsibilities in a public transport company for planning and control is assigned.

3.174
other organisation
(Additional Organisation MODEL)
Generic ORGANISATION being neither an AUTHORITY, neither a public transport OPERATOR (TRAVEL AGENT, MANAGEMENT AGENT, etc.).

3.175
other sign
(Sign Equipment)
Specialisation of SIGN EQUIPMENT which are not HEADING SIGNs neither PLACE SIGNs.

3.176
overtaking possibility
(Network Restriction MODEL)
NETWORK RESTRICTION specifying a POINT or a LINK where vehicles of specified VEHICLE TYPEs are or are not allowed to overtake each other.

3.177
parking
(Parking MODEL)
Designated locations for leaving vehicles such as cars, motorcycles and bicycles.

3.178
parking entrance for vehicles
(Parking MODEL)
An entrance for vehicles to the PARKING from the road.

3.179
parking area
(Parking MODEL)
A marked zone within a PARKING containing PARKING BAYs.

3.180
parking bay
(Parking MODEL)
A place to park an individual vehicle.

3.181
parking capacity
(Parking MODEL)
PARKING properties providing information about its CAPACITY.

3.182
parking component
(Parking MODEL)
Generic COMPONENT of a PARKING (e.g. PARKING AREA or PARKING BAY).

3.183
parking passenger entrance
(Parking MODEL)
An entrance to the PARKING for passengers on foot or other out-of-vehicle mode, such as wheelchair.

3.184
parking point
(Vehicle & Crew Point MODEL)
A TIMING POINT where vehicles may stay unattended for a long time. A vehicle's return to park at a PARKING POINT marks the end of a BLOCK.

3.185
parking properties
(Parking MODEL)
PARKING specific properties other than its CAPACITY.

3.186
passenger accessibility need
(Accessibility MODEL)
A passenger's requirement for accessibility, comprising one or more USER NEEDs. For example, that he is unable to navigate stairs, or lifts, or has visual or auditory impairments. PASSENGER ACCESSIBILITY NEEDS can be used to derive an accessibility constraint for the passenger, allowing the computation of paths for passengers with specifically constrained mobility. Example: Wheelchair, No Lifts, No Stairs.

3.187
passenger carrying requirement
(Vehicle Type MODEL)
A classification of requirements for a public transport vehicle according to the passenger carrying capabilities of the vehicle.

3.188
passenger equipment
(Generic Equipment MODEL)
An item of equipment of a particular type actually available at a location within a PLACE or a VEHICLE

3.189
passenger information equipment
(Passenger Information Equipment MODEL)
A public transport information piece of equipment, as for instance terminals (on street, at information desks, telematic, etc.) or printed material (leaflets displayed at stops, booklets, etc.).

3.190
passenger safety equipment
(Passenger Service Equipment MODEL)
Specialisation of PASSENGER EQUIPMENT for passenger safety.

3.191
passenger stop assignment
(Stop Assignment MODEL)
The allocation of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) to a specific STOP PLACE for a PASSENGER SERVICE, and also possibly a QUAY and BOARDING POSITION.

3.192
passing time
(Passing Times MODEL)
Time data concerning public transport vehicles passing a particular POINT; e.g. arrival time, departure time, waiting time.

3.193
path junction
(Path & Navigation Path MODEL)
A designated point, inside or outside of a STOP PLACE or POINT OF INTEREST, at which two or more PATH LINKs may connect or branch.

3.194
path link
(Path & Navigation Path MODEL)

A link within a PLACE of or between two PLACEs (that is STOP PLACEs, ACCESS SPACEs or QUAYs, BOARDING POSITIONs, POINTs OF INTEREST etc. or PATH JUNCTIONs) that represents a step in a possible route for pedestrians, cyclists or other out-of-vehicle passengers within or between a PLACE.

NOTE 1 to entry: It is possible but not mandatory that a PATH LINK projects onto a more detailed set of infrastructure or mapping links that plot the spatial course, allowing it to be represented on maps and to tracking systems.

3.195
path link end
(Path & Navigation Path MODEL)
Beginning or end SITE for a PATH LINK. May be linked to a specific LEVEL of the SITE.

3.196
path link in sequence
(Path & Navigation Path MODEL)
A step of a NAVIGATION PATH indicating traversal of a particular PATH LINK as part of a recommended route.

NOTE 1 to entry: The same PATH LINK may occur in different sequences in different NAVIGATION PATHs.

3.197
place
(Generic Place MODEL)
A geographic place of any type which may be specified as the origin or destination of a trip. A PLACE may be of dimension 0 (a POINT), 1 (a road section) or 2 (a ZONE).

3.198
place access equipment
(Access Equipment)
Specialisation of PLACE EQUIPMENT dedicated to access (e.g. lifts, entrances, stairs, ramps, etc.).

3.199
place equipment
(Generic Equipment MODEL)
An item of equipment of a particular type actually available at a location within a PLACE.

3.200
place lighting
(Access Equipment)
Specialisation of PLACE EQUIPMENT for LIGHTING EQUIPMENT (e.g. lamp post).

3.201
place sign
(Sign Equipment)
Sign with the name of a STOP PLACE on it.

3.202
point
(Generic Point & Link MODEL)
A 0-dimensional node of the network used for the spatial description of the network. POINTs may be located by a LOCATION in a given LOCATING SYSTEM.

3.203
point in journey pattern
(Journey Pattern MODEL)
A STOP POINT or TIMING POINT in a JOURNEY PATTERN with its order in that JOURNEY PATTERN.

3.204
point in link sequence
(Generic Point & Link Sequence MODEL)

A POINT in a LINK SEQUENCE indicating its order in that particular LINK SEQUENCE.

3.205
point of interest
(Point Of Interest MODEL)
A type of PLACE to or through which passengers may wish to navigate as part of their journey and which is modelled in detail by journey planners.

3.206
point of interest classification
(Point Of Interest MODEL)
A classification of a POINT OF INTEREST that may be used in a CLASSIFICATION HIERARCHY to categorise the point by nature of interest using a systematic taxonomy, for example Museum, Football, Stadium.

3.207
point of interest classification hierarchy
(Point Of Interest MODEL)
A logical hierarchy for organizing POINT OF INTEREST CLASSIFICATIONs. A POINT OF INTEREST CLASSIFICATION can belong to more than one hierarchy.

3.208
point of interest classification membership
(Point Of Interest MODEL)
The POINT OF INTEREST CLASSIFICATION and POINT OF INTEREST CLASSIFICATION MEMBERSHIP are used to encode a hierarchy of classifications to index and find different types of POINT OF INTEREST. For example, Educational Building -> School -> Primary School, or Cultural Attraction -> Museum -> Art Museum.

NOTE 1 to entry: POINT OF INTEREST CLASSIFICATION MEMBERSHIP does not have to be disjoint, i.e. the same category may appear in more than one CLASSIFICATION.

3.209
point of interest component
(Point Of Interest MODEL)
Specialisation of SITE COMPONENT for COMPONENT of POINT OF INTEREST. Usually used for POINT OF INTEREST SPACEs.

3.210
point of interest entrance
(Point Of Interest MODEL)
Specialisation of ENTRANCE to enter/exit a POINT OF INTEREST.

3.211
point of interest space
(Point Of Interest MODEL)
Specialisation of POINT OF INTEREST COMPONENT for SPACEs. A physical area within the POINT OF INTEREST, such as a concourse.

3.212
point of interest vehicle entrance
(Point Of Interest MODEL)
A physical entrance or exit to/from a POINT OF INTEREST for vehicles.

3.213
point on link
(Generic Point & Link MODEL)
A POINT on a LINK which is not needed for LINK definition, but may be used for other purposes, e.g. for purposes of AVM or PI, or for driver information.

3.214
point on route
(Route MODEL)
A ROUTE POINT used to define a ROUTE with its order on that ROUTE.

3.215
point projection
(Generic Projection MODEL)
An oriented correspondence from one POINT of a source layer, onto an entity in a target layer: e.g. POINT, LINK, LINK SEQUENCE, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

3.216
postal address
(Topographic MODEL)
A specification of ADDRESS refining it by using the attributes used for conventional identification for mail. Comprises variously a building Identifier, Street name, Post code and other descriptors.

3.217
property of day
(Service Calendar MODEL)
A property which a day may possess, such as school holiday, weekday, summer, winter etc.

3.218
psychosensory need
(Accessibility MODEL)
A specific USER NEED, i.e. a constraint of a passenger as regards his psycho-sensory impairments, such as visual impairment, auditory impairment, averse to confined spaces, etc.

3.219
purpose of equipment profile
(Vehicle Type MODEL)
A functional purpose which requires a certain set of equipment of different types put together in a VEHICLE EQUIPMENT PROFILE or STOP POINT EQUIPMENT PROFILE.

3.220
purpose of grouping
(Generic Grouping MODEL)
Functional purpose for which GROUPs of elements are defined. The PURPOSE OF GROUPING may be restricted to one or more types of the given object.

3.221
purpose of journey partition
(Coupled Journey MODEL)
An operational purpose changing within a JOURNEY PATTERN and with this subdividing the SERVICE JOURNEY into JOURNEY PARTs.

3.222
quay
(Stop Place MODEL)
A place such as platform, stance, or quayside where passengers have access to PT vehicles, Taxi, cars or other means of transportation. A QUAY may serve one or more VEHICLE STOPPING PLACEs and be associated with one or more STOP POINTS.

A QUAY may contain other sub QUAYs. A child QUAY shall be physically contained within its parent QUAY.

3.223
queuing equipment
(Access Equipment)
Specialisation of PLACE ACCESS EQUIPMENT dedicated to queuing.

3.224
railway element
(Infrastructure Network MODEL)
A type of INFRASTRUCTURE LINK used to describe a railway network.

3.225
railway junction
(Infrastructure Network MODEL)
A type of INFRASTRUCTURE POINT used to describe a railway network.

3.226
ramp equipment
(Access Equipment)
Specialisation of PLACE ACCESS EQUIPMENT for ramps (provides ramp characteristics like length, gradient, etc.).

3.227
refreshments service
(Local Commercial Service MODEL)
Specialisation of LOCAL SERVICE dedicated to refreshments service.

3.228
relief opportunity
(Vehicle Service MODEL)
A time in a BLOCK where a vehicle passes a RELIEF POINT. This opportunity may or may not be actually used for a relief.

3.229
relief point
(Vehicle & Crew Point MODEL)
A TIMING POINT where a relief is possible, i.e. a driver may take on or hand over a vehicle. The vehicle may sometimes be left unattended.

3.230
responsibility role
(Responsibility Role MODEL)
A particular role an ORGANISATION or an ORGANISATION PART is playing as regards certain data, for example data origination, data augmentation, data aggregation, data distribution, planning, operation, control, ownership etc.

3.231
responsibility role assignment
(Responsibility Role MODEL)
The assignment of one or more roles to an ORGANISATION or an ORGANISATION PART as regards the responsibility it will have as regards specific data (e.g. ownership, planning, etc.) and the management of this data (e.g. distribution, updates, etc.).

3.232
responsibility set
(Responsibility Role MODEL)
A list of possible responsibilities over one or more ENTITies IN VERSION, resulting from the process of the assignment of RESPONSIBILITY ROLEs (such as data origination, ownership, etc) on specific data (instances) to ORGANISATIONs or ORGANISATION PARTs.

3.233
retail service
(Local Commercial Service MODEL)
Specialisation of LOCAL SERVICE dedicated to retail services.

3.234
rhythmical journey group
(Vehicle Journey Times MODEL)
A group of VEHICLE JOURNEYS following the same JOURNEY PATTERN having the same rhythm "every hour (for example runs at xxh10, xxh25 and xxh45) between a specified start and end time."

3.235
road address
(Topographic MODEL)
Specialization of ADDRESS refining it by using the characteristics such as road number, and name used for conventional identification of along a road.

3.236
road element
(Infrastructure Network MODEL)
A type of INFRASTRUCTURE LINK used to describe a road network.

3.237
road junction
(Infrastructure Network MODEL)
A type of INFRASTRUCTURE POINT used to describe a road network.

3.238
rough surface
(Access Equipment)
Specialisation of PLACE EQUIPMENT for rough surfaces, giving properties of surface texture, mainly for impaired person information.

3.239
route
(Route MODEL)
An ordered list of located POINTs defining one single path through the road (or rail) network. A ROUTE may pass through the same POINT more than once.

3.240
route link
(Route MODEL)
An oriented link between two ROUTE POINTs allowing the definition of a unique path through the network.

3.241
route point
(Route MODEL)
A POINT used to define the shape of a ROUTE through the network.

3.242
routing constraint zone
(Routing Constraint MODEL)
A ZONE defining a ROUTING CONSTRAINT. The ZONE may be defined by its contained SCHEDULED STOP POINTS or by its boundary points.

Examples of routing constraints are: if you board in this ZONE, you can't alight in the same ZONE.

3.243
rubbish disposal
(Passenger Service Equipment MODEL)
Specialization of EQUIPMENT for rubbish disposal, describing rubbish types, etc.

3.244
sanitary equipment
(Passenger Service Equipment MODEL)
Specialisation of PASSENGER EQUIPMENT for sanitary facilities.

3.245
scheduled stop point
(Service Pattern MODEL)
A POINT where passengers can board or alight from vehicles.

3.246
schematic map
(Schematic Map MODEL)
A map representing schematically the layout of the topographic structure of the public transport network (a set of LINEs) and a set of SITEs. It can include a pixel projection of a set of ENTITies onto a bitmap image so as to support hyperlinked interactions.

3.247
seating equipment
(Site Equipment MODEL)
Specialisation of PLACE EQUIPMENT describing the properties of seating

3.248
service calendar
(Service Calendar MODEL)
A collection of DAY TYPE ASSIGNMENTs.

3.249
service exclusion
(Routing Constraint MODEL)
A constraint expressing the fact that the service, on a specific JOURNEY PATTERN (usually a FTS JOURNEY PATTERN) cannot operate when another (regular) service operates. This may occur only on a subpart of the JOURNEY PATTERN, or only on one or some specific SCHEDULED STOP POINTS.

3.250
service facility set
(Facility MODEL)
Set of FACILITies available for a SERVICE JOURNEY or a JOURNEY PART. The set may be available only for a specific VEHICLE TYPE within the SERVICE (e.g. carriage equipped with low floor).

3.251
service journey
(Service Journey MODEL)
A passenger carrying VEHICLE JOURNEY for one specified DAY TYPE. The pattern of working is in principle defined by a SERVICE JOURNEY PATTERN.

3.252
service journey interchange
(Interchange MODEL)
The scheduled possibility for transfer of passengers between two SERVICE JOURNEYs at the same or different STOP POINTs.

3.253
service journey pattern
(Service Pattern MODEL)
The JOURNEY PATTERN for a (passenger carrying) SERVICE JOURNEY.

3.254
service journey pattern interchange
(Interchange MODEL)
A recognised/organised possibility for passengers to change public transport vehicles using two STOP POINTs (which may be identical) on two particular SERVICE JOURNEY PATTERNs, including the maximum wait duration allowed and the standard to be aimed at. These may supersede the times given for the DEFAULT INTERCHANGE. Schedulers may use this entity for synchronisation of journeys.

3.255
service link
(Service Pattern MODEL)
A LINK between an ordered pair of SCHEDULED STOP POINTs.

3.256
service pattern
(Service Pattern MODEL)
The subset of a JOURNEY PATTERN made up only of STOP POINTs IN JOURNEY PATTERN.

3.257
service restriction
(Fare-Related Service Restriction MODEL)
Parameters describing the limitations as regards the use of equipment or service.

3.258
service site
(Site MODEL)
A sub-type of SITE which is of specific interest for the operator (e.g. where a joint service or a joint fee is proposed), other than a STOP PLACE.

3.259
serviced organisation
(Additional Organisation MODEL)
A public or private organisation for which public transport services are provided on specific days, e.g. a school, university or works.

3.260
shelter equipment
(Site Equipment MODEL)
Specialisation of WAITING EQUIPMENT for a shelter.

3.261
sign equipment
(Sign Equipment)
Specialisation of PLACE EQUIPMENT for signs (heading signs, etc.).

3.262
simple feature
(Generic Zone and Feature MODEL)
An abstract representation of elementary objects related to the spatial representation of the network. POINTs (0-dimensional objects), LINKs (1-dimensional objects) and ZONEs (2-dimensional objects) may be viewed as SIMPLE FEATUREs.

3.263
site
(Site MODEL)
A type of PLACE, such as a STOP PLACE, POINT OF INTEREST or ADDRESS, to which passengers may wish to travel. A SITE can have designated ENTRANCEs that represent the available points of access for different USER NEEDs.

3.264
site component
(Site MODEL)
An element of a SITE describing a part of its structure. SITE COMPONENTs share common properties for data management, accessibility and other features.

3.265
site connection
(Service Connection MODEL)
The physical (spatial) possibility for a passenger to change from one public transport vehicle to another to continue the trip, determined by physical locations, such as SITEs and/or its components and/or ENTRANCEs, in particular STOP PLACEs and/or its components. Different times may be necessary to cover the resulting distance, depending on the kind of passenger.

3.266
site connection end
(Service Connection MODEL)
One end of a SITE CONNECTION.

3.267
site element
(Site MODEL)
A type of PLACE specifying common properties of a SITE or a SITE COMPONENT to describe it, including accessibility.

3.268
site equipment
(Site Equipment MODEL)
Specialisation of PLACE EQUIPMENT for SITEs (e.g. LUGGAGE LOCKER, WAITING EQUIPMENT, TROLLEY STAND, etc.).

3.269
site facility set
(Facility MODEL)
Set of FACILITies available for a SITE or SITE ELEMENT.

3.270
site point in sequence
(Path & Navigation Path MODEL)
POINT building up a NAVIGATION PATH within an ordered set of other SITE POINTs IN SEQUENCE. May also be the beginning or end of a PATH LINK IN SEQUENCE.

3.271
special service
(Service Journey MODEL)
A work of a vehicle that is not planned in a classical way, i.e. that is generally not based on VEHICLE JOURNEYs using JOURNEY PATTERNs. It involves specific characteristics (such as specific access rights) and/or may be operated under specific circumstances.

3.272
stair equipment
(Stair Equipment)
Specialisation of PLACE ACCESS EQUIPMENT for stairs (stair, escalator, staircase, etc.).

3.273
staircase equipment
(Stair Equipment)
Specialisation of STAIR EQUIPMENT for stair cases.

3.274
stop area
(Service Pattern MODEL)
A group of SCHEDULED STOP POINTs close to each other.

3.275
stop assignment
(Stop Assignment MODEL)
The allocation of a SCHEDULED STOP POINT (i.e. a SCHEDULED STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) to a specific STOP PLACE, for either a Passenger JOURNEY or VEHICLE SERVICE

3.276
stop place
(Stop Place MODEL)
A place comprising one or more locations where vehicles may stop and where passengers may board or leave vehicles or prepare their trip. A STOP PLACE will usually have one or more well-known names.

3.277
stop place component
(Stop Place MODEL)
An element of a STOP PLACE describing part of its structure. STOP PLACE COMPONENTs share common properties for data management, accessibility and other features.

3.278
stop place entrance
(Stop Place MODEL)
A physical entrance or exit to/from a STOP PLACE for a Passenger. May be a door, barrier, gate or other recognizable point of access.

3.279
stop place space
(Stop Place MODEL)
A physical area within a STOP PLACE, for example, a QUAY, BOARDING POSITION, ACCESS SPACE or EQUIPMENT PLACE.

3.280
stop place vehicle entrance
(Stop Place MODEL)
A physical entrance or exit to/from a STOP PLACE for a vehicle.

3.281
stop point in journey pattern
(Service Pattern MODEL)
A POINT in a JOURNEY PATTERN which is a SCHEDULED STOP POINT.

3.282
submode
(Reusable Transport Mode MODEL)
A variant of a MODE, as for instance international or domestic rail (rail being the MODE).

3.283
suitability
(Accessibility MODEL)
A statement of whether a particular USER NEED can be met. It can be used to state whether a SITE can be accessed by a passenger with a particular USER NEED.

3.284
target passing time
(Passing Times MODEL)

Time data about when a public transport vehicle should pass a particular POINT IN JOURNEY PATTERN on a particular DATED VEHICLE JOURNEY, in order to match the latest valid plan.

3.285
tariff zone
(Generic Zone and Feature MODEL)
A ZONE used to define a zonal fare structure in a zone-counting or zone-matrix system.

3.286
template service journey
(Service Journey MODEL)
A passenger carrying TEMPLATE SERVICE JOURNEY. As TEMPLATE SERVICE JOURNEY, it may represent multiple journeys.

3.287
template vehicle journey
(Vehicle Journey MODEL)
A repeating VEHICLE JOURNEY for which a frequency has been specified, either as a HEADWAY JOURNEY GROUP (e.g. every 20 minutes) or a RHYTHMICAL JOURNEY GROUP (e.g. at 15, 27 and 40 minutes past the hour). It may thus represent multiple journeys.

3.288
ticket scope
(Fare-Related Service Restriction MODEL)
Scope of ticket.

3.289
ticket validator equipment
(Ticketing Equipment MODEL)
Specialisation of PASSENGER EQUIPMENT (PLACE EQUIPMENT) describing ticket validators.

3.290
ticketing equipment
(Ticketing Equipment MODEL)
Specialization of PASSENGER EQUIPMENT for ticketing.

3.291
ticketing service
(Local Service Equipment MODEL)
Specialization of LOCAL SERVICE for ticketing, providing ticket counter and online purchase information, also associated with payment method and TYPE OF TICKET.

3.292
time band
(Service Calendar MODEL)
A period in a day, significant for some aspect of public transport, e.g. similar traffic conditions or fare category.

3.293
time demand type
(Time Demand Type MODEL)
An indicator of traffic conditions or other factors which may affect vehicle run or wait times. It may be entered directly by the scheduler or defined by the use of TIME BANDs.

3.294
time demand type assignment
(Time Demand Type MODEL)
The assignment of a TIME DEMAND TYPE to a TIME BAND depending on the DAY TYPE and GROUP OF TIMING LINKS.

3.295
timetabled passing time
(Passing Times MODEL)
Long-term planned time data concerning public transport vehicles passing a particular POINT IN JOURNEY PATTERN on a specified VEHICLE JOURNEY for a certain DAY TYPE.

3.296
timing link
(Timing Pattern MODEL)
An ordered pair of TIMING POINTs for which run times may be recorded.

3.297
timing link in journey pattern
(Journey Pattern MODEL)
The position of a TIMING LINK in a JOURNEY PATTERN. This entity is needed if a TIMING LINK is repeated in the same JOURNEY PATTERN and separate information is to be stored about each iteration of the TIMING LINK.

3.298
timing pattern
(Timing Pattern MODEL)
The subset of a JOURNEY PATTERN made up only of TIMING POINTs IN JOURNEY PATTERN.

3.299
timing point
(Timing Pattern MODEL)
A POINT against which the timing information necessary to build schedules may be recorded.

3.300
timing point in journey pattern
(Journey Pattern MODEL)
A POINT in a JOURNEY PATTERN which is a TIMING POINT.

3.301
topographic place
(Topographic MODEL)
A geographical settlement which provides topographical context when searching for or presenting travel information, for example as the origin or destination of a trip. It may be of any size (e.g. County, City, Town, Village) and of different specificity e.g. Greater London, London, West End, Westminster, St James.

A TOPOGRAPHICAL PLACE shall always have a canonical gazetteer name. It may be necessary to use the hierarchical topographical relationships of the TOPOGRAPHICAL PLACE to establish a unique context with which to distinguish between two TOPOGRAPHICAL PLACEs with the same name.

3.302
traffic control point
(Activation MODEL)
A POINT where the traffic flow can be influenced. Examples are: traffic lights (lanterns), barriers.

3.303
train
(Train MODEL)
A VEHICLE TYPE composed of TRAIN ELEMENTs in a certain order, i.e. of wagons assembled together and propelled by a locomotive or one of the wagons.

3.304
train block
(Vehicle Service MODEL)

A composite train formed of several BLOCKs coupled together during a certain period. Any coupling or separation action marks the start of a new TRAIN BLOCK.

3.305
train block part
(Vehicle Service MODEL)
The position of a vehicle BLOCK within a TRAIN BLOCK.

3.306
train component
(Train MODEL)
A specification of the order of TRAIN ELEMENTs in a TRAIN.

3.307
train element
(Train MODEL)
An elementary component of a TRAIN (e.g. wagon, locomotive).

3.308
train in compound train
(Train MODEL)
The specification of the order of TRAINs in a COMPOUND TRAIN.

3.309
train number
(Vehicle Journey MODEL)
Specification of codes assigned to particular VEHICLE JOURNEYs when operated by TRAINs or COMPOUND TRAINs according to a functional purpose (passenger information, operation follow-up, etc).

3.310
train stop assignment
(Train Stop Assignment MODEL)
The association of a TRAIN COMPONENT at a SCHEDULED STOP POINT with a specific STOP PLACE and also possibly a QUAY and BOARDING POSITION.

3.311
transfer
(Generic Place MODEL)
A couple of POINTs located sufficiently near that it may represent for a passenger a possibility to reach one of these POINTs when starting at the other one in a timescale which is realistic when carrying out a trip, e.g. CONNECTION, ACCESS.

3.312
transfer end
(Generic Place MODEL)
End point of a TRANSFER.

3.313
transfer restriction
(Routing Constraint MODEL)
A CONSTRAINT that can be applied on a CONNECTION or INTERCHANGE between two SCHEDULED STOP POINT, preventing or forbidding the passenger to use it.

3.314
travel agent
(Additional Organisation MODEL)
Specialisation of ORGANISATION for TRAVEL AGENT

3.315
travelator equipment
(Stair Equipment)
Specialisation of PLACE EQUIPMENT for travelators (provides travelator properties like speed, etc.).

3.316
trolley stand equipment
(Site Equipment MODEL)
Specialisation of STOP PLACE EQUIPMENT for trolley stands.

3.317
turnaround time limit
(Journey Pattern Times MODEL)
The maximum time for which a vehicle may be scheduled to wait at a particular TIMING POINT (often included in a TURN STATION) without being returned to a PARKING POINT. A minimum time for a vehicle to turn its direction may also be recorded. This may be superseded by a DEAD RUN.

3.318
type of access feature
(Check Constraint MODEL)
A Classification of ACCESS FEATURE for CHECK CONSTRAINT (e.g. barrier, narrow entrance, confined space, queue management, etc.).

3.319
type of accessibility limitation
(Accessibility MODEL)
A classification for ACCESSIBILITY LIMITATIONs, e.g. audio, visual, step free, etc.

3.320
type of accessibility tools
(Local Service Equipment MODEL)
A classification of ACCESSIBILITY TOOLS used by or available from ASSISTANCE SERVICE (e.g. wheelchair, walking stick, audio navigator, visual navigator, etc.).

3.321
type of activation
(Activation MODEL)
A classification of real-time processes that are activated when vehicles passes an ACTIVATION POINT or an ACTIVATION LINK.

3.322
type of assistance service
(Local Service Equipment MODEL)
A classification of ASSISTANCE SERVICE (e.g. boarding assistance, onboard assistance, porterage, foreign language, sign language translation, etc.).

3.323
type of boarding position
(Stop Place MODEL)
A classification for BOARDING POSITIONs.

3.324
type of check constraint
(Check Constraint MODEL)
A classification of CHECK CONSTRAINT (e.g. ticket collection, ticket purchase, baggage check-in, incoming customs, outgoing customs, tax refunds, etc.).

3.325
type of communication service
(Local Commercial Service MODEL)
A classification of COMMUNICATION SERVICE (e.g. free wifi, public wifi, phone, mobile coverage, internet, video entertainment, audio entertainment, post box, post office, business services).

3.326
type of congestion
(Check Constraint MODEL)
A typology of congestions resulting from CHECK CONSTRAINT (e.g. no waiting, queue, crowding, full).

3.327
type of coupling
(Coupled Journey MODEL)
A classification for COUPLING of BLOCK PARTs.

3.328
type of cycle parking equipment
(Parking Equipment MODEL)
A classification of CYCLE PARKING (e.g. racks, bars, railings, etc.).

3.329
type of delivery variant
(Notice MODEL)
A classification of a DELIVERY VARIANT. The way of delivering a NOTICE: by vocal announcement, by visual display, issuing printed material.

3.330
type of direction of use
(Check Constraint MODEL)
Direction in which EQUIPMENT can be used. (e.g. up, down, level, one way, both way, etc.).

3.331
type of emergency service
(Local Service Equipment MODEL)
A typology of emergency services (e.g police, first aid, SOS point, cctv).

3.332
type of entity
(Generic Entity MODEL)
Classification of ENTITies, for instance according to the domain in which they are defined or used.

3.333
type of equipment
(Generic Equipment MODEL)
A classification of equipment items to be installed at stop points or onboard vehicles, for instance.

3.334
type of fare class
(Fare-Related Service Restriction MODEL)
A classification for FARE CLASSes.

3.335
type of flexible service
(Flexible Service MODEL)
A typology of flexible services:

- Virtual line service.
- Flexible service with main route.

- Corridor service.
- Fixed stop area-wide flexible service.
- Free area-wide flexible service.
- Mixed types of flexible service (not at POINT level).

The type of flexibility can be defined at JOURNEY PATTERN level or at POINT IN JOURNEY PATTERN level in case of mixed types of flexible service inside the same JOURNEY PATTERN.

3.336
type of frame
(Generic Version MODEL)
A classification of VERSION FRAMEs according to a common purpose. E.g. line descriptions for line versions, vehicle schedules, operating costs. A TYPE OF FRAME is ruled by a unique TYPE OF VALIDITY.

3.337
type of gender limitation
(Passenger Service Equipment MODEL)
A classification for GENDER LIMITATIONSs (mainly for SANITARY EQUIPMENT, e.g. male only, female only, both).

3.338
type of handrail
(Stair Equipment)
A classification of HANDRAIL (one side, both sides).

3.339
type of hire service
(Local Commercial Service MODEL)
A classification of HIRE SERVICEs (e.g. car hire, motor cycle hire, cycle hire, recreational device hire).

3.340
type of journey pattern
(Journey Pattern MODEL)
A classification of JOURNEY PATTERNs used to distinguish other categories of JOURNEY PATTERN than SERVICE JOURNEY PATTERN and DEAD RUN PATTERN.

3.341
type of line
(Route MODEL)
A classification for LINEs.

3.342
type of link
(Generic Point & Link MODEL)
A classification of LINKs to express the different functional roles of a LINK.

3.343
type of link sequence
(Generic Point & Link Sequence MODEL)
A classification of LINK SEQUENCEs used to define the different functions a LINK SEQUENCE may be used for. E.g ROUTE, JOURNEY PATTERN, road, TRIP PATTERN, border line, etc.

3.344
type of local service
(Local Service Equipment MODEL)
A generic (abstract) classification of LOCAL SERVICEs.

3.345
type of luggage locker
(Site Equipment MODEL)
A classification for LUGGAGE LOCKER EQUIPMENT (e.g. left luggage, lockers, bike carriage, porterage, free trolleys, paid trolleys).

3.346
type of money service
(Local Commercial Service MODEL)
A classification of MONEY SERVICE (e.g. cash machine, bank, insurance, bureau de change).

3.347
type of notice
(Notice MODEL)
A classification for a NOTICE.

3.348
type of operation
(Generic Organisation MODEL)
A classification of OPERATIONs to express the different functional roles of a DEPARTMENT.

3.349
type of organisation
(Generic Organisation MODEL)
A classification for the ORGANISATIONs according to their activity, e.g. a public transport company, an IT company, etc).

3.350
type of passage
(Stop Place MODEL)
A classification for spaces to express how the space can be used as a passage (e.g. pathway, corridor, overpass, underpass, tunnel, etc.).

3.351
type of passenger information equipment
(Passenger Information Equipment MODEL)
A classification for PASSENGER INFORMATION EQUIPMENT (e.g. next stop indicator, stop announcements, passenger information facility).

3.352
type of payment method
(Fare-Related Service Restriction MODEL)
A classification for payment method (e.g. cash, credit card, debit card, travel card, contactless travel card, mobile phone, token, etc.).

3.353
type of place
(Generic Place MODEL)
A classification for PLACEs.

3.354
type of point
(Generic Point & Link MODEL)
A classification of POINTs according to their functional purpose.

3.355
type of point of interest space
(Point Of Interest MODEL)
A classification for POINT OF INTEREST SPACEs.

3.356
type of product category
(Vehicle Journey MODEL)
A classification for VEHICLE JOURNEYs to express some common properties of journeys for marketing and fare products.

3.357
type of projection
(Generic Projection MODEL)
A classification of the projections according to their functional purpose, the source and target layers.

3.358
type of refreshment service
(Local Commercial Service MODEL)
A classification of REFRESHMENT SERVICE (e.g. beverage vending machine, buffet, food vending machine, restaurant, snacks, trolley service, no beverages available, no food available).

3.359
type of relation to vehicle
(Vehicle Stopping MODEL)
A classification of the way a VEHICLE STOPPING POSITION is used (e.g. front left, front right, back left, back right, driver left, driver right).

3.360
type of responsibility role
(Responsibility Role MODEL)
A classification of RESPONSIBILITY ROLEs, e.g. data ownership.

3.361
type of retail service
(Local Commercial Service MODEL)
A classification of RETAIL SERVICE (e.g. food, newspaper tobacco, health hygiene beauty, fashion accessories, bank finance insurance, tourism, photo booth).

3.362
type of sanitary facility
(Passenger Service Equipment MODEL)
A classification for SANITARY EQUIPMENT (e.g. toilet, wheelchair access toilet, shower, baby change, wheelchair baby change).

3.363
type of seating equipment
(Site Equipment MODEL)
A classification for SEATING EQUIPMENT.

3.364
type of service
(Vehicle Journey MODEL)
A classification for VEHICLE JOURNEYs and SPECIAL SERVICEs to express some common properties of journeys to be taken into account in the scheduling and/or operations control process.

3.365
type of service nature
(Check Constraint MODEL)
A classification for service available for a CHECK CONSTRAINT (e.g. self-service machine, counter service).

3.366
type of shelter
(Site Equipment MODEL)

A classification for SHELTERs.

3.367
type of staffing
(Local Service Equipment MODEL)
A classification for the availability of the STAFF associated with an ASSISTANCE SERVICE (e.g. full time, part time).

3.368
type of stop place
(Stop Place MODEL)
A classification for STOP PLACEs (e.g. complex, simple, multimodal, etc).

3.369
type of stop point
(Service Pattern MODEL)
A classification of SCHEDULED STOP POINTs, used for instance to characterize the equipment to be installed at stops (post, shelter, seats, etc.).

3.370
type of suitability
(Accessibility MODEL)
A classification for SUITABILITY, i.e. assessments as regards a possible SUITABILITY of access according to USER NEEDS.

3.371
type of surface
(Access Equipment)
A classification for ROUGH SURFACE types.

3.372
type of ticket
(Fare-Related Service Restriction MODEL)
A classification for tickets available at a TICKETING EQUIPMENT (e.g. standard, concession, promotion, group, season, travel card, etc.).

3.373
type of ticketing
(Fare-Related Service Restriction MODEL)
A classification for ticketing available at a TICKETING EQUIPMENT (e.g. purchase, collection, card top up, reservations).

3.374
type of traffic control point
(Activation MODEL)
A classification of TRAFFIC CONTROL POINTs.

3.375
type of train element
(Train MODEL)
A classification of TRAIN ELEMENTs.

3.376
type of transfer
(Generic Place MODEL)
A classification for TRANSFER.

3.377
type of user need
(Accessibility MODEL)
A classification of USER NEEDS.

3.378
type of validity
(Generic Version MODEL)
A classification of the validity of TYPEs OF FRAME. E.g. frames for schedules designed for DAY TYPEs, for specific OPERATING DAYs.

3.379
type of version
(Generic Version MODEL)
A classification of VERSIONs. E.g shareability of ENTITies between several versions.

3.380
type of waiting room
(Site Equipment MODEL)
A classification for WAITING ROOM EQUIPMENT.

3.381
type of zone
(Generic Zone and Feature MODEL)
A classification of ZONEs. E.g. TARIFF ZONE, ADMINISTRATIVE ZONE.

3.382
user need
(Accessibility MODEL)
A user's need for a particular SUITABILITY.

3.383
validity condition
(Generic Validity MODEL)
Condition used in order to characterise a given VERSION of a VERSION FRAME. A VALIDITY CONDITION consists of a parameter (e.g. date, triggering event, etc.) and its type of application (e.g. for, from, until, etc.).

3.384
validity rule parameter
(Generic Validity MODEL)
A user defined VALIDITY CONDITION used by a rule for selecting versions. E.g. river level > 1,5 m and bad weather.

3.385
validity trigger
(Generic Validity MODEL)
External event defining a VALIDITY CONDITION. E.g exceptional flow of a river, bad weather, road closure for works.

3.386
vehicle
(Vehicle Type MODEL)
A public transport vehicle used for carrying passengers.

3.387
vehicle access equipment
(Vehicle Passenger Equipment MODEL)
Specialisation of VEHICLE EQUIPMENT dedicated to access vehicles providing information such as low floor, ramp, access area dimensions, etc.

3.388
vehicle entrance
(Site MODEL)
A physical entrance or exit to/from a STOP PLACE for a VEHICLE. May be a door, barrier, gate or other recognizable point of access.

3.389
vehicle equipment profile
(Vehicle Type MODEL)
Each instantiation of this entity gives the number of items of one TYPE OF EQUIPMENT a VEHICLE MODEL should contain for a given PURPOSE OF EQUIPMENT PROFILE. The set of instantiations for one VEHICLE MODEL and one purpose gives one complete 'profile'.

3.390
vehicle journey
(Vehicle Journey MODEL)
The planned movement of a public transport vehicle on a DAY TYPE from the start point to the end point of a JOURNEY PATTERN on a specified ROUTE.

3.391
vehicle journey headway
(Vehicle Journey Times MODEL)
Headway interval information that is available for a VEHICLE JOURNEY (to be understood as the delay between the previous and the next VEHICLE JOURNEY). This information shall be consistent with HEADWAY JOURNEY GROUP if available (HEADWAY JOURNEY GROUP being a more detailed way of describing headway services).

3.392
vehicle journey layover
(Vehicle Journey Times MODEL)
A time allowance at the end of a specified VEHICLE JOURNEY. This time supersedes any global layover or JOURNEY PATTERN LAYOVER.

3.393
vehicle journey run time
(Vehicle Journey Times MODEL)
The time taken to traverse a specified TIMING LINK IN JOURNEY PATTERN on a specified VEHICLE JOURNEY. This gives the most detailed control over times and overrides the DEFAULT SERVICE JOURNEY RUN TIME and JOURNEY PATTERN RUN TIME and the DEFAULT DEAD RUN RUN TIME.

3.394
vehicle journey wait time
(Vehicle Journey Times MODEL)
The time for a vehicle to wait at a particular TIMING POINT IN JOURNEY PATTERN on a specified VEHICLE JOURNEY. This wait time will override the JOURNEY PATTERN WAIT TIME.

3.395
vehicle mode
(Reusable Transport Mode MODEL)
A characterisation of the public transport operation according to the means of transport (bus, tram, metro, train, ferry, ship).

3.396
vehicle model
(Vehicle Type MODEL)
A classification of public transport vehicles of the same VEHICLE TYPE, e.g. according to equipment specifications or model generation.

3.397
vehicle position alignment
(Vehicle Stopping MODEL)
The alignment of a particular BOARDING POSITION with the entrance of a VEHICLE as the result of positioning the VEHICLE at a particular VEHICLE STOPPING PLACE.

3.398
vehicle quay alignment
(Vehicle Stopping MODEL)
The alignment of a particular QUAY with a vehicle as the result of positioning a VEHICLE at a particular VEHICLE STOPPING PLACE.

3.399
vehicle service
(Vehicle Service MODEL)
A workplan for a vehicle for a whole day, planned for a specific DAY TYPE. A VEHICLE SERVICE includes one or several VEHICLE SERVICE PARTs.

3.400
vehicle service part
(Vehicle Service MODEL)
A part of a VEHICLE SERVICE composed of one or more BLOCKs and limited by periods spent at the GARAGE managing the vehicle in question.

3.401
vehicle stopping place
(Vehicle Stopping MODEL)
A place on the vehicle track where vehicles stop in order for passengers to board or alight from a vehicle.

A vehicle track is located on the respective INFRASTUCTURE LINK for the MODE (RAILWAY ELEMENT of rail network, ROAD ELEMENT of road network, etc). A VEHICLE STOPPING PLACE may be served by one or more QUAYs.

3.402
vehicle stopping position
(Vehicle Stopping MODEL)
The stopping position of a vehicle or one of its components at a location: may be specified as a ZONE corresponding to the bounding polygon of the vehicle, or one or more POINTs corresponding to parts of the vehicle such as a door.

If given as a single point, indicates the position for the door relative to an indicated side of the vehicle.

3.403
vehicle type
(Vehicle Type MODEL)
A classification of public transport vehicles according to the vehicle scheduling requirements in mode and capacity (e.g. standard bus, double-deck, etc.).

3.404
vehicle type at point
(Network Restriction MODEL)
The number of vehicles of a specified VEHICLE TYPE which may wait at a specified POINT at any one time. If the capacity is 0, then that type of vehicle may not stop there.

3.405
vehicle type preference
(Journey Pattern Times MODEL)

The preference for the use of a particular VEHICLE TYPE for a SERVICE JOURNEY PATTERN, depending on the DAY TYPE and TIME DEMAND TYPE. The rank of preferences shall be recorded. Different VEHICLE TYPEs may be given the same rank.

3.406
version
(Generic Version MODEL)
A group of operational data instances which share the same VALIDITY CONDITIONs. A version belongs to a unique VERSION FRAME and is characterised by a unique TYPE OF VERSION. E.g. NETWORK VERSION for Line 12 starting from 2000-01-01.

3.407
version frame
(Generic Version MODEL)
A set of VERSIONS referring to the same DATA SOURCE and belonging to the same TYPE OF FRAME. A FRAME may be restricted by VALIDITY CONDITIONs.

3.408
via
(Route MODEL)
A location (e.g. a ROUTE POINT) used to distinguish a ROUTE from another ROUTE. It may be used for DESTINATION DISPLAYs

3.409
waiting equipment
(Site Equipment MODEL)
Specialisation of STOP PLACE EQUIPMENT for WAITING EQUIPMENTs (shelter, waiting room, etc.).

3.410
waiting room equipment
(Site Equipment MODEL)
Specialisation of WAITING EQUIPMENT for waiting rooms, classified by TYPE OF WAITING ROOM.

3.411
wheelchair vehicle equipment
(Vehicle Passenger Equipment MODEL)
Specialisation of VEHICLE EQUIPMENT for wheel chair accessibility on board a VEHICLE providing information such as the number of wheel chair areas and the access dimensions.

3.412
wire element
(Infrastructure Network MODEL)
A type of INFRASTRUCTURE LINK used to describe a wire network.

3.413
wire junction
(Infrastructure Network MODEL)
A type of INFRASTRUCTURE POINT used to describe a wire network.

3.414
zone
(Generic Zone and Feature MODEL)
A two-dimensional PLACE within the service area of a public transport operator (administrative zone, TARIFF ZONE, ACCESS ZONE, etc.).

3.415
zone projection
(Generic Projection MODEL)

An oriented correspondence: from one ZONE in a source layer onto a target entity : e.g. POINT, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

4 Symbols and abbreviations

API	Application Programming Interface.
AVL	Automated Vehicle Location.
AVLS	AVL System.
AVMS	Automated Vehicle Management System.
DDL	Data Definition Language.
DRT	Demand Responsive Transport.
FTS	Flexible Transport Service.
GPS	Global Positioning System.
HTTP	HyperText Transfer Protocol.
IANA	Internet Assigned Numbers Authority.
IETF	Internet Engineering Task Force.
ISO	International Standards Organisation.
NeTEx	Network and Timetable Exchange.
PT	Public Transport.
PTO	Public Transport Operator.
RPC	Remote Procedure Call.
SIRI	Service Interface for Real-time Information.
SOAP	Simple Object Access Protocol.
UML	Unified Modelling Language.
URI	Uniform Resource Identifier.
URL	Universal Resource Locator.
UTMC	Urban Traffic Management Control.
VDV	Verband Deutscher Verkehrsunternehmen (D).
W3C	World Wide Web Consortium.
WGS	World Geodetic Standard.
WS	Web Service.
WSDL	Web Services Description Language.
XHTML	Extensible Hyper Text Mark-up Language.
XMI	XML Metadata Interchange.
XML	eXtensible Mark-up Language.
XSD	XML Schema Document.
XSLT	XSL Transformations Language.

5 Use Cases for Network Topology Exchange

5.1 Purpose

This documented is intended to outline the use cases for the NeTEx interface. Defining the use cases the scope of the standardisation work should become more precise.

This document collects use cases for all parts of NeTEx. Use cases contained in this document are valid for NeTEX part 2 and 3 as well and are not be redefined for these parts.

5.2 Actors and Use Case Types

5.2.1 Actors

The following table gives an overview of information technology systems that are likely to use the NeTEx interface. The columns Producer and Consumer indicate whether the systems will provide or receive the information content. In the last column examples for organisations are given that might operate such systems. The list in this table is not complete and may be extended.

Table 1 — NeTEx actors

Systems	Producer	Consumer	Organisations
Timetable planning systems	x	x	Local authorities
AVL systems		x	Transport operators
Journey planning systems		x	Local authorities
Ticketing systems		x	Local authorities, Transport operators
UTMC		x	Local authorities/operators
DRT operating systems	x	x	DRT operators
Passenger information systems (stationary, on-board, mobile, web-based etc.)	x	x	Transport operators, local authorities
Traffic flow simulation systems		x	Local authorities
Traffic flow prediction systems		x	Local authorities
Service tendering and registration systems	x	x	Local authorities
Mapping systems		x	Commercial and non-commercial services providers like Google Maps, Yellow Pages, etc.
Strategic planning systems	x	x	Long-term transport development planning, infrastructure and urban development authorities
Performance monitoring systems		x	Local authorities
General third-party systems		x	Various

5.2.2 Delivery Use Cases

Delivery use cases are intended to illustrate how the information is delivered from the data supplier to the data consumer. They should clarify delivery aspects stated in questions like: what amount of data is to be delivered? How many data records are to be delivered? What level of detail is needed? How can information be selected/filtered that is relevant to the data consumer? Who is the initiator of the communication? How can the data consumer get updates?

NeTEx is supposed to support the following delivery modes:

DELIVERY-001: Full delivery

> Using a full delivery request the requestor will get a complete and unfiltered set of data.

DELIVERY-002: Incremental updates of changes

> Using an update request the requestor will receive all data records that were updated with respect to the data version number that the requestor currently holds. This means, in particular, that data records that were deleted on the data provider side shall be marked as deleted in the delivery for the data receiver as well.

DELIVERY-003: Filtered delivery

> The requestor can specify filter criteria so that the data provider selects only a subset of the complete data set. The filter criteria may define object contents (e.g. specific lines, modes or operators), the level of delivery detail (e.g. all object details are requested or only summary information) or the number of records to be returned (e.g. "return all stops in town x, but no more than 100 data records").
>
> Incremental updates are a specific form of filtered delivery, it means filtering by data version number.

DELIVERY-004: Synchronous single request/response communication model.

> The requestor can only send a single request and obtain a synchronous response to it. The data provider does not record what he sent earlier to a specific requestor. This means the data exchange is stateless. There is no subscription mechanism in place that would allow for transmitting only the differences with respect to the last delivery.

DELIVERY-05: Periodic Delivery, Scheduled delivery at regular intervals.

> A variation to the request/response model is the Periodic Delivery communication model. In this model the data provider simply delivers content without any request sent by the data receiver. The periodic delivery may happen at regular predefined intervals. The data receiver may or may not confirm the receipt of the data.
>
> The definition of communication partners that are to receive data and details how often data is transmitted via Periodic Delivery is not a task of NeTEx.

NOTE none of the delivery Use Cases explicitly allow asynchronous event delivery (only a request/response model). Asynchronous delivery was not maintained as this type of communication is typically dedicated to real-time exchange, therefore being mainly manage by SIRI. However, as NeTEx and Siri share the same communicatin layer, asynchronous communication available for SIRI is also available for NeTEx.

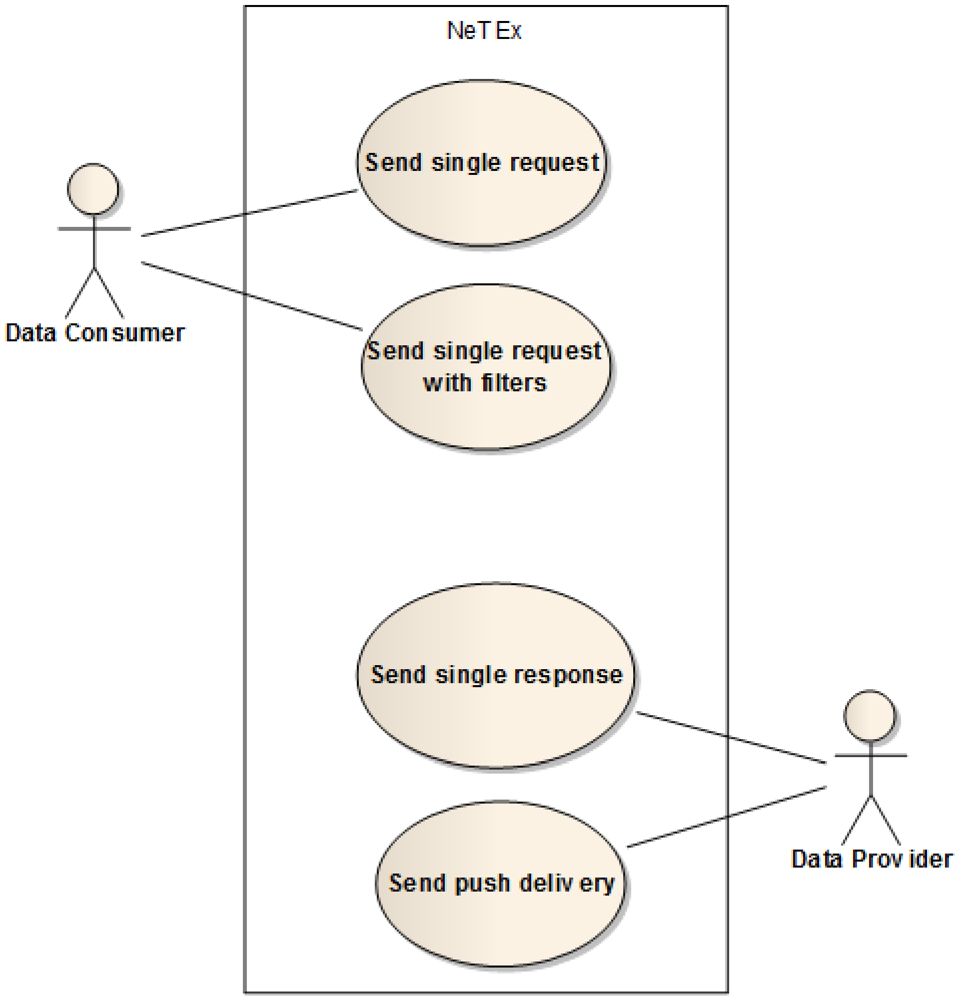


Figure 1 — NeTEx requests

5.2.3 Content Use Cases

Structure follows Transmodel's Functional Areas.

Table 2 — Transmodel's Functional Areas.

Transmodel area number	Transmodel area name
1	Study passenger behaviour and determine the demand
2	(Re)design the network
3	Plan the service to be offered
4	Define a fare policy
5	Plan detailed timetables
6	Schedule vehicle blocks
7	Schedule driver duties
8	Prepare driver rosters
9	Manage drivers
10	Manage vehicles
11	Perform and control the driving process
12	Plan and organise passenger information
13	Provide passenger information on the planned service
14	Provide passenger information on the actual service

15	Plan maintenance work
16	Plan and process maintenance orders
17	Control maintenance work
18	Manage incidents
19	Analyse maintenance actions
20	Manage material
21	Organise sales
22	Operate sales
23	Validate/check and charge
24	Manage money transactions
25	Manage statistical results
26	Manage personnel
27	Define marketing policy
28	Improve and maintain the relation with the public

5.2.3.1 Provide the data for network design, analysis and organisation

NETWORK-001: (Re)design the network (Transmodel)

NETWORK-002: Plan the service to be offered (Transmodel)

NETWORK-003: Plan detailed timetables (Transmodel)

5.2.3.2 Provide information to passenger information systems

PASSENGER-001: Provide passenger information on the planned service (Transmodel)

PASSENGER-002: Provide passenger information on the actual service (Transmodel)

5.2.3.3 Provide information to other service providers

5.2.3.4 Provide information to AVMS and other operational units

VEHICLE-001: Schedule vehicle blocks (Transmodel 6)

VEHICLE-002: Perform and control the driving process (Transmodel).

5.2.3.5 Provide information about maintenance work and temporary services, restrictions

MAINT-001: Plan maintenance work (Transmodel)

VALIDITY-001: Provide support for defining object validities

5.2.3.6 Provide fare-related information

FARE-001: Define a fare policy (Transmodel)

FARE-002: Organise sales (tickets) (Transmodel)

FARE-003: Validate/check and charge (Transmodel)

5.2.4 Object Lifecycle Support Use Cases

VERSIONING-001: Mark stops, routes, timetables as

- under construction;
- suggested;
- released;
- obsolete.

VERSIONING-002: Provide support for delivery versioning

5.2.5 Security Use Cases

SECURITY-001: Authorisation

SECURITY-002: Authentication

SECURITY-003: Access rights

SECURITY-004: Data integrators that collect and re-distribute data from various feeds to different channels shall obey rules about who is allowed to upload particular data records and who is allowed to receive them.

5.2.6 Excluded Use Cases

Table 3 — Examples that illustrate what is out of the scope of NeTEx.

Excluded use cases or business domains	Source	Reason for exclusion
1) To identify monitoring points to the public as stations/stops/platforms	TRANSMODEL RT#01	Same as SIRI reference data exchange (TransModel area 17) Note: NeTEx will include a physical model for the exchange of IFOPT stop data.
2) To relate situations to the network and to journeys in the network	TRANSMODEL OPS#02	Same as SIRI SX reference data exchange (TransModel area 18).
3) To report unplanned disruptions within the network in relation to specific stops or arrivals at a stop	TRANSMODEL UTMC#02	Same as SIRI SX reference data exchange (TransModel area 18).
4) To show signs on maps and virtual-reality displays	TRANSMODEL GIS#02	Technology specific.
5) Detailed use of parking for passengers and vehicles		Not scheduled information (see DATEXX II).
6) Traffic light prioritisation and control of roadside traffic management equipment		Note: Reference data (e.g. beacons, sensors) can be part of point reference data.
7) Road crossings and road interchange data		No relation to public transport.
8) Control actions		Possible future extension.
9) Driver duties and rostering		Possible future extension.
10) Ressource plan		Possible future extension.
11) Freight		Freight will be a transport mode, in order to plan track utilization, but there will be no detailed freight planning.
12) Public transport services performance monitoring		NeTEX can provide such systems with planned timetable data, but will not deal with actual timings.

5.3 Use Cases

5.3.1 Requirements Table

NOTE 1 Version filter means a filter by version and validity condition (e.g. date, time from, until).

Table 4 – Overview of all content use cases

Use Case	Focus on	Filtering	Comments
NETWORK-001-001 Provision of different network versions for simulation, planning or analysis tools	Network	by version, by planning state, by mode, by operator, by line, by area,	NeTEx Part 2
NETWORK-001-002 Planning and understanding the accessibility of an area or region	Stop places, Stop points, Vehicles	by version, by area, by operator, by line, by accessibility attributes	
NETWORK-001-003 Provision of several network descriptions corresponding to different level of service	Network	by version, by planning state, by mode, by operator, by area, by line	
NETWORK-002-001 Identifying the stops and connection stops in a PT network	Stops, Routes	by version, by mode, by operator, by area, by line	
NETWORK-002-002 Identifying the interchange and connection points in journeys	Connection links	by version, by operator, by mode, by area, by line	
NETWORK-002-003 Definition of service policy	Service patterns	by version, by mode, by operator, by area, by line	
NETWORK-002-004 Provision of flexible transport topology	Stops, Routes, Service patterns	by version, by mode, by operator, by area, by line	
NETWORK-003-001 Electronic registration of public transport routes	Timetables	by version, by planning state, by operator, by mode, by line	NeTEx Part 2

NETWORK-003-002 Planning frequency based services	Timetables	by version, by planning state, by operator, by mode, by line	NeTEx Part 2
NETWORK-003-003 Provision of means to exchange timetables of different precision	Timetables	by version, by planning state, by operator, by mode, by line	NeTEx Part 2
NETWORK-003-004 Provision of timetables that specify passing times at timing points only	Timetables	by version, by planning state, by operator, by mode, by line	NeTEx Part 2
NETWORK-003-005 Provision of timing information related to flexible transport services	Timetables	by version, by planning state, by operator, by mode, by line	NeTEx Part 2
PASSENGER-001-001 Provision of passenger information on the planned service	Network, Stops, Lines, Routes,	by version, by mode, by operator, by line, by area, by data provider	NeTEx Part 2
PASSENGER-001-002 Relating physical stops to complex data objects such as stations	Stops	by version, by mode, by area, by data provider	
PASSENGER-001-003 Planning the usage of platforms	Stop points	by version, by mode, by operator, by area, by data provider	
PASSENGER-001-004 Planning the shared use of platforms	Stop points	by version, by mode, by operator, by area, by data provider	
PASSENGER-001-005 Planning for flexible pick-up points for demand-responsive and other services	Stop places, Stop points	by version, by mode, by operator, by area	

PASSENGER-001-006 Assignment of labels	Stop places, Lines	by version, by mode, by area, by operator, by data provider, by language	
PASSENGER-001-007 Finding stops and stations for a place	Stop places, Routes	by version, by mode, by area, by operator, by data provider	
PASSENGER-001-008 Describing the relation of the stop to nearby objects and landmarks	Stop places	by version, by area, by mode, by operator	
PASSENGER-001-009 Relating physical access points to stations	Stop places	by version, by area, by mode, by operator, by data provider, by accessibility attributes	
PASSENGER-001-010 Renaming stops	Stop places	by version, by data provider, by operator,	
PASSENGER-001-011 Finding DRT and flexible services	Stop places	by version, by area, by mode, by operator	
PASSENGER-001-012 Exchange the topology of On-demand transport	Service patterns, Stop places	by version, by area, by mode, by operator, by line	
PASSENGER-001-013 Planning journeys between places	Localities, Stops, Connection links, Timetables	by version, by area, by mode, by operator, by line, by data provider	NeTEx Part 2
PASSENGER-001-014 Planning journeys through the network including detailed connection times	Stops, Connection links, Timetables	by version, by area, by mode, by operator, by data provider, by line,	NeTEx Part 2

PASSENGER-001-015 Planning journeys through the network for mobility restricted users	Stops, Connection links, Timetables, Accessibility attributes	by version, by area, by mode, by operator, by data provider, by line	NeTEx Part 2
PASSENGER-001-016 Identifying stop points when exchanging data between distributed journey planners	Common (global) identifiers for stop points, localities and lines	by version, by area, by mode, by data provider	
PASSENGER-001-017 National language support	Stop places, Lines, Localities	by version, by area, by mode, by data provider, by operator	
PASSENGER-001-018 Managing car parks as connection point between the road network and the PT network	Stop components, Connection links	by version, by area, by mode, by data provider, by operator	
PASSENGER-001-019 Distributed assignment of responsibility for data management	Data ownership attributes	by version, by area, by mode, by operator, by data provider	
PASSENGER-001-020 Referencing locations to schematic maps	Service Patterns, Interchanges	by version, by area, by mode, by operator, by line, by data provider	
PASSENGER-001-021 Linking multimedia content to public transport objects	Object references	by version, by area, by mode, by operator, by line, by data provider, by media type	
PASSENGER-002-001 Provision of passenger information on actual service	Object references for stop points, lines, journey patterns, journeys	by version, by area, by mode, by operator, by line, by data provider	

OTHER-001-001 Showing stops on a map	Geographical position of stop points and routes	by version, by location (area or point), by mode, by operator, by line, by data provider,	
OTHER-001-002 Provision of necessary information for geographic information or Inspire	Network	by version, by area, by mode, by operator, by data, by type (only geographic information)provider	
VEHICLE-001-001 Plan vehicle demand for timetable	Journeys, Blocks, Garages, Vehicle types	by version, by mode, by operator, by line, by group of line, per garage	NeTEx Part 2
VEHICLE-002-001 Exchange between scheduling system and AVMS	Timing patterns, Connection links, Activation points	by version, by mode, by operator, by line, by validity	NeTEx Part 2
VEHICLE-002-002 Provision of reference data to manage incidents	Common identifiers for stop places, routes, lines and journeys	by version, by mode, by area, by operator, by line, by data provider	NeTEx Part 2
VEHICLE-002-003 Managing the connections between stop points at an interchange	Stop points, Connection links	by version, by mode, by area, by operator, by validity	
VEHICLE-002-004 Planning transfer margins for guaranteed connections between services	Connection links	by version, by mode, by area, by operator, by line	NeTEx Part 2

VEHICLE-002-005 Identifying the monitoring points exchanged between different AVL systems	Stop points, Connection links	by version, by mode, by area, by operator, by line	
VEHICLE-002-006 Identifying the relationship of the monitoring points to other points in an interchange	Stop points, Connection links	by version, by mode, by area, by operator, by data provider	
VEHICLE-002-007 Managing the traffic for an area	Stop points, Routes	by version, by mode, by area, by operator	
VEHICLE-002-008 Management of statistical results	Common identifiers for PT objects	by version, by mode, by area, by operator, by data provider, by type of PT object	NeTEx Part 2
VEHICLE-003-001 Management of vehicles	Vehicle blocks	by version, by mode, by line, by operator, by data provider, by equipment	NeTEx Part 2
MAINT-001-001 Planning of maintenance work	Stop points, Routes, Journey patterns, Timetables, vehicle blocks	by version, by mode, by operator, by area, by line, by validity	NeTEx Part 2
VALIDITY-001-001 Temporary change in stop availability	Stop points, Routes, (timetables)	by version, by mode, by operator, by area, by line, by validity	
VALIDITY-001-002 Temporary timetable and service changes	Timetables, Validities	by version, by mode, by operator, by area, by line, by validity	NeTEx Part 2

NOTE 2 Data Producers should support a “capability request” telling which filters it supports. It is recommended that data producers support filter by version, line and mode at least.

5.3.2 Collection of Use Cases

The list of use cases in this section presents a variety of public transport activities that could benefit by using the NeTEx data exchange interface. So in most cases the focus lies on the data contents that are to be exchanged (content use cases).

The use cases are not directly NeTex use cases. The following tables describe how NeTex is used to facilitate these use cases and which requirements for NeTex originate from them.

Use cases number 1-9 originate primarily from TRANSMODEL, numbers 11- 46 from IFOPT.

The numbering may have gaps because of removal of use cases (numbers are currently persistent).

5.3.2.1 Provide the data for network design, analysis and organisation

NETWORK-001: (Re)design the network (Transmodel)

NETWORK-002: Plan the service to be offered (Transmodel)

NETWORK-003: Plan detailed timetables (Transmodel)

5.3.2.1.1 NETWORK-001: (Re)design the network (Plan the main characteristics of the service)

Use Case: NETWORK-001-001(#1)	
Name	Provision of different network and timetable versions (suggested, real, etc.) for simulation, planning or analysis tools (like design network tools).
Source	TRANSMODEL IFOPT LA#01
Description	Area 2: Re-Design the network: The process of defining the network involves the following activities: Take all global decisions regarding the network structure, taking into account the defined quality of service that has to be offered and the guidelines from management (budgets, policy). Define the area that will be served. Define the stop points. Define the routes. Area 3: Plan the service to be offered: Take all global decisions regarding the quality of the service that has to be offered, within the limits of budgets and other guidelines from management (define minimum and maximum travel times, interchange points etc.). Define the service quality level and quality objectives. Define the frequency on the line(s). Define the type of service (demand responsive, line oriented etc.). Define the service constraints (boarding or alighting restrictions). Calculate the global running times. Calculate the global need for vehicles and drivers. Multiple versions of networks can co-exist for planning purposes.
NeTEx contribution	NeTEx provides a means to exchange the actual state of the Network. NeTEx provides a means to exchange suggested services and to exchange different network versions.
Main actors	Strategic planning systems, timetable planning, traffic flow prediction systems.

Main objects	Such type of exchange may need less precise information than for passenger information or AVMS. It may be more “frequency oriented” (NeTEx part 2) for timing information for example. Need exchange of Journey Patterns (sequence stop and timing points).

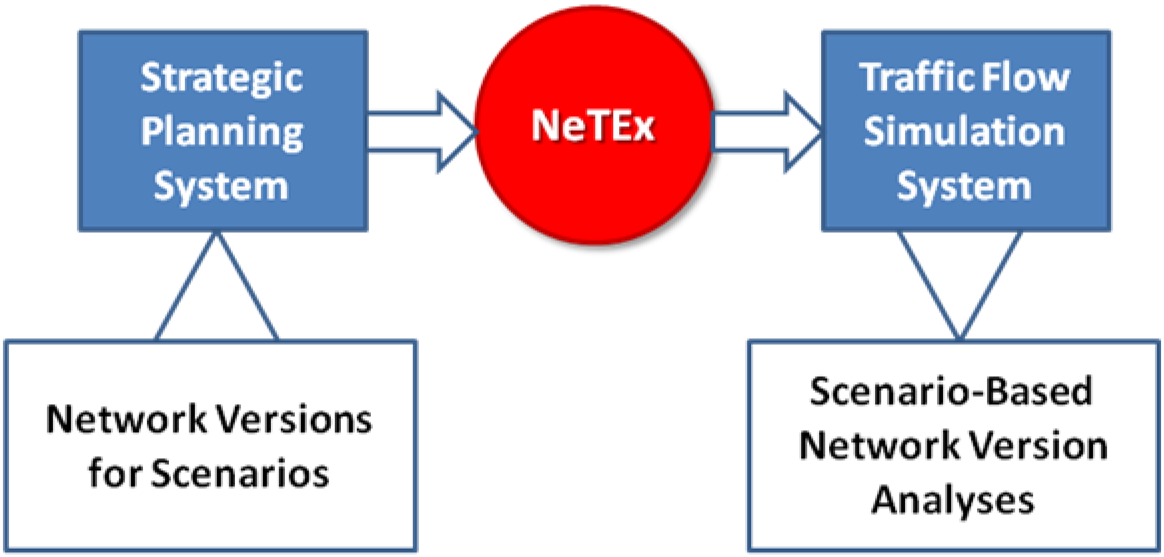


Figure 2 – Illustration for use case *Provision of different network and timetable versions*

Use Case: NETWORK-001-002(#2)	
Name	Planning and understanding the coverage of an area or region by public transport
Source	IFOPT : LA#02.
Description	In order to plan the adequate accessibility on the PT service within a region, a detailed model of actual coverage is needed.
NeTEx contribution	NeTEx allows for describing the availability and accessibility of stop places and stop points as well as the PT service offer within a region.
Main actors	Urban development departments, authorities.
Main objects	Stop places, stop points, accessibility attributes.

Use Case: NETWORK-001-003 (#3)	
Name	Provision of several network descriptions, corresponding to different level of service that may be applicable when a strike or a specific event (flood, pollution level, etc.) occurs.
Source	NeTEx
Description	In addition to the currently active PT network it may be beneficial to exchange network versions that reflect different situations that may occur regularly (e.g. sports events), sometimes (e.g. traffic reduction due to high pollution), rarely (e.g. flooding of city areas) or hopefully never (e.g. catastrophic scenarios).
NeTEx contribution	PT network topology.

Main actors	Operators, authorities.
Main objects	Validity definition of stops and lines.

5.3.2.1.2 NETWORK-002: Plan the service to be offered in detail

Use Case: NETWORK-002-001(#4)	
Name	Identifying the stops and connection stops in a PT network when creating a schedule
Source	IFOPT : SCHED#01
Description	Schedulers of Public Transport timetables plan and schedule journeys for PT vehicles that will move between designated stops and timing points. To do this the stops and timing points shall be given unique identifiers that can be used to reference unambiguously the scheduled stop points of the journey. The identifiers for internal scheduling and operational use may be different from those given out on the timetable or other information to the public. Public identifiers may comprise a label, a short code, a long code or all three. *EXAMPLE 1 London Heathrow, Heathrow, LHR, 4900345671, or Kings Cross, Kings Cross Rail Station, KGX, and '9100KINGX'.* Timing points that are not stops are not normally published to the public but will still need to be identified internally. There may be multiple internal identifiers for the same stop point relating to reservation, operation or control systems. A scheduled stop point is, in effect, a planned break in the journey for alighting or boarding that may be, but is not necessarily, related to an actual physical stop point. For example, for scheduled stops at large stations the vehicle journey may be additionally assigned to a specific platform or stance within the station. *EXAMPLE 2 Kings Cross, Platform 9* The scheduled stop point exists independently of this assignment, which may change when the vehicle journey occurs. To prepare a schedule, a scheduler normally will begin by creating a service pattern, a list of the designated stops points in sequence that defines a route. Where an existing set of reusable stop points already exists, the list will be selected from these existing points. If there is no general system the stops will be assigned arbitrary names on an ad hoc basis, and identifiers allocated from a designated name scope. If the schedule involves a new stop there may be a need to define a stop and allocate new identifiers. The definition of the route may be identified and exchanged independently of the timetable.
NeTEx contribution	NeTEx provides means to exchange the detailed network topology, including stop places, timing points, routes and journey patterns.
Main actors	Authorities and operators.
Main objects	Stop places, boarding position, quays, access points, path, interchange and connection links, routes, journey patterns, timing points. Timing information can be published/guaranteed in certain connection links only (attribute of connection links).

Use Case: NETWORK-002-002(#5)	
Name	When creating a schedule, specifying the connection points where interchange between journeys takes place.
Source	IFOPT : SCHED#03

Description	Places in the network where there is a possibility of making a transfer may be designated as points where interchanges take place and explicitly identified by connection links or other grouping mechanisms that associate stop points. To represent the possibility of transfer between two stop points, schedulers may create connection links which may have unique identifiers within the system so that they may be reused. Connections may also be made between services using the same stop point. Connections ideally should represent the physical movement between stop points; any contingency time at the connection point to allow for irregular operation of services should be identified separately. Certain connection points may be favoured over others and be assigned weightings.
NeTEx contribution	NeTEx (part 1) provides a way to exchange all the topological and geographical information about connections.
Main actors	Authorities and operators.
Main objects	Stop places (with their neighbours). Geographic location. Infrastructure links (to discover if there's a way on the road to go from one stop place to another). Connection links, path links and navigation path.

Use Case: NETWORK-002-003(#6)	
Name	Definition of service policy.
Source	NeTEx
Description	For concessionary reasons sometimes passengers are only allowed to alight from a service after the vehicle has left a certain area.
NeTEx contribution	NeTEx must be able to express rules like: If you board at that stop, you can’t alight at these stops (Routing constraints).
Main actors	Authorities and operators.
Main objects	Service policy.

Use Case: NETWORK-002-004(#7)	
Name	Provision of flexible transport topology
Source	NeTEx
Description	FTS (Flexible Transport Service) and DRT (Demand Responsive Transport) are more and more used in public transport and need to be described and exchanged. FTS being more generic than DRT since flexibility may not be directly linked to the demand, but may be related to some operating needs or cost optimisations. The following flexible service types are to be considered: — Virtual line service. — Flexible service with main route. — Corridor service (Flexible service without main route). — Fixed stop area-wide flexible service.

	— Free area-wide flexible service. — Mixed types of flexible service.
NeTEx contribution	NeTEx (part 1) provides a way to exchange all the topological and geographical information about FTS and DRT.
Main actors	Authorities and operators.
Main objects	Stop places, stop points, zones, lines, routes, journey patterns.

5.3.2.1.3 NETWORK-003: Plan detailed timetables

Use Case: NETWORK-003-001(#8)	
Name	Electronic registration of public transport lines and services
Source	NeTEx
Description	In some countries or regions PT operators have to tender for the concession to operate a PT line. This is due to a legal requirement, which makes sure that passenger information is available a certain time before service changes take place. Operators send their application together with the timetable they intend to run to the local authority. The local authority chooses the most beneficial tender and awards the concession to the winning transport operator. The timetables of the winning PT operator are published when the services go in operation. Tender documents including the suggested timetables shall not be disclosed to competitors.
NeTEx contribution	NeTEx part 1 provides means to define a common understanding of stop places, stop points and connection links. It is able to express ownership of data. The definition of timetables is up to NeTEx part 3.
Main actors	PT operators, authorities.
Main objects	Stop places, stop points, journey patterns, timetables.

Use Case: NETWORK-003-002 (#9)	
Name	Planning frequency based services.
Source	NeTEx
Description	For non-timetabled services, so called “Frequency based services”, the frequency of the service at each stop for any time and day type shall be planned and published.
NeTEx contribution	NeTEx part 2 allows to define frequency-based services.
Main actors	Authorities, operators.
Main objects	Journey patterns, timetables.

Use Case: NETWORK-003-003 (#10)	
Name	Provision of means to exchange timetables of different precision.

Source	NeTEx
Description	Some timetables, in particular in urban rail environments, are planned with a high degree of accuracy (with a precision of a few seconds). But in other cases, especially in rural settings or when planning demand-responsive transport, arrival and departures times cannot be stated very precisely in advance.
NeTEx contribution	NeTEx allows to express how precise arrival and departure times in timetables are. (Netex Part 2)
Main actors	Operators, authorities.
Main objects	Journey patterns and timetables.

NOTE Potentially it could be helpful to have precision information on other data, e.g. precision of coordinates.

Use Case: NETWORK-003-004 (#11)	
Name	Provision of timetables that specify passing times at timing points only.
Source	NeTEx
Description	One approach for planning timetables is to distinguish between timing points and non-timing points. The timetables then contain only the passing times of the services at the timing points. To determine the passing times at non-timing-points interpolation methods shall be used. For interpolation the distances between stops shall be known. Distances between stops belong to the route layer of the network. This means that projections of the service stop points layer to the geometric route network layer shall be possible.
NeTEx contribution	NeTEx considers only linear interpolation based on stop distances on the route network layer. NeTEx provides projection mechanisms between timing patterns, service patterns and the route network.
Main actors	operators, authorities, passenger information systems.
Main objects	Service pattern, Timing pattern, timing points, timing links, routes.

Use Case: NETWORK-003-005(#12)	
Name	Provision of timing information related to flexible transport services.
Source	NeTEx
Description	Flexible transport is characterised by flexible routing and scheduling. Flexible services can operated on regular line topologies or on a flexible topology. Flexibility may can be applicable for a all service but also for partially flexible service or for "mixed types of flexible service" inside the same journey. Several types of flexible services are to be considerered: — Fixed passing times (meaning schedules passing time: there is a timetable, but the service will only run under condition, mainly sufficient demand). — Dynamic passing times. — Fixed headway frequency (in this case, a maximum waiting time is available, but no passing times are defined, all is done dynamically depending on the demand or any other needs).

NeTEx contribution	NeTEx (part 2) provides a way to exchange all the timing information about FTS and DRT. Note that NeTEx point of view on flexible services is a passenger information point of view, not an operator point of view.
Main actors	Authorities and operators.
Main objects	Journey patterns and timetables.

5.3.2.2 Provide information to passenger information systems

- PASSENGER-001: Provide passenger information on the planned service (Transmodel)
- PASSENGER-002: Provide passenger information on the actual service (Transmodel)

5.3.2.2.1 PASSENGER-001: Provide passenger information on the planned service

Use Case: PASSENGER-001-001 (#13)	
Name	Provision of passenger information on the planned service.
Source	TRANSMODEL
Description	Give information on the planned services (i.e. information available before the 24 hours operational cycle) to passengers, either by display video, audio (incl. phone desks) or online (web, wap, etc.). This information can be passive or delivered on request. Provide passive information on the planned service by electronic or paper media at stops, on-board or through other distribution channels. Deliver on request information on the planned service (trip preparation help): — Journey planning. — Timetable passing time info (NeTEx part 2 only). — Mapping&geographic information (optional). — Fare information (NeTEx part 3).
NeTEx contribution	NeTEx provides a means to exchange the full needed PT network and timetable and fares description (network topology for part 1). NeTEx can provide information about the data owner, the data administrator and the data source for each data item. Referenced organisation can be described by NeTEx. NeTEx is only a data exchange interface (data carrier). Nothing will be defined on responsibilities (has to be stated in a specific contract). NeTEx provides information about “media” equipment at stop places and on-board.
Main actors	NeTEx is used between scheduling systems (producer) and passenger information systems (consumer) or between several passenger information systems.
Main objects	Stop places (boarding positions, quays, etc.), lines, routes, journey patterns, timetables, places, connection links, access links, POIs. Additional (generic) information like notes, line colours, descriptive text, icons etc. Stop Place component: passenger information equipment. Data source and owner. Organisation types: authorities, operators, service providers.

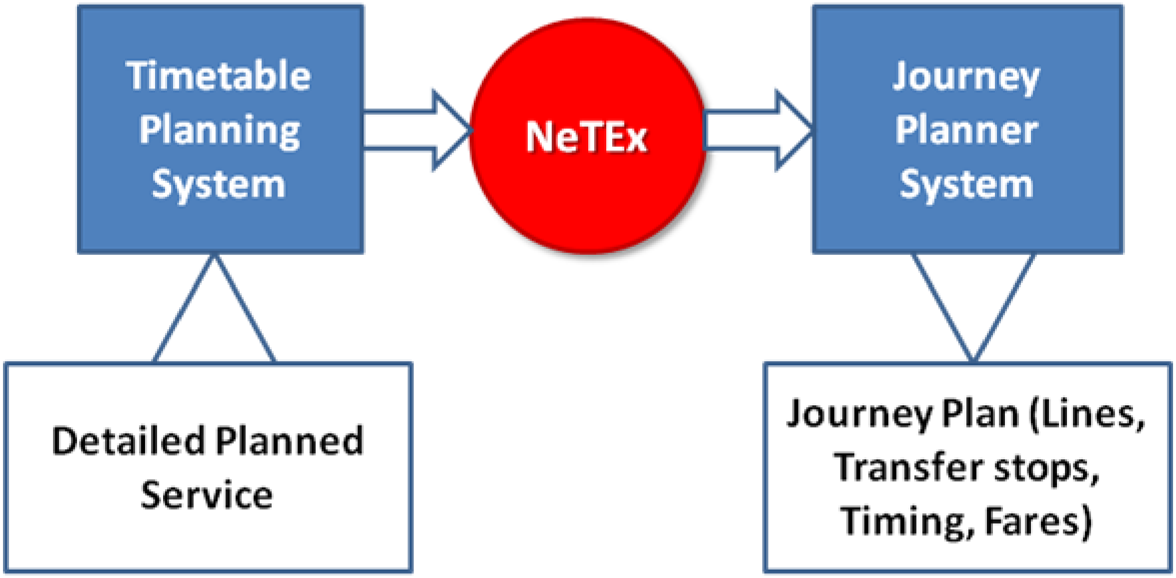


Figure 3 – Illustration for use case *Provision of passenger information on the planned service*

Use Case: PASSENGER-001-002 (#14)	
Name	Relating physical stops to complex data objects such as stations.
Source	IFOPT : SCHED#06
Description	In preparing a schedule that has a scheduled stop point at a large station or interchange, schedulers may need to give an indication of where in the interchange each stopping point is located. This advice may be given at various levels. *EXAMPLE Waterloo, Waterloo East, Waterloo East platform 6.* When planning schedules, transfer times may be a necessity that need to be recorded for different types of passengers.
NeTEx contribution	NeTEx provides a way to exchange a detailed physical stop description based on the IFOPT Stop Place concept.
Main actors	Authorities, operators and service providers.
Main objects	Stop places and stop place components. IFOPT Stop Place concept.

Use Case: PASSENGER-001-003 (#15)	
Name	Planning the usage of platforms so that sections of a train may be related to sections of the platform.
Source	IFOPT : SCHED#07

Description	Accommodation on trains is not always the same in each section - first class seating, catering accommodation, sleeping accommodation etc will be allocated to different sections of a train. In some cases trains will be split or combined during the journey - so passengers making journeys between two stop points may need to be in a particular section of a train. Also on express trains, tickets are also often pre-assigned to a specific seat in a specific carriage. Both of these situations lead to a need to identify to the public the section(s) of the stop point (platform) at which a section of a train with relevant features will stop. This may be in formats such as "platform 3, front part of the train" or "platform 3, yellow zone" or "Platform 3, section D".
NeTEx contribution	NeTEx provides a way to exchange a detailed and unambiguous platform description. Also linked to NeTEx Part 2 (timetable and journey information).
Main actors	Operators, passenger information systems, AVLS.
Main objects	Quay, boarding position, vehicle stop point, vehicle stop position.

Use Case: PASSENGER-001-004 (#16)	
Name	Planning the shared use by different trains at the same time of the same single platform that is broken down into sectors.
Source	IFOPT : SCHED#08
Description	At some major rail stations it is possible for a single "platform" to be occupied by more than one separate train, each departing at a different time for different destinations. This is a special version of previous use case. The platform may have a secondary split into sectors "platform 4A", "platform 4B", to indicate to passengers where they should stand in order to board their designated carriage efficiently, or to access the correct part of a multi-part train.
NeTEx contribution	NeTEx provides a way to exchange a detailed and unambiguous platform description. A multimodal use of the same platform is possible. Also linked to NeTEx Part 2 (for timetable and journey information).
Main actors	Operators, passenger information systems, AVLS.
Main objects	Quay, boarding position, vehicle stop point, vehicle stop position.

Use Case: PASSENGER-001-005 (#17)	
Name	Planning for flexible pick up points for demand responsive and other services.
Source	IFOPT : SCHED#09
Description	Demand responsive services, and Hail & Ride services, seek to serve passengers by stopping at a location close to their door on a pre-ordained route. That route may be one which is followed regularly, or it may be one that is created dynamically for each journey to meet specific passenger requirements that have been booked in advance. In these cases there are both operational and public information needs to be able to identify stop places which may represent a linear section of a predetermined route, or a zone of streets, on which the vehicle will stop on one or more occasions as necessary on an individual vehicle journey. There is a need both for codes for such stops and labels that can be used for the public provision of information.
NeTEx contribution	NeTEx provides a way to exchange the description of pick up points for demand responsive services.

Main actors	Operators, passenger information systems, booking call centres.
Main objects	Stop places that can be dynamically allocated in a defined area (geographical location may be predefined points inside the area, or any point inside the area). Service information (such as booking phone number, web site, minimum booking delay, etc) should also be available.

Use Case: PASSENGER-001-006 (#18)	
Name	Assignment of labels for destinations, places, and stops, for use in destination boards, stops, tickets, on-board displays, announcements etc
Source	IFOPT : SCHED#10
Description	Schedulers need to assign consistent unambiguous names to stops, and stop points, on all forms of printed and electronic media. Normally there will be a number of canonical labels to use for different media types and footprints, such as a short name (e.g. Kings Cross), a long name (e.g. London Kings Cross Station), a name to print on tickets, (e.g. London Kings Cross)a name to use in SMS messages (e.g. KingsX), etc. Typically systems will hold a set of definitive labels that follow uniform typographical and spelling conventions for example as to hyphenation, capitalisation, use of abbreviations, etc. *EXAMPLE 1 Frankfurt. a. m. vs. Frankfurt am M,. or St. Jean-en-Provence vs. St Jean en Provence)).* This is to ensure a consistent brand, for the service provider and to make the presentation of lists of similar names ergonomic to the user. Different delivery systems will need to compose different labels for the same stop in different contexts. Where there are identical or similarly-named elements it may be necessary to include additional qualifiers such as the name of the administrative area to discriminate between them. Destination displays on buses and on signs at stop points may be constrained to be within a limited number of characters, and formatted in a style which is appropriate to the context of the service being provided. A destination may be described differently between a local service and a long-distance one. *EXAMPLE 2 Consider two services heading for Bristol, Marlborough Street Bus Station* *- the local service within Bristol will not state the name of the town, for instance, and just say "Marlborough Street Bus Station", whilst the long-distance service may only state the name of the town and ignore the detail of where in the town the service stops, "Bristol".* Journey planning systems reference stops and stations in many different ways; both associating them with topographical and address data to support place finding, and describing the scheduled stop points of the journey. Suppliers of journey planning systems typically shall assemble topographical address, stop and timetable data into a normalised format in order to transform it into the internal representation used by their journey planning engines. Stop data provides a crucial role in this process as it constitutes a separate and distinct information layer that can be used to relate timetable data to GIS data layers. AVL systems will use these same headings for dynamic displays. Mechanisms are needed to distribute changes to reflect renaming and the addition of new stops.
NeTEx contribution	NeTEx provides a way to exchange all the labels used to name public object such as lines and stop places.
Main actors	Authorities, operators, passenger information systems.
Main objects	Naming attributes of stop places and lines.

Use Case: PASSENGER-001-007 (#19)	
Name	Finding stops and stations for a place (point, locality, or general area) and their public transport services.
Source	IFOPT : JP#02
Description	Passengers using journey planning systems will need to find the nearest points of access to PT for the origin and destination of their trip. They may not necessarily know of the existence of, let alone the name of, the relevant stop or stopping point, and may use a variety a strategies to indicate their location: entering a point of interest name, a place name, an address or postcode, or clicking on a map. The journey planner will use gazetteers and other mechanism to make a location-based search to find the nearest stop or stops that may be relevant. In order to be able to do this the journey planner shall be able to relate the stop to a GIS context and/or to associate it with a topographical name. Journey planners may support multiple aliases for both places and stops. *EXAMPLE London Kings Cross', 'Kings Cross', 'Kings Cross station.* The precision of the search may vary widely in different contexts. To take a local bus the passenger will be interested in a few stops within easy walking distance. For long-distance travel passengers may tolerate a much larger search area: typically journey planners will support specific PT related concepts of place such as London any airport or London (any mainline station)". Moreover, passengers will be interested in knowing the services available at a specific stop. Having found a stop (either by a location based search or by explicitly entering a name or public stop code), a passenger will be shown the services available at the stop. This could be either as a departure board, or as a list of modes (for example to see how they might travel to an airport). The stop may be identified either by its canonical name, public short code or both - thus providing a means of familiarising passengers with the alternative ways of accessing information about the stop place. The results will also show the destinations of different services as either place names, stop names or Point of Interest names.
NeTEx contribution	NeTEx provides ways to deliver lines and the stops they are calling at. Thus enabling to find at which locations and points of interest are served by each line.
Main actors	Authorities, operators, passenger information systems.
Main objects	IFOPT Stop places and components: Names, geographical location, address. Infrastructure links (mainly road). Point of Interest (connected to Stop Places). Lines, Journey patterns, Connection links

Use Case: PASSENGER-001-008 (#20)	
Name	Describing the relation of the stop to nearby objects and landmarks to help identify them to the public.
Source	IFOPT : JP#05
Description	A journey planner needs to be able to offer a passenger detailed guidance on how to find a stop. The name of a stop may be additionally labelled or qualified in terms of places or POIs in the immediate vicinity. Map based presentations may show the relative positioning and access paths needed to reach the stop in relation to landmarks and Points of Interest.

NeTEx contribution	NeTEx provides means to exchange all attributes of stops points and stop places, including their relative position to nearby landmarks.
Main actors	Authorities, operators, passenger information systems.
Main objects	Location descriptive information (not geographic coordinates).

Use Case: PASSENGER-001-009 (#21)	
Name	Relating physical access points (entrances) to complex objects such as stations and POIs.
Source	IFOPT : JP#06
Description	Large transport interchanges and POIs may have a number of different entrances, addresses, parking areas etc and a journey planner may need to indicate to the user which entrance they should use and how to identify it in relation to the urban context.
NeTEx contribution	NeTEx provides means to exchange all attributes of stops points and stop places, including all their stop areas (entrances, parking areas, etc).
Main actors	Authorities, operators, passenger information systems.
Main objects	Stop place areas.

Use Case: PASSENGER-001-010 (#102)	
Name	Renaming stops.
Type	Content use case.
Source	NeTEx
Description	Physical stops sometimes happen to be re-named. In such an event all stop attributes and service information remain unchanged, except the name of the stop or station. Transport authorities plan such re-naming in advance and inform other parties (e.g. passenger information systems) about the change and when it will be effective.
NeTEx contribution	NeTEx provides means to exchange naming attributes and their validity periods.
Main actors	Authorities, passenger information systems.
Main objects	Stop names and validities.

Use Case: PASSENGER-001-011 (#22)	
Name	Finding DRT and flexible services.
Source	IFOPT : JP#14 and SCHED#09

Description	Some stops do not have simple point geometries. *EXAMPLE* *Hail and Ride sections: These will typically be a contiguous section of road marked by a start and end point* *Flexible zones: these will typically be arbitrary polygons* *Variable bays: These will be dynamically allocated bay assignments within a bus station.* Journey planners will typically use additional mechanism to find such stops and additional labels to describe such stops.
NeTEx contribution	through stop places and components.
Main actors	Authorities, passenger information systems.
Main objects	Stop places.

Use Case: PASSENGER-001-012 (#23)	
Name	Exchange the topology of On Demand Transport (flexible line) with (or from) call and booking centres, information systems or journey planners.
Source	NeTEx
Description	Demand responsive transport (DRT) operates in several forms: Some services follow a journey pattern with defined stops, but omit some stops if no passenger wants to board or alight. Other services offer to pick up or set down passengers at defined stops within an area. The most flexible services offer to carry passengers to their homes (within a service area). Information systems and call centres that advertise demand responsive transport need to know about the service area and service type of DRT.
NeTEx contribution	NeTEX part 1 defines the service area and service type of DRT. NeTEx part 2 defines the service timetables and availability times of DRT. NeTEx part 3 defines the fare policies of DRT.
Main actors	DRT operators, passenger information systems.
Main objects	DRT lines, DRT service areas.

Use Case: PASSENGER-001-013 (#24)	
Name	Planning journeys between places.
Source	IFOPT : JP#07

Description	Passengers may often wish to use topographical place names to state their travel objectives. *EXAMPLE Cambridge to Oxford.* The topographical scope may be quite large, for example London to Paris reflecting considerable flexibility in the user's choice of stops. The journey planner needs to be able to associate topographical places with stop places at different scales appropriate to the scale and mode of travel, and in accordance with common sense notions of place - for example airports may be outside the physical boundaries of a city, but are the air access point stops to the city. A within-city journey would consider local stations, whilst an intercity journey would consider mainline termini as the starting points. Journey planners provide passengers with both summary and detailed itineraries that describe the way points on their journey. The stop names and any public codes shown on the itinerary should correspond to the signage used on the stops so that users can follow their progress. The departure stop, interchange points and sometimes the arrival stop may also include the platform identifier.
NeTEx contribution	NeTEx provides means to deliver all stops within a place and the modes of transport that serve them. Timetables of part 2 will allow for estimating how frequent stops are served.
Main actors	Authorities, operators, passenger information systems.
Main objects	Localities, places, stop places, stop points, lines, journey patterns, platform identifiers, connection links.

Use Case: PASSENGER-001-014 (#25)	
Name	Planning journeys through the network, including detailed connection times and detailed guidance for making an interchange between two services over a connection.
Source	IFOPT : JP#11 and JP#13
Description	In planning a multi-leg journey plan through a network, a journey planning system will take into account the transfer time needed to interchange between services at an interchange point. Depending on the sophistication of the journey planning system and the availability of data, the individual timings required for transfers between different stop areas or stop points within an interchange may be taken into account or just an average time for any transfer at the station may be used. In particular, in the case of frequency based services enough information about the service frequency shall be provided to the journey planning system in order to allow for sufficient waiting time. Journeys may be made by passengers on foot, on a bicycle, in a car or using a combination, for example park and ride or kiss and ride. The stop model should support journey planning of intermodal journeys. Journey planners providing journey plans for journeys involving the traversal of large interchanges will be concerned to provide detailed guidance on the navigation of the interchanges, typically as step by step instructions that can be related to the signage found within the interchange building. Depending on the sophistication of the journey planning system and the availability of data, the individual paths required for transfers between different stop areas or stop points within an interchange using specifically identified navigation paths and accessibility needs may be given.

NeTEx contribution	Journeys for NeTEx part 2 Connection times are part of NeTEx part 1. NeTEx only supports guidance information if it is inside a Stop place (not on the road network, etc.) The link of Stop Places (or components) to the road has to be taken into account Guidance information may be attached to the access path links.
Main actors	Authorities, operators, passenger information systems.
Main objects	Connection links, stop places, accessibility attributes.

Use Case: PASSENGER-001-015 (#26)	
Name	Planning journeys through the network, including detailed connection times, under different constraints for mobility restricted users.
Source	IFOPT : JP#12
Description	As a refinement to previous use case, in planning a multi-leg trip through a network, a journey planner may additionally take into account the accessibility requirements and different transfer times needed for different types of user, with different walk speeds or other needs, in particular for impaired mobility accessibility. Depending on the sophistication of the journey planning system and the availability of data, the individual timings required for transfers between different stop areas or stop points within an interchange using specifically identified navigation paths and accessibility needs may be taken into account or just an average time for any transfer at the station may be used, scaled to a particular walk speed.
NeTEx contribution	Providing enough information to support the needs of mobility restricted users is of importance for NeTEx. Stops places, Equipment, accessibility on path links. Journey accessibility has to be taken into account in part 3. This comprises the assignment of timetabled trips to vehicle attributes (e.g. low-floor, lift-equipped vehicles).
Main actors	Authorities, operators, passenger information systems.
Main objects	Connection links, stop places, accessibility attributes.

Use Case: PASSENGER-001-016 (#27)	
Name	Identifying stop points when exchanging data between distributed journey planners.
Source	IFOPT : JP#10
Description	For large scale distributed Journey planning, networks of separate regional journey planners exchange queries and results collaboratively in order to establish a multi-regional journey plan. In order to do this they shall be able to use common identifiers for stops and locations. For efficient computation certain stops or other points in the transport network will be distinguished as points known to both engines.
NeTEx contribution	This point has to be agreed upon with CEN TC278 WG3-SG8 Availability to exchange a subset of Stop Place, to be used for connection between Journey Planners (a dedicated attribute/classification may point this out) … (boundary points).
Main actors	Passenger information systems, authorities.
Main objects	Stop points, adjacent region points.

Use Case: PASSENGER-001-017 (#28)	
Name	National language support (based on ISO-639 /IETF 1766).
Source	IFOPT : GEN#03
Description	Systems that display or accept textual labels may need to allow for alternative names in different National languages for textual names for entities and their terms of relationship. Typically systems will be coded in a primary language and aliases will be added for other languages, tagged with a language code.
NeTEx contribution	NeTEx allows to define names in different national languages for any textual information, e.g. stop places, points of interest, lines, places.
Main actors	Authorities, passenger information systems, operators.
Main objects	Naming attributes of stop places, lines, places.

Use Case: PASSENGER-001-018 (#29)	
Name	Managing car parks as connection point between the road network and the PT network.
Source	NeTEx
Description	Car parks are special kind of POI : when nearby to a PT Stop Place they can be used as connections between road and public transport. This information is relevant for journey planners, display and analysis of the transport offer, etc.
NeTEx contribution	NeTEx allows to exchange stop places and stop place components and their geocoded location.
Main actors	Authorities, passenger information systems.
Main objects	Stop places, stop components and connection links.

Use Case: PASSENGER-001-019 (#30)	
Name	Distributed assignment of responsibility for data management.
Source	IFOPT : GEN#01
Description	The data sets covered by NeTEx are large and belong to many different stakeholders. The task of gathering, collating and aggregating the data necessarily shall be distributed among many different organisations. Some degree of central coordination is needed to agree who is responsible for which type of data, to agree common interfaces, and to agree the partition of code namespaces so that data coded to a common standard can be aggregated without clashes as to the unique identifiers. Each data object should have: - Operational responsibility. - Owner (legal owner of the data who holds the copyright). - Data Provider (Organization which runs the producer of the data). - Administrator (Organization which is responsible for the content of the data). - “Data System” : Name Space in which identifiers are unique.

NeTEx contribution	NeTEx provides means to define ownership of data objects so that it becomes clear who is responsible to maintain the data. It also provides unique object identifiers that allow mapping of object references across system boundaries.
Main actors	Data aggregating systems, authorities.
Main objects	Ownership attributes for stop places, stop points, lines, equipment, connections links. Object reference mechanisms.

The 2 following diagrams (Figure 4 and Figure 5) describe an example of an organisational context that requires a distributed role and corresponding responsibility assignment.

In this example it is illustrated that several organisations do maintain the data that describes the informational results of the processes that are executed by them. The following roles are shown:

— Schedule planning role as executed by PTO's.

— A PTO may be responsible to plan it's operations and provide the service view on the operational schedule plans for external usage.

— Stop place accessibility planning as executed by infrastructure managers.

— Municipal infrastructure management organisations may be responsible to provide the service view on build and maintenance plans of stop places for external usage.

— Matching of schedule and stop place information as well is maintaining general integrity.

— A separate role, the information manager, is shown that maintains the matching of the different information sets. Matching of schedules and stop information is realised through the NeTEx concept of "passenger stop assignments" that match logical stops (schedule) to physical stops (stop place information).[2]

— Collating and aggregating of sets of information.

— The integrator is responsible for collection and aggregation of the various datasets that collectively describe the entire operation within an area (e.g. region or country). The result is the provision of information to users that completely and consistently describe the operation as carried out, regardless of service provider origin.

— End-usage of information.

— Information can be used by various types of users, e.g. providers of a journey planning service. The PTO's and infrastructure managers are users as well.

One major justification for the existence of an integrator is the requirement that (sub) regional information can be used consistently, regardless the partitioned structure of the original information sets that are provided by the service providers that implement their part of the services. This is illustrated by Figure 5, where an end-user requests data through a NeTEx filtered request, that describes the operation in e.g. a single city. This filtered sub-set is represented as the grey rectangle.

[2] Parts of this role could be carried out by the PTO's. For the sake of clarity of the example, these considerations are not taken into account, apart from the fact that NeTEx should not limit the implementation of different organizational structures that are different from this example.

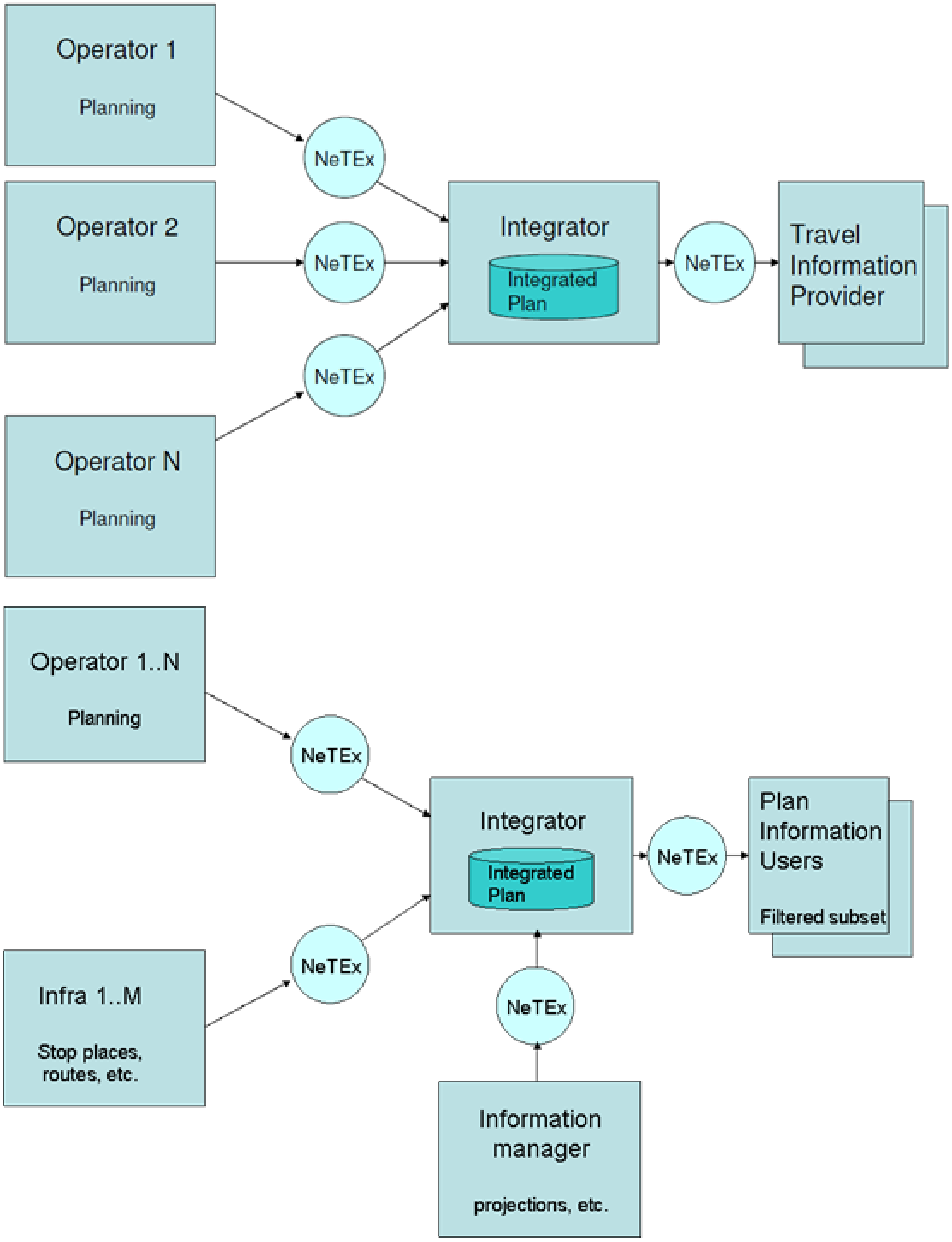


Figure 4: Illustration for use case *Distributed assignment of responsibility*.

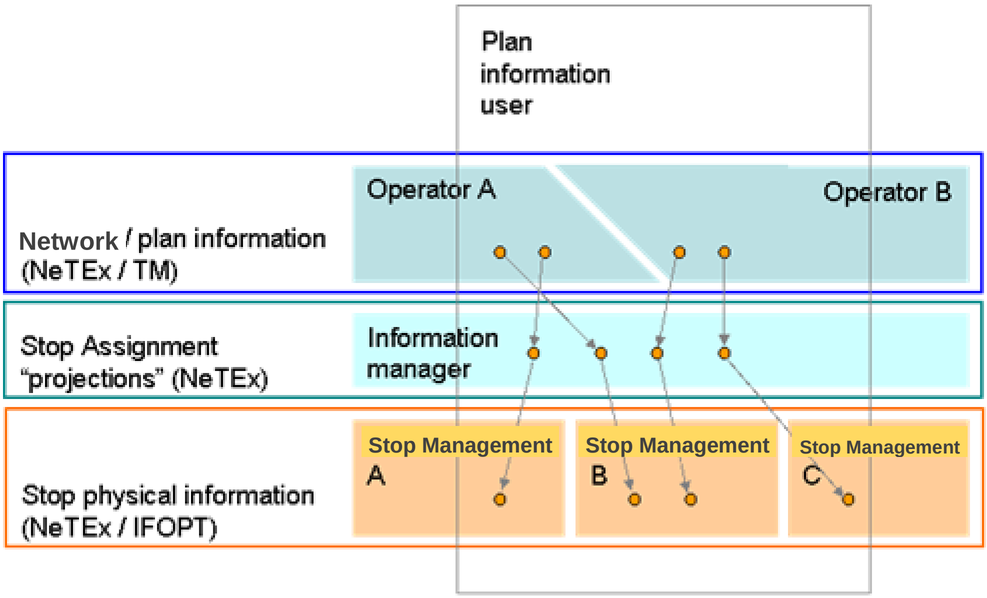


Figure 5: Illustration for use case *Distributed assignment of responsibility*, showing the referencing of layers and the use of filtering

Result of discussion on identifiers and data integration:

The NeTEx data model should allow for multiple identifiers per data object, depending on data system.

Use Case: PASSENGER-001-020 (#31)	
Name	Referencing locations to schematic maps
Source	NeTEx
Description	Schematic maps help passengers in finding locations (e.g. stations on network maps) or in choosing the best walking direction (e.g. finding a way through a complex interchange station to the departure point of the connecting service).
NeTEx contribution	NeTEx provides services patterns for lines and a model for stops and interchanges (IFOPT). It also provides unique object identifiers that allow mapping of object references across system boundaries.
Main actors	Passenger information systems
Main objects	Service patterns, Stops, Stop areas, Stop points, interchange paths, Object reference mechanisms.

Use Case: PASSENGER-001-021 (#32)	
Name	Linking multimedia content to public transport objects
Source	NeTEx
Description	For many public transport objects multimedia content is available. Examples of multimedia content are: area maps for stations, announcements at stations and in vehicles, content available on the internet, etc. To give passenger information systems access to this content whenever it is meaningful for passengers, the multimedia content shall be linked to public transport objects. This is a refinement of the general use case *Provision of passenger information on the planned service* (PASSENGER-001-001).
NeTEx contribution	NeTEx provides references and descriptions on how to access the content. It also provides unique object identifiers that allow mapping between content and public transport objects.
Main actors	Passenger information systems
Main objects	Media types and references, Service patterns, Stops, Stop points, Object reference mechanisms.

5.3.2.2.2 PASSENGER-002: Provide passenger information on the actual service

Use Case: PASSENGER-002-001 (#33)	
Name	Provision of passenger information on actual service
Source	TRANSMODEL
Description	Give passive and on-request (or on-subscription) information on the actual service on the actual (traffic) conditions and different events related to the use of PT. Provide passive information on the actual service Deliver on request information on the actual service. Exchange and record additional information. Give additional information. Passive information here means information that is published without any request or action taken by the user/passenger.
NeTEx contribution	Real-time information is out of the scope of NeTEx, but NeTEx will provide reference data (network topology) to real-time information system, based on SIRI requirements.
Main actors	Any SIRI producer/consumer
Main objects	Stop points (stop places, as SIRI Monitoring Points), Lines, vehicle journeys, routes, interchange and connections, equipments, PI monitoring displays (see SIRI Part 1: Context and framework - 3.3)

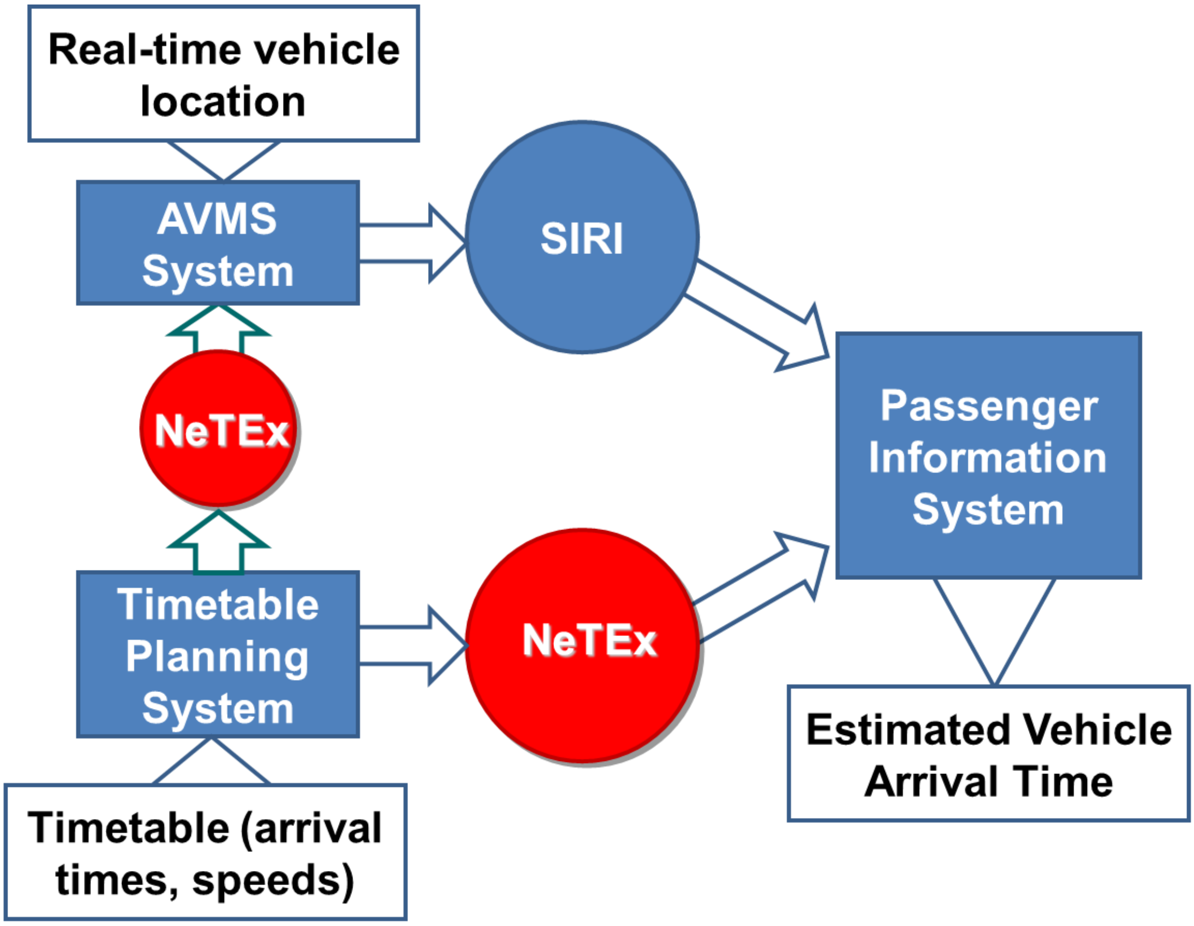


Figure 6 – Illustration for use case *Provision of passenger information on the actual service*

5.3.2.3 Provide information to other service providers

Use Case: OTHER-001-001(#34)	
Name	Showing stops on a map.
Source	IFOPT : GIS#01
Description	Stop Places and Stop Points are significant features on maps. Typically geocoded stop point data will be distributed as bulk data files to the map suppliers who will transform it into the representation used in their datasets.
NeTEx contribution	NeTEx provides means to bulk exchange stop places and stop points and their geographical position.
Main actors	Mapping service providers, authorities, operators.
Main objects	Stop places, stop points, geographical references, links.

Use Case: OTHER-001-002 (#35)	
Name	Provision of necessary information for geographic information or an Inspire dataset production.
Source	NeTEx
Description	http://inspire.jrc.ec.europa.eu/

	Inspire directive intends to trigger the creation of a European spatial information infrastructure that delivers to the users integrated spatial information services. These services should allow the users to identify and access spatial or geographical information from a wide range of sources, from the local level to the global level, in an inter-operable way for a variety of uses. The target users of INSPIRE include policy-makers, planners and managers at European, national and local level and the citizens and their organisations. Possible services are the visualisation of information layers, overlay of information from different sources, spatial and temporal analysis, etc. Transport networks are in the scope of Inspire.
NeTEx contribution	Provide all information required by Inspire NOTE NeTEx is focused on public transport network topology, timing information and fares, not on infrastructure. NeTEx only provides necessary means to be connected to an infrastructure (rails, roads, etc.). Therefore NeTEx will not be the appropriate source of information for INSPIRE required infrastructure.
Main actors	Authorities, operators
Main objects	Physical objects of the transport network

5.3.2.4 Provide information to AVMS and other operational units

- VEHICLE-001: Schedule vehicle blocks (Transmodel)
- VEHICLE-002: Perform and control the driving process (Transmodel)
- VEHICLE-003: Manage the vehicles (Transmodel)

5.3.2.4.1 VEHICLE-001: Schedule vehicle blocks

Use Case: VEHICLE-001-001 (#36)	
Name	Plan vehicle demand for timetable
Type	Content use case
Source	TRANSMODEL
— Description	After a service has been defined in terms of route run and frequency of operation, the next task in the scheduling process is to join the individual trips to vehicle blocks. One vehicle block describes the operation of a logical vehicle from depot pull out to depot pull in. The objective of the vehicle scheduling process is to cover the planned journeys in a cost effective manner. The process takes into consideration Required vehicle type per journey Garage location and capacity (per vehicle type) Dead run distances Duty scheduling considerations, e.g. relief points, layover times.
NeTEx contribution	NeTEx provides information about planned journeys, vehicle types, dead runs, vehicle blocks, relief points
Main actors	Timetable planning systems, tactical planning systems, AVMS, duty scheduling systems
Main objects	Journeys, dead run patterns, vehicle types, garages, relief points

5.3.2.4.2 VEHICLE-002: Perform and control the driving process

Use Case: VEHICLE-002-001 (#37)	
Name	Exchange between scheduling system and AVMS
Type	Content use case
Source	NeTEx Provides data for TRANSMODEL
Description	Public transport operators run control centres with dedicated staff (Control centre agents) and equipped by AVMS to monitor and control the running of the vehicles. The main purpose of a control centre is to provide a reliable and sufficient transport service – even in case of incidents and disruptions. For their operation control centres need up-to-date timetables and network descriptions including stop point information, connection links and connections that are to be protected. In case of network disruptions (e.g. because of flooded streets, a tree on the railways, etc.) control centres need to have replacement routes and timetables.
NeTEx contribution	NeTEx provides information about stop points, connection links, routes, journey patterns, timetables.
Main actors	Timetable planning systems, tactical planning systems, AVMS
Main objects	Journey patterns, Timing patterns, Connection links, timing points, activation points,

Use Case: VEHICLE-002-002 (#38)	
Name	Provision of reference data to manage incidents
Source	TRANSMODEL Provides reference data for IFOPT OPS#02 Provides reference data for IFOPT UTMC#02
Description	Support all actions that are necessary to maintain the right number of vehicles available to "transportation" and record and process all data relevant to accidents and damages. Tagging of incidents with the objects that are affected Collect incident alerts and data Establish diagnosis and action to be undertaken See also SIRI SX use cases
NeTEx contribution	Incident management is out of scope of NeTEx, but NeTEx will provide reference data to for incident management, based on SIRI SX requirements. NeTEx provides all SIRI SX reference Data
Main actors	Any SIRI SX actor
Main objects	Stop place, lines, route, journey patterns, journeys, interchange links (see SIRI SX 8.3)

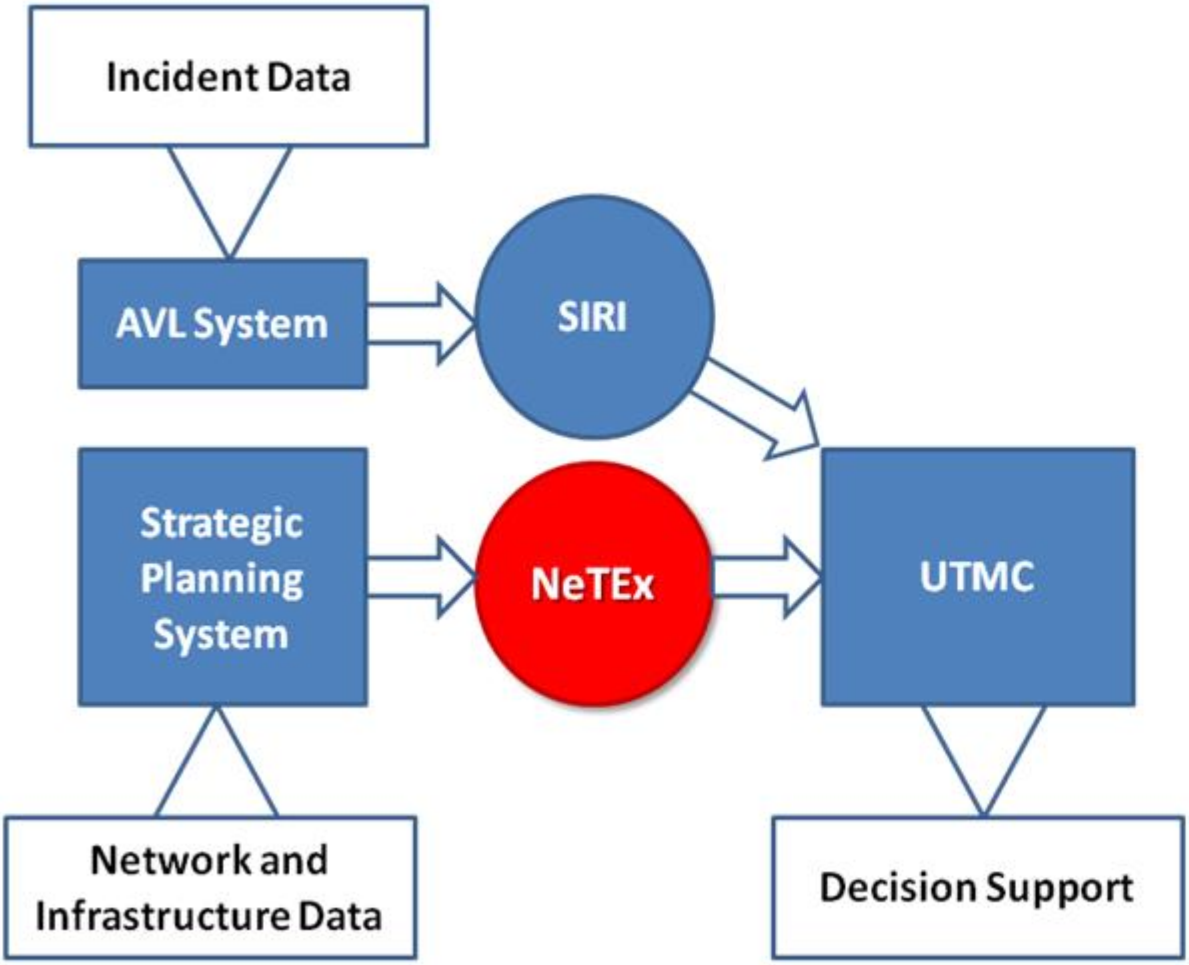


Figure 7 – Illustration for use case *Provision of reference data to manage incidents*

Use Case: VEHICLE-002-003 (#39)	
Name	Managing the connections between stop points at an interchange
Source	IFOPT : RT#04
Description	A particular concern of AVL systems exchanging information about connecting services is to relate the real-time connection information with the relative physical locations of stops in an interchange so as to be able to allow for accurate transfer times. This is especially important for large interchanges. Where there may be a big difference between near and far transfers.
NeTEx contribution	NeTEx provides means to exchange stop points, stop places, their positions and connection paths.
Main actors	Authorities, AVL systems
Main objects	Stop points, stop places, connection paths

Use Case: VEHICLE-002-004 (#40)	
Name	Planning transfer margins for guaranteed connections between services
Type	Content use case
Source	IFOPT : SCHED#05

Description	The Scheduler may give the connection link and interchange record, specific properties to support the management of guaranteed connections. These will be used subsequently by AVLS systems.
NeTEx contribution	NeTEx provides attributes (on connection links, stop places and lines) needed to elaborate a sufficient margin.
Main actors	Authorities and operators
Main objects	Detailed attributes of connection links and path links (distance, walk times, etc.). This can also be an attribute of an entire stop place (easy/difficult to cross). May also be an attribute of lines (interchange of journey patterns).

Use Case: VEHICLE-002-005 (#41)	
Name	Identifying the monitoring points exchanged between different AVL systems
Source	IFOPT : RT#02
Description	Different AVL systems exchanging information about buses roaming between their respective coverage areas will want to exchange information about arrival and departures at monitoring points and predictions for vehicle progress in order to include roaming buses in stop displays. To do this they need a mutually agreed set of monitoring points that are either the same as, or can be associated with, the known stop places and stop place components.
NeTEx contribution	NeTEx provides means to exchange stop points, stop places, their positions and labels.
Main actors	AVL systems
Main objects	Stop places, stop points, connection links.

Use Case: VEHICLE-002-006 (#42)	
Name	Identifying the relationship of the monitoring points to other points in an interchange
Source	IFOPT : RT#03
Description	In order to provide effective real-time information to the public, AVL systems shall understand the relationship between the monitoring points in an interchange and the physical layout of the interchange, including the actual quays and boarding positions
NeTEx contribution	NeTEx provides means to exchange stop points, stop places, their positions and connection paths.
Main actors	Authorities, AVL systems
Main objects	Stop points, stop places, connection paths

Use Case: VEHICLE-002-007 (#43)	
Name	Managing the traffic for an area including through a particular stop or interchange
Type	Content use case
Source	IFOPT : UTMC#01 *(part of..)*

Description	Urban Traffic Management Control systems seek to optimise the flow of traffic over a road network, often giving precedence to public transport vehicles which provide a more efficient method of transporting large numbers of users, for example, giving buses priority at traffic lights.
NeTEx contribution	NeTEx provides means to deliver stop points and their positions, as well as the modes of transport that serve those stop points. Additionally it can describe the paths of the PT lines between two subsequent calls at stop points.
Main actors	Authorities, operators, UTMC
Main objects	Relation between the infrastructure (road, rail, etc.) and the PT Network

Use Case: VEHICLE-002-008 (#44)	
Name	Management of statistical results
Source	TRANSMODEL
Description	Manage storage and administration of the data collected during the operational processes in order to provide relevant data for the other functions (e.g. "strategic planning") and issue periodic reports on the operations for various levels of management. Manage operational database (operations, fare use, equipment and personnel statistics) Manage connection with external databases (traffic, urban planning, law) Issue regular reports (daily, weekly, etc.) for management control (board, depots, etc.)
NeTEx contribution	NeTEx only provides the reference data None of NeTEx parts 1,2,3 manage PT service quality information. This may be the scope of a later part (4 or more).
Main actors	Authorities and operators
Main objects	No specific object, but the entire PT service offer is supposed to be carried by NeTEx in order to be compared to the recorded information.

5.3.2.4.3 VEHICLE-003: Manage the vehicle

5.3.2.5 Provide information about maintenance work and temporary services, restrictions s

Use Case: VEHICLE-003-001 (#45)	
Name	Management of vehicles
Source	TRANSMODEL
Description	Manage the use of vehicles according to the planned vehicle schedules."Equipment management" makes available the required number and type of vehicles according to the planning. "Transportation" is responsible for the optimal use of the vehicles. — Assign vehicles to blocks — Monitor the availability (status) of the vehicles — Control depot (vehicle location in the depot, entry and exit) — Prepare the vehicles for the next block (tank, read status, clean etc)
NeTEx contribution	NeTEx provides scheduled operational timetables together with their required facilities, as well as vehicle blocks consisting of productive and non-productive vehicle journeys.
Main actors	Operators
Main objects	Scheduled Timetable, vehicle block, vehicle equipment.

- MAINT-001: Plan maintenance work (Transmodel)

- VALIDITY-001: Provide support for defining object validities

5.3.2.5.1 MAINT-001: Plan maintenance work

Use Case: MAINT-001-001 (#46)	
Name	Planning of maintenance work
Source	TRANSMODEL
Description	Ensure that maintenance work on vehicles and infrastructure is spread evenly over the available capacity (personnel and tools). The capacity has to be optimised, taking into account time tables and vehicle schedules. Define all work that can be planned in advance Determine working steps Chain working steps to blocks Combine blocks to production sheets Define mean time to repair (MTTR's) Set up production schedules Plan capacity Forecast material consumption Estimate costs
NeTEx contribution	The organisation of maintenance work is mainly out of scope of NeTEx. Only the consequences and impact of maintenance work is within the scope of NeTEx. It may result in subsequent network versions. NeTEx provides means to exchange a temporary modified part of network (moved stop point, modified line, etc.)
Main actors	Operators, passenger information system, scheduling systems, AVLS
Main objects	Any NeTEx object with validity conditions or within network versions.

NOTE From a scheduling perspective, regular and special timetables will be treated the same way. However, there are impacts on passenger information. A special timetable is a temporary adaptation to the regular time table. This temporary adaption is to be integrated into the regular timetable for instance to allow for consistent trip planning advices.

5.3.2.5.2 VALIDITY-001: Provide support for defining object validities

Use Case: VALIDITY-001-001 (#47)	
Name	Temporary change in stop availability
Source	TRANSMODEL : GEN#02
Description	Systems shall allow for the temporary change of stop availability or accessibility, through the addition or closing of stops or the reassigning of scheduled stop points from one physical location to another the moving of stops, or by alterations to accessibility. Typically such changes are planned in advance and can be distributed with a validity condition along with other updates to the stop data.

NeTEx contribution	NeTEx provides enough support for temporary changes to stops that current and up-to-date timetables can be exchanged to reflect the correct situation. Support for advising passengers about what the impact to public transport will be and how they could avoid potential disruptions is likely to be applied in the following two typical situations: — During preparation of a trip, e.g. by using a trip planner: — NeTEx objects can optionally refer to a SIRI-SX structure as part of the NeTEx data delivery to provide situation related cause / effect information. During travelling however, the provision of this cause / effect information is the task of the operational SIRI interfaces.
Main actors	Authorities, operators, passenger information systems
Main objects	Validity information on stop places and stop point attributes

Use Case: VALIDITY-001-002 (#48)	
Name	Temporary timetable and service changes
Source	NeTEx
Description	Maintenance work in roads, bridges etc. can force planning departments to temporarily change timetables. The changes can have different effects: moving of stops, closed stops, detours, rail replacement services etc. Typically such changes are planned in advance and can be distributed with a validity condition along with other updates.
NeTEx contribution	NeTEx provides enough support for planned, temporary changes to stops, timetables and services so that current and up-to-date timetables and service information can be exchanged to reflect the correct situation. New versions of existing objects can be exchanged that over-ride the existing objects during the specified version validity period. For exchanging information on temporary changes a frame level shall be found that suits the needs and capabilities of both: the sender and receiver. Exchanging single passing times that have changed may prove to be too fine-grained, instead it may be more practical to deliver new version of objects on a higher level: on e.g. Line, Route, Service Pattern, Vehicle Journey level. The chosen level will be system and context dependent (and therefore not restricted by NeTEx). Support for advising passengers about what the impact to public transport will be and how they could avoid potential disruptions is likely to be applied in the following two typical situations: — During preparation of a trip, e.g. by using a trip planner: — Different versions of timetables (or, more generally, NeTEx objects) can optionally refer to a SIRI-SX structure as part of the NeTEx data delivery to provide situation related cause / effect information. — During travelling however, the provision of this cause / effect information is the task of the operational SIRI interfaces.

Main actors	Authorities, operators, passenger information systems
Main objects	Validity information on service patterns, journey patterns, timetables and services

Example, temporary planned changes as seen from a version management perspective:

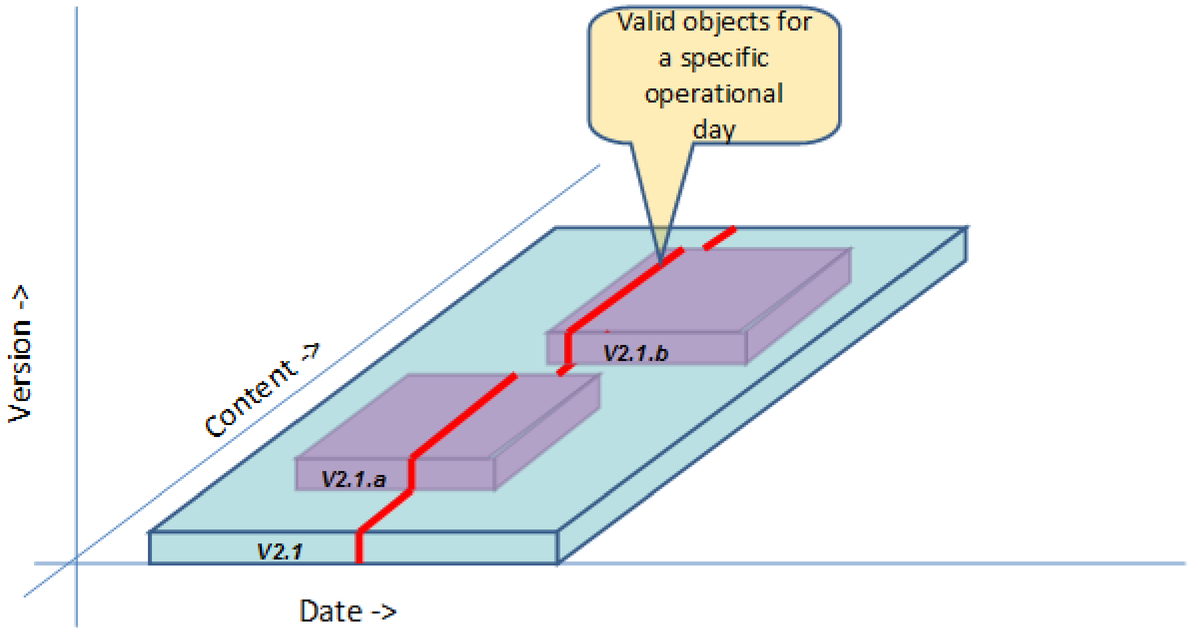


Figure 8 – Illustration for use case *Temporary timetable and service changes*

6 Generic Physical Model and XSD mapping rules

6.1 Introduction

Public Transport Timetable and Fare information systems are a large and complex subject. One of the concerns of NeTEx is to reduce the complexity of understanding the extensive design models and their implementations. To do this NeTEx uses a systematic and uniform approach to design and documentation wherever possible. Understanding the common underlying conventions, techniques and design patterns can simplify the task of comprehending the many NeTEx model elements. Although NeTEx has several hundred classes and packages, the actual number of concepts needed to be understood to make effective use of it is much smaller.

This section describes the methods and design patterns used to describe the NeTEx models and the transforms used to map between them.

6.2 Model Driven Design

NeTEx uses a Model Driven approach, that is, the design is first described as a high level conceptual model that tries to represent the problem domain as entities and relationship, as identified by the uses cases and functional scope. This model is then elaborated to create a design for a physical implementation that can subsequently be transformed into an implementation, either or automatically or semi-automatically.

The use of models allows designs to be reviewed and validated by interested parties and to be fully documented with narrative text that describes the intention of the design. Modelling in particular helps identify common abstractions and generalisations that simplify the implementation. It also allows dependencies between components to be understood so that the system can be modularised in a way that minimises coupling and optimises flexibility. This in turns makes it easier for implementers to select just those components needed for a given purpose, while ensuring future extensibility.

Models may exist at different levels of abstraction. NeTEx distinguishes between three levels;

- **Conceptual Model** – A high level implementation neutral representation.
- **Physical Model –** A design for implementing the Conceptual model using a specific technology.
- **Implementation Model** – An implementation of the model in a specific language that supports declarative modelling constructs – The NeTEx standard uses W3C XML schemas for this purpose, but in principle other languages could be chosen in future.

6.3 Models – levels of abstraction.

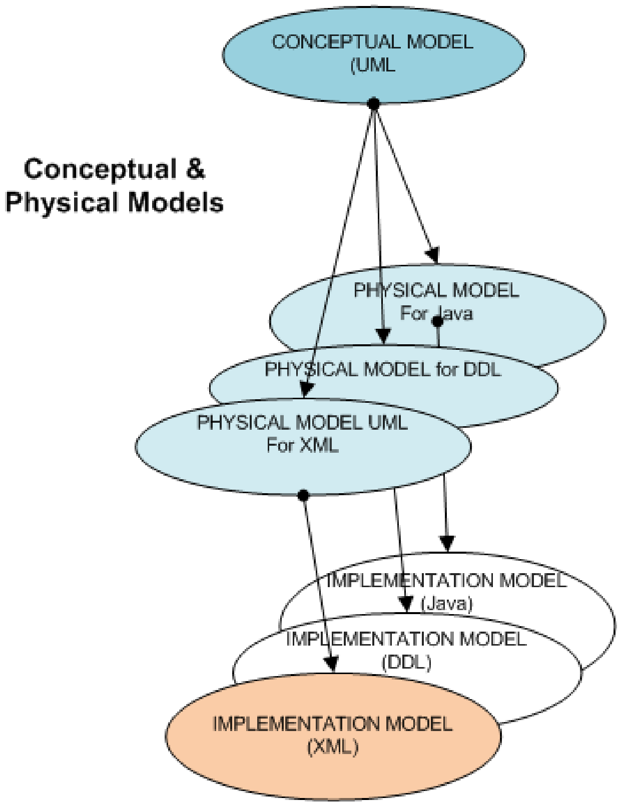


Figure 9 – NeTEx modelling layers

Elaborating slightly on the above:

— A *conceptual model* describes the elements and relationships of a model independently of any specific implementation technology. It may be unconcerned about technical or implementation details such as the exact scope of identifiers, or the implementations of attributes. It can be used to understand and relate different implementations using many different technologies and physical models.

— A *physical model* maps a conceptual model into a more detailed model that assumes a specific implementation technology, subject to particular constraints of that technology. For example, different object programming technologies (XML, Java, C#, CLOS, Smalltalk, Scala) support inheritance in different ways, whilst relational data base technologies do not support inheritance or strict encapsulation etc., but may support referential integrity. In addition a physical model will add specific details about typing, identifier scope, etc. It will indicate how the mechanisms of an implementation language will be used to create a representation that is efficient to use and to maintain. In general a physical model will have more limited semantics than a conceptual model because of the constraints of the chosen implementation language and of the need to simplify the use of a rich conceptual model for any practical implementation. It must also make specific choices on common aspects of object behaviour like name spaces, versioning and data rights. A physical model is still however not usually an executable or completely finalised representation, but rather a translation tool that shows how to go from a very high level technology-independent conceptual model to a concrete implementation in a specific language. To do this in a single step would be too difficult.

— An *implementation model* represents a physical model in the actual language constructs and types of an implementation language, such as a DDL for a database, Java Classes for an application, or an XML schema for an exchange format. It is directly executable using tools or programs.

6.4 Open Implementation and technology use

Here we elaborate on the relevance of design models and schemas for implementing data.

Data standards are primarily concerned with models of data exchange between systems so that multiple parties can integrate data from different sources. As long as tools and products can accept data in a common format, software developers should be free to implement their actual data bases and choose the internal representations used in their tools in different ways to suit their product objectives and favoured technologies.

It is important that standards allow implementation economically using mainstream software tools and technologies and with a high degree of automation. NeTEx uses an XML schema as the primary software artefact for achieving this, as shown in Figure 10, which illustrates the derivation of an XML exchange implementation from the conceptual model. Thus;

a) A conceptual model is formulated in UML based on Transmodel.

b) The conceptual model UML is used to specify a more detailed Physical model, also in UML (*XML-1* in Figure 10*).*

c) The Physical UML Model is used to specify an XML model (*XML-2* in Figure 10*)* which can be used to automatically validate documents exchanging data conforming to the model using widely available tools (*XML-3* in Figure 10*).*

d) The XML schema can also be used to automatically derive software language bindings (for example, in Java JAXB or equivalent) for the adaptor programs (*XML-4 a & b* in Figure 10) which input or output XML from a database in an implementation.

e) Implementations will support an XML import/export tool which will translate data from an implementation database into the XML format for exchange with another system. (*ImPex a & b* in Figure 10*).*

f) Each data model will be capable of representing the entities of the conceptual model as mapped to relational database design in UML or other notation (*DB-1* in Figure 10). And then translated into an actual Data Definition Language such as SQL (*DB-2* in Figure 10) However, the database schema will typically be a proprietary design whose details do not need to be exposed.

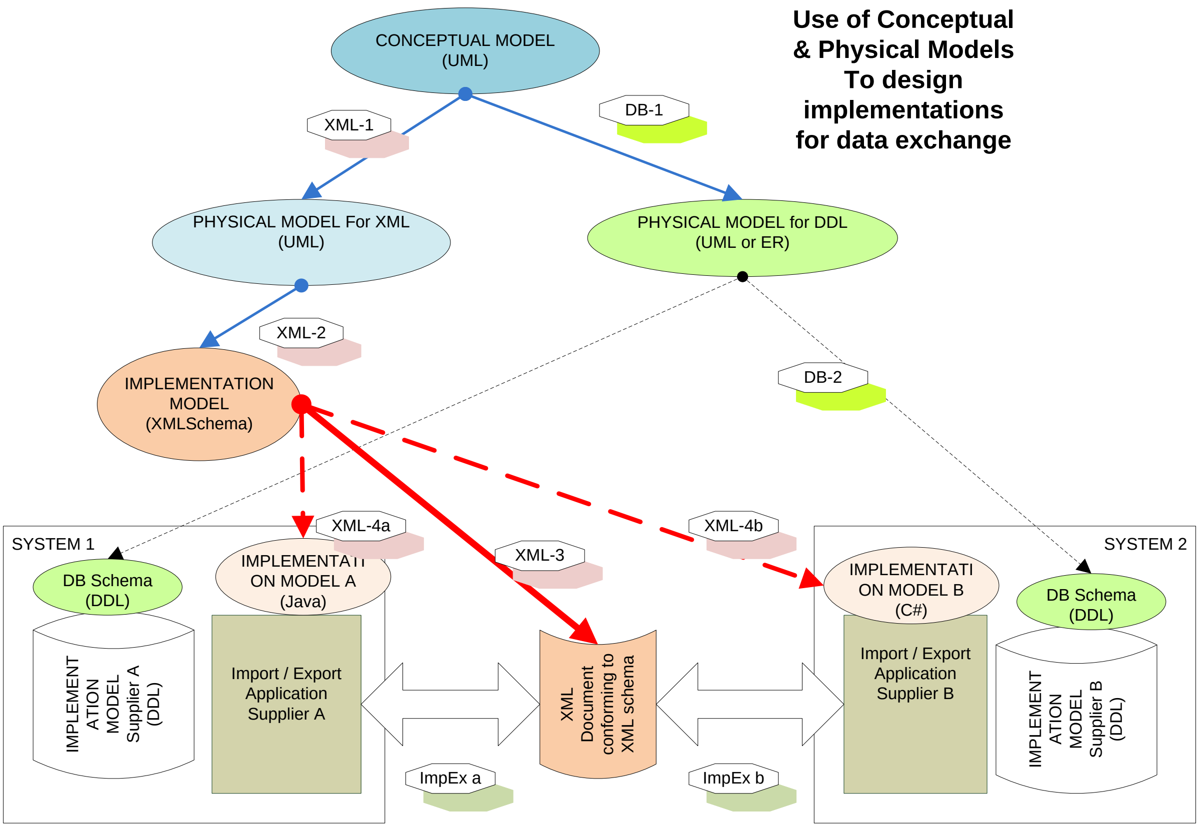


Figure 10 – Use of Standards in Implementations

6.5 Models versus Protocols

Another of the guiding principles in NeTEx is to separate out the concerns of the *data model* (and the data formats used to serialize the model) from the *data protocol* (the communications technology and sequence of messages needed to transport data between two systems. The same data model (serialised with any of a number of different technologies such as say XML, JSON, proto-buffers, etc.) may be used with a variety of different protocols (for example FTP, http, SMTP, SIRI, etc.) in different interaction patterns to support different levels of concurrency (for example occasional batch transfer, continuous dynamic update, publish subscribe, etc.).

Consideration of format and protocol is not completely separate in that choice of granularity – how big are the data aggregations exchanged at a given time - is dependent on the aggregation mechanisms available in the representation model.

This chapter is mainly concerned with the data model. The use of data protocols is discussed separately in chapters 6.4 and 6.5.

6.6 Modularisation

Large conceptual models such as NeTEx (which has several hundred entities) need to be modularised into smaller submodels so that they are manageable, both to understand and to allow the separate evolution of unrelated concepts. NeTEx encapsulates model elements in small packages of just a few related elements, each with their own self-contained diagrams and documentation.

The NeTEx models are modularised consistently so that the same entities are found in a corresponding package in the Conceptual, Physical models and the XML Schema.

Thus for example there is an Infrastructure model, a Site model etc.

As far as possible the coupling between models is minimised so that submodels only reference the other models they require, and a circular dependencies are avoided. There is a core set of fundamental framework elements that are referenced by all other packages, dependent packages are built out with a linear dependency graph.

6.7 Summary of Modelling Approach

NeTEx uses a systematic process to map the Transmodel conceptual model into a concrete XML schema in three steps:

1) The Transmodel **conceptual** model is modularised into core framework of **generic** element packages and a number of **functional** modules that share any necessary common generic packages, but are not promiscuously interdependent. This modularisation further refines the original loose functional partition of Transmodel functional areas into a strictly modularised organisation with a linear dependency graph between models so that components required for a given purpose can be used independently of other parts of Transmodel. The modularised "NeTEx conceptual model" is also updated to add some new NeTEx concepts and to integrate Transmodel concepts. The model is documented in UML in electronic form that can be exchanged and versioned.

   For each package of the conceptual model, a corresponding set of **physical** model packages, also documented in UML, is developed. This "NeTEx physical model" adds in detailed property attributes for the conceptual model entities based on the requirements of current European national standards and reference exchange protocols such as VDV452 (de/ch/at), Trident / NEPTUNE (fr), BISON (nl) and NaPTAN/TransXChange (uk).

   The NeTEx physical model is targeted towards XML, and assumes XML as a programming model, for example it assumes a single inheritance model and XML types & enumerations. In particular, the NeTEx physical model is concerned with an efficient serialisation of complex objects for data exchange in XML (this may be contrasted with, say, a physical model optimized for database storage or for in memory computing). As such, it is concerned to identify elements and groupings of elements that are actually exchanged in practice (such as a network or a whole timetable) and to optimise their representation for reuse in many different use cases. The aim is to simplify and have a smaller number of "first class elements" that represent the semantically significant entities, and to organise these into a small number of coherent “**frames**” of objects that are usually exchanged together. Other secondary entities are then only exchanged in the context of these frames and their first class objects. For example, a call at a stop in a journey is only exchanged as part of a whole journey, not as an independent element (though it is in fact a design entity in its own right).

   Specifically the physical model:

   i) Names elements, attributes and all types and assigns them each to a package (typically a refinement of the conceptual model packages);

   ii) Identifies identifiers and their scopes;

   iii) Indicates how relationships are to be serialized;

   iv) Indicates a composition of objects from reusable elements using inheritance hierarchies and/or composition. For serialisation to XML, a decision about the implementation of relationships is an especially important consideration.

2) The NeTEx Physical UML model is used to create an XML schema. For each Physical model entity and attribute, corresponding XML elements are created, modularised within a similar package structure to that of the physical model, but further broken down (for example one schema file is used to declare the base types, another the actual reusable elements). Where possible, semantic constraints are enforced in the XML, so that the built-in capabilities of standard XML validators and parser can be harnessed. For example, through XML language mechanism such as types and enumerations, uniqueness and referential integrity constraints. However in order to have a single set of XML elements that can be used in a wide set of applications, cardinality constraints are generally more relaxed in the XML schema than

in the conceptual model; for example a name attribute might be mandatory in the conceptual model, but optional in the schema. The XML schema also includes container elements (Publication) to group all the model elements.

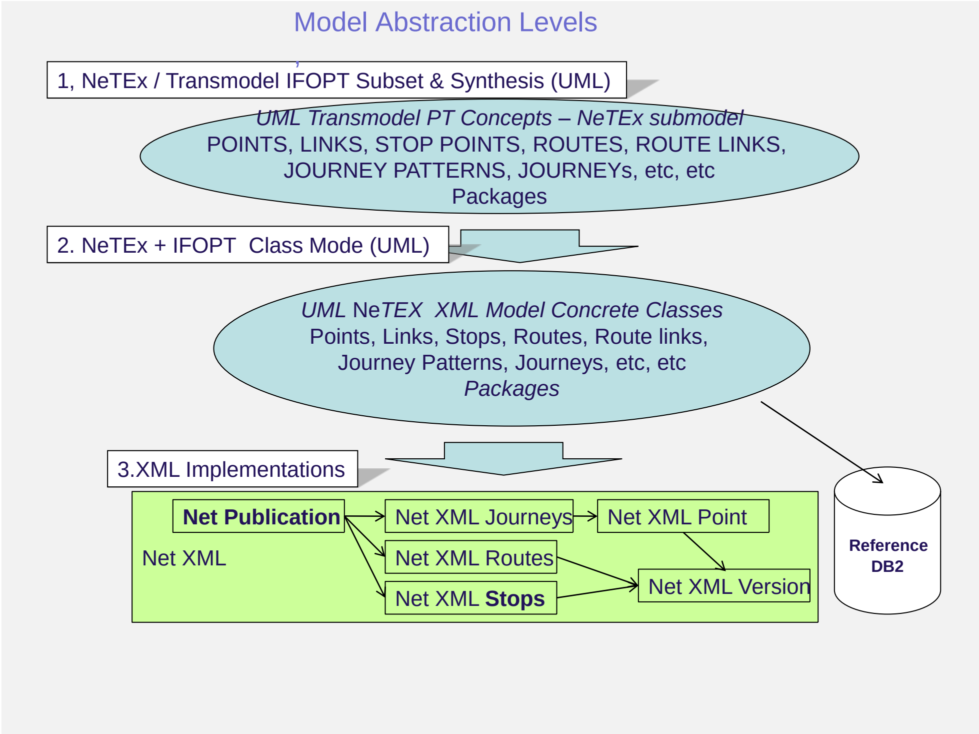


Figure 11 – NeTEx modelling approach

6.7.1 Use of packages in NeTEx models

Each level of modelling adds detail useful for implementation. Typically for each Conceptual MODEL package there are at least two Physical UML model packages, one for the type declarations used in the model, one for the model itself.

Thus for example for the packages relating to Timing Patterns in the three different models,

— The *Timing Pattern MODEL* package in the Part1 NeTEx Conceptual model contains three entities: TIMING POINT, TIMING LINK and TIMING PATTERN (These are each specialisations of entities in the generic framework). Versioning and other implementation considerations are only described generically in other framework packages (Generic Version MODEL, Generic Responsibility MODEL, etc.), and there is no visible distinction between an entity and versions of that entity.

— The *TimingPatternPackage* in the Part1 NeTEx physical model has two subpackages.

1) The *TimingPatternModel* package contains three entities ***TimingPoint***, ***TimingLink*** and ***TimingPattern*** that represent physical versions of the timing pattern classes for exchange in the context of a version frame.

2) The *TimingPatternSupport* package additionally contains definitions for a ***TimingPointIdType***, ***TimingLinkIdType*** and ***TimingPatternIdType*** as well as elements to provide versioned references to them (***TimingPointRef***, ***TimingLinkRef*** and ***TimingPatternRef***). It also contains a ***TimingPointStatus*** enumeration.

— The Part1 NeTEx XML schema has two subschema files containing schema definitions for Timing Patterns.

1) The *netex_timingPattern_support.xsd* subschema contains corresponding XML type definitions for the identifier classes specified in the Physical Model *TimingPatternSupport* of the Part1 NeTEx XML schema*.*

2) *The* netex_timingPattern_version.xsd subschema contains XML elements and complex type definitions for the ***TimingPoint***, ***TimingLink*** and ***TimingPattern***.

6.8 Model transforms and Traceability

NeTEx uses a small number of design patterns to transform the conceptual model to the physical model and the physical model to XML. An understanding of these patterns makes an understanding of NeTEx simpler to grasp. It also makes it straightforward to trace the design through to the implementation.

6.8.1 Conceptual Model UML Package

The Conceptual model UML package contains the Entity classes for a given functional module. Its name has the form XX XX MODEL, where XX XX is the domain area. For example, Route MODEL, Timing Pattern MODEL, Call MODEL

The conceptual model entities are always in upper case with a space between any compound words, for example ROUTE, VEHICLE JOURNEY, TIMING POINT.

6.8.2 Physical Model UML Container Packages and Mapping from Conceptual model

For each Conceptual MODEL there is a corresponding Physical model UML Container package containing one or more submodels. The Physical model Container package name has the form ***XxXxPackage***, where XxXx is the domain area. For example, ***RoutePackage, TimingPatternPackage***.

The container contains two or three different types of subpackages:

a) Physical Model UML Package – ***XxxxModelPackage.***

b) Physical Support UML Package – ***XxxxSupportPackage.***

c) Views UML Package – ***XxXxView.***

These are described further below.

6.8.2.1 Physical Model UML Package - *XxxxModelPackage*

The Physical model ‘Model’ UML subpackage contains all the main physical classes for the functional module corresponding to those found in the Conceptual MODEL. The ‘Model’ package name has the form ***XxXxModel***, where XxXx is the domain area. For example, ***RouteModel***. ***TimingPatternModel***. Complex Models may be broken down into multiple models within a physical package – for example the ***Equipment*** models.

For each entity (i.e. UML Class) in the Conceptual MODEL package, a corresponding class is defined in the corresponding physical model package. For example TIMING POINT becomes ***TimingPoint***.

— If the class is a first class object it will be a descendant of ***DataManagedObject,*** which defines common version and responsibility attribute***s***.

— If the class is a child entity of it will be a descendant of ***VersionedChild,*** which defines common version attribute, but assumes responsibility attributes will be the same as the parent ***DataManagedObject.*** For example, ***PointInJourneyPattern*** is contained within ***JourneyPattern.***

Each element It will have an ***id*** field of type XxXxIdType, as defined in the Support Package.

6.8.2.2 Physical Support UML Package – *XxxxSupportPackage*

The Physical model 'Support subpackage contains ancillary types and classes needed to implement the physical classes of the functional module. The Model package name has the form ***XxxxSupport***, where Xxxx is the domain area. For example, ***RouteSupport***, ***TimingPatternSupport***.

— For each entity in the conceptual model, a class is defined in the physical model for its identifier. This will be a descendant of the ***ObjectIdType*** in the NeTEx entity framework.

— For each entity in the conceptual model, a class is defined for a versioned reference to its identifier with an attribute identifier class is for a reference to its identifier. This will be a descendant of the ***VersionOfObjectRef*** type in the NeTEx entity framework, which has attribute *version*. This reference is used to implement one to one relationships.

For simple data types that take a restricted set of values an enumeration class will be defined that declares these values.

6.8.2.3 Views UML Package – *XxXxView*

Some physical models also include an additional View Package. This contains definitions of views of some model objects for use in the XML in some contexts where derived information is needed. Views can include attributes derived though an entity's associations with other entities, or omit unneeded attributes. The Model package name has the form ***XxXxView***, where XxXx is the domain area, for example ***RouteSupportPackage***.***ServicePatternView***.

— Each view entity class is defined in the view model with an association with the underlying model object. The view will be a descendant of the ***DerivedView*** in the NeTEx physical framework.

6.8.3 XSD Model subschemas and Mapping from Physical model

For each Physical submodel there are corresponding W3C xsd subschemas containing the XML classes corresponding to the UML ones and adding in further implementation properties. There are usual two for each functional domain, one for any support classes and one for any model classes.

The names of the xsd subschemas have the form *netex_xxXx_version-v9.9.xsd*, and *netex_xxXx_support* - v9.9.xsd, where ***xxXx*** is the domain area, for example, *netex_route_version-v1.0.xsd, netex_route_support-v1.0.xsd* or *netex_timingPattern_version-v1.0.xsd.*

— For each class in the Physical UML Model representing an ENTITY, a corresponding element is defined in the XSD. For example ***Route, TimingPattern***.

— The XML element will have a corresponding XML complex type, whose name has the form *XxxxStructure,* which adds the attributes of the class, *F*or example *TimingLink_VersionStructure*, *Route_VersionStructure.* These attributes will be grouped with an XML Group whose name has the form ***XxXxGroup***, for example ***TimingLinkGroup*** contains the properties of a ***TimingLink*** such ***FromPoint***, ***ToPoint*** etc.

— For each class in the Physical Model representing to a data type or an enumeration, a corresponding type is defined in the XSD. For example ***RouteIdType****.*

6.8.4 Summary of Basic Mapping

The following Table summarises the basic mappings of Models and entities in NeTEx.

Table 5 — Basic mappings of Models and entities.

		Conceptual	Contents	Physical	XSD

Model	General Form	XX XX MODEL	*Container*	***XxXxPackage***	-
			Classes	***XxXxModel***	netex_xxXx_version-v9.9.xsd
			Data types	***XxXxSupport***	netex_xxXx_support-v9.9.xsd
			Views	***XxXxView***	netex_xxXx_version-v9.9.xsd
	Example	ROUTE MODEL	*Container*	***RoutePackage***	-
			Classes	***RouteModel***	netex_route_version-v9.9.xsd
			Data types	***RouteSupport***	netex_route_support-v9.9.xsd
			View	***RouteView***	netex_route_version-v9.9.xsd
Entity	General Form	XX XX	*Class*	***XxXx***	***XxXx***
			Structure	**--**	***XxXx_VersionedStructure***
			Group	**--**	***XxXx_Group***
			Identifier	***XxXxIdType***	***XxXxIdType***
			Ref	***XxXxRef***	***XxXxIdRef***
			RefStructure	--	***XxXxRefStructure***
			Enumeration	***XxXxEnum***	***XxXxEnumeration***
	Example	ROUTE POINT	*Class*	***RoutePoint***	***RoutePoint***
			id	***RoutePoint /id***	***RoutePoint /id***
			Structure	**--**	***RoutePoint_VersionedStructure***
			Group	**--**	***RoutePointGroup***
			Identifier	***RoutePointIdType***	***RoutePointIdType***
			Ref	***RoutePointRef***	***RoutePointdRef***
			Enumeration	***ViaTypeEnum***	***ViaTypeEnumeration***

6.9 Physical model to XSD schema mapping notes

In general the XML uses the “Garden of Eden” coding style, with explicit XML complex types being declared for each XML element, rather than deep nested sequences. Inheritance is used in the type structures to reflect the Physical Model, and substitution groups are provided on elements to help support extensibility. This increases reusability and maintainability but is often more verbose at a code level.

XML attributes are used mainly for metadata, not for the data attributes of UML model elements, which are implemented as child xml elements, for example ***Point / Name***, ***Point / Description*** etc. The main exception is that an actual XML attribute is used for the element identifier – this has the name ***id***.

Most data attributes are optional – this makes it possible to reuse a single schema for many different purposes – but means that the checking for attributes required in a particular context has to be done programmatically rather than by the XML validator.

A strong typing of identifiers is mostly used to help enforce correctness of reference.

NeTEx also uses specific complex types to represent each of the relationships in the model. This helps with traceability from the physical model to the XML schema.

- One-to-one relationships are implemented with the ***Ref*** mechanism indicated above. There is a data structure and element for referencing each element. For example, ***StopPointRef*** of type, *StopPointRefStructure.*

- One-to-many relationships are implemented with an explicit data structure for each relationship. For example, the "***StopPlace*** - quays **-** ***Quay***" relationship is implemented with an explicit complex structure *quays_relStructure*. See the NeTEx Framework Guide for further details (available on NeTEx web site).

The XML root schema (NeTEx_Publication.xsd) includes referential integrity constraints such that fully versioned references will be validated. See NeTEx Framework Guide for further details.

Explicit name spaces are used – chameleon name spaces are avoided.

Xsd groups are used extensively to organize elements into functionally related groups and document the schema. These are syntactic and do not appear in the resulting type structures of an XML binding.

6.10 Uniqueness of reference and Namespaces

NamespaceCodespaces (namespaces for code identifies - See NeTEx Framework Guide available on NeTEx web site) can be used to ensure that each object is given a globally unique identifier regardless of its data source. Identifiers are in most cases unique within their class. Thus for example, the TimingPoint with id of "*hde:123*" is different from the RoutePoint with an id of "*hde:123*", although both are specialisations of Point.

6.11 Handling of inheritance

Inheritance is used in the NeTEx models to increase usability and simplify the implementation. For example there are many specialisations of Point, such as **ActivationPoint, RoutePoint, TimingPoint** etc. with corresponding specialisation of Link such as **ActivationLink, RouteLink, TimingLink** etc. In most cases a single inheritance hierarchy suffices. This can be mapped readily from the UML model to the inheritance of an xsd type structure. To make clear the source of inherited attributes, the elements of each parent are grouped in a named xsd Group, thus for example the following summarise the Hierarchy for ***ParkingPoint.***

DataManagedObject

Point *Point_VersionedObjectStructure* *PointGroup*

TimingPoint *TimingPoint_VersionedObjectStructure* *TimingPointGroup*

ReliefPoint *ReliefPoint_VersionedObjectStructure* *ReliefPointGroup*

ParkingPoint *ParkingPoint_VersionedObjectStructure* *ParkingPointGroup*

In a few cases, multiple inheritance is used in the UML model. In order to map this into XML (which only supports a single inheritance model) groups are also used. One of the generalization relationships is selected as primary and implemented using the XML inheritance as described above. Additional inheritance is implemented using a Group of elements from the additional element included in the structure.

6.12 NeTEx Notation, presentation and naming conventions

NeTEx XML uses a consistent set of naming and coding conventions. These have been evolved from those used in the SIRI schemas and are similar to them with a number of refinements. For example NeTEx uses camel case for Element and attributes names.

Here we summarise the naming conventions used in NeTEx

6.12.1 Presentation of Element Names

NeTEx follows certain conventions for presentation of technical terms:

Transmodel / NeTEx conceptual model elements are shown in UPPER CASE, for example "LINE", "SCHEDULED STOP POINT", "ENTITY IN VERSION". Names may be pluralized for readability, for example, "SCHEDULED STOP POINTs", "ENTITies IN VERSION".

Concrete XML elements are shown ***in bold italic, fo***r example, ***Line***. Compound names are camel cased without a space and are never pluralized, e.g. "***ScheduledStopPoint",*** "instances of ***ScheduledStopPoint".***

Entity and class names are capitalised, as are properties of an element that are implemented as a XML child element, for example, "***ScheduledStopPoint*** / ***Name"***.

Properties that are implemented as XML attributes are lower cased, for example "***id"***, "***order*** ".

One-to-one relationships that are implemented as versioned references are shown as simple attribute names, for example ***ScheduledStopPointRef, ZoneRef.*** Sometimes these are qualified by a name to distinguish them for example ***ParentZoneRef, FromPointRef, ToPointRef***, etc.

One-to-many relationships that are implemented as collections are lower cased, for example, ***stopPoints***, ***vehicleJourneys.***

The XML elements used to implement – one-to-many relationships are shown in lower case and are always in the plural, for example "***StopPlace / quays / Quay***", or "***Timetable / journeys / ServiceJourney".***

6.12.2 Presentation of Data Type Names

Data types are shown in *italic*, Built-in SML types are shown in *lowerCamelcase*, and prefixed by the '*xsd'*; namespace, for example *xsd*:*dateTime,* xsd:*normalizedString, xsd:integer*. NeTEx defined Data types are shown in ***UpperCamelCase***, e.g. *LengthType, DirectionTypeEnumeration,* without a namepace prefix. Compound names are camel cased without a space, for example *ScheduledStopPointIdType*.

In the tables used to list the properties of each XML element

- Complex types are generally underlined, e.g. <u>*DataManagedObject*</u>; <u>*Line*</u>, <u>*Presentation*</u>. this is so as to highlight types for which detailed information can be found elsewherw*.* An exception to this are *VersionOfObjectRef* and its descendants, which are not highlighted*.*

- Simple data types are *italicised* but not underlined, e.g. *DistanceType, LenthType, EmailType, WeightType, StatusEnum*.

- Abstract Types are shown in brackets e.g*. (PlaceRef) (DataManagedObject)* a concrete subtype must be used intead.

- If a concrete type has substitutable subtypes, this may be indicated by a '+', e.g *TimingPatternRef+*, *JourneyPatternRef+*.

6.12.3 Naming conventions

In the XSD schema, element and attribute names are the same as the physical UML model entities, that is, they are ***UpperCamelCase*** and ***lowerCamelCase*** respectively. In addition the names of complex types end in '*Structure',* for example '*HeadwayIntervalStructure'*. NeTEx simple type names mostly end in '*Type',* for example *'LengthType'*.

Names of complex types used to implement a one to many relationships end in '_*relStructure'*, for example, '*stopPoints_relStructure'.*

Complex types used to implement a one to one relationship- end in '*RefStructure'*, e.g. '*StopPointRefStructure'.*

Data types used for identifiers end in '*IdType'*, for example '*RouteIdType'*, '*TimingPatternIdType'.* The use of 'Type' in names of elements and structure is otherwise mostly avoided. The main exception is the set elements and structures used to implement the ***TypeOfXxxx*** pattern.

XML enumerations of fixed values are used widely; their names always end in '*Enumeration'*, e.g. '*QuayTypeEnumeration'*. In the type colums of table 'Enumeration' is shortened to 'Enum' for brevity, e.g. '*QuayTypeEnum'*. They are indicated by a stereotype of 'enum'.

Enumerated values are mostly in lowerCamelCase and are shown in italic font, *e.g. 'railStation'.*

XSD schema Groups are used extensively as syntactic aids to organize schema elements. The name of a group always ends in '***Group'***, for example '***StopPointGroup'***. The use of 'Group' in names of elements and structure is otherwise mostly avoided. The main exception is the set elements and structures used to implement the ***GroupOfXxxx*** pattern.

XML Groups are indicated by a stereotype of 'XGRP' and a data type of ***xmlGroup***.

6.12.4 Presentation of UML Diagrams

Standard UML notation is used for structure diagrams. In the physical model a top down, right to left semantic is followed in most cases. The root objects appear at the top and child objects are shown below.

A number of different variants of UML class diagram are provided for each domain model; each variant makes a different selection of UML model elements in order to give different view of the underlyuing model. Generally the folloiwng are provided.

- Conceptual model: A **MODEL** diagram giving an overview of the domain ENTITIEs.
- Physical: **Intro**, **Model, Hierarchy** and **Support** diagrams are always supplied for each functional domain in the electronic version of the specification, but only the **Model** diagram is included in the specification document (though a few **Intro** digrams are include for overviews) . Other diagrams are only supplied if relevant.

Additionally UML package **Dependency** diagrams are provided at the beginning of each part to show the dependencies between packages and between submodels.

Table 6 — Diagram types.

	Subpackage Name	Diagram Type	Description
Conceptual	XX XX MODEL	**Class MODEL**	Shows entities and relationships
Physical Model	XxXxModel	**Intro**	Shows Physical model elements and relationships of all elements in package. May include elements from other packages to show context.

			Attributes are hidden. Should In most cases correspond one-to-one to with conceptual model diagram.
		Model	Shows Physical model classes with detailed attributes
		Concise	Shows Physical model classes omitting the attributes needed to implement relationships
		Classification	Shows tables of fixed values in context.
		Views	Shows any Physical Model View classes
		Hierarchy	Shows Inheritance hierarchy of just those classes in the model
SupportModel	XxXxSupport	**Support**	Shows data types used in model

6.12.5 Use of Stereotypes

Stereotypes are used on attributes to indicate their role in common design patterns.

Table 7 — Stereotypes.

Stereotype	Description
«PK»	Identifier of ENTITY. (usually an XML attribute «atr» with name ***id***)
«FK»	Reference to the identifier of another element (i.e. the Foreign key" serialising a specific relationship in the model)
«AK»	Alternative unique identifier of ENTITY.
«EV»	View («FK» + attributes): Reference to the identifier of another element (serialising a relationship with another entity), along with one or more attributes derived through the relationship which are useful in the current context.
«atr»	Implemented as an XML attribute.
«enum»	Reference to a classification or characterision property of an entity implemented as an XML enumeration
«cntd»	Element is embedded inline, variously either as (i) a single element or (ii) a set of elements or (ii) as a set of references.

6.12.6 Use of Colour

Colour is used systematically in NeTEx to help make the diagrams easier to understand. Elements from the same functional domain are mostly given the same colour in all diagrams across all the different levels of abstraction.

Generic framework elements are mostly shown in grey.

6.13 Mapping between models in NeTEx

6.13.1 Common Design Patterns in NeTEx

Several design patterns are commonly found in NeTEx and are supported by the Framework classes

a) *Versioned Entity:* A first class entity that has versioning and responsibility attributes.

b) *Entity - Type of Entity:* A classifier entity associated with a specific first class entity. For example PLACE / TYPE of PLACE, JOURNEY / TYPE OF JOURNEY.

c) *Composite - Parent / Child Entity Aggregation:* A parent entity that is composed of dependent child entities that have the same responsibility set as their parent. For example TIMING PATTERN and TIMING POINT IN PATTERN.

d) *Composite - Group of Entities:* An arbitrary Collection of references to other first class entities. For example a GROUP of LINEs / LINE.

e) *Composite – Part Of:* A reflective reference between an entity and itself indicating that an instance may be made up of other instances. For example a STOP PLACE may be part of another STOP PLACE.

f) *Composite Frame Aggregation:* A collection of elements of related types grouped together for the purposes of exchange as a coherent set of data. For example SITE FRAME and its contents.

6.13.2 Mapping Example – Thing Model

6.13.2.1 Thing UML Conceptual MODEL

We illustrate some of the design patterns in NeTEx and their transformation between conceptual model and schema with a hypothetical example of an entity called THING. This may be classified by TYPE OF THING and grouped with GROUP of THINGs. Each THING can be made up of one or more CHILD THINGs. A THING can be associated with many SOMETHING ELSEs. The entities together make up a THING MODEL.

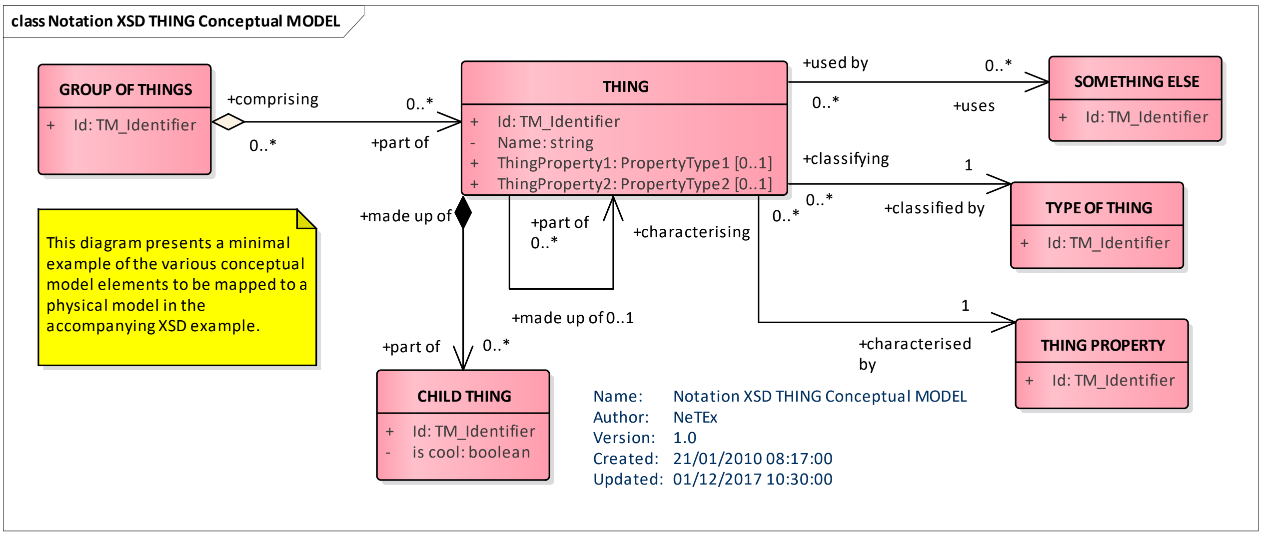


Figure 12 — Mapping Example – Thing Conceptual MODEL

6.13.2.2 Thing UML Physical Thing Model - Overview

The Conceptual Model can be mapped to a Physical model straightforwardly, transforming the names in line with XML naming conventions. For example, 'THING' becomes '***Thing'***, 'SOMETHING ELSE' becomes '***SomethingElse',*** 'TYPE OF THING' becomes '***TypeOfThing'***. All model Elements are made specializations of the ***DataManagedObject*** or other more specific supertype. For example ***TypeOfThing*** is a specialisation of ***TypeOfValue***; ***GroupOfThings*** is a specialisation of ***GroupOfEntities.***

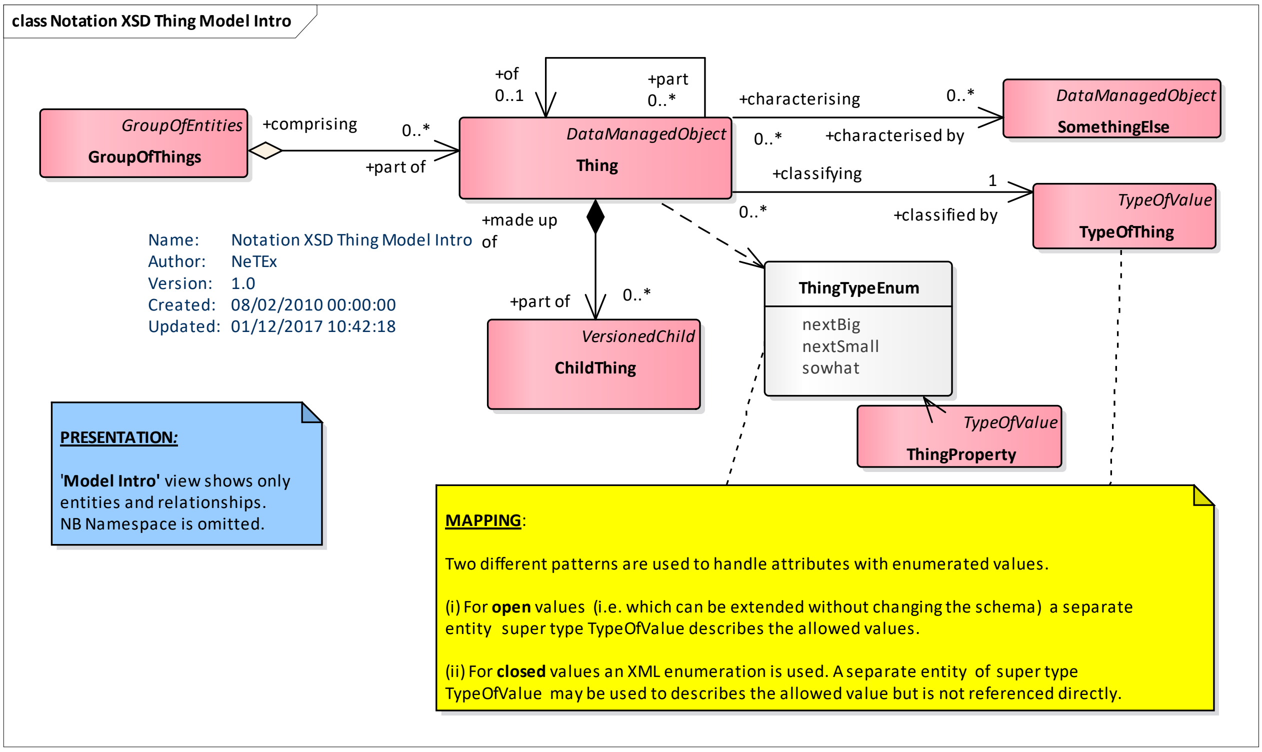


Figure 13 — Mapping Example – Thing Physical MODEL - Summary

6.13.2.3 Thing UML Physical Thing Model - Details

In addition the UML Physical model contains additional detailed attributes to describe the domain, to provide unique identifiers (eg ***id***) and handle versioning and to implement the associations between elements.

Examples of implementing the latter include

— The classifier relationship of THING using <u>TYPE OF THING</u> is a one-to-one relationship and is implemented by placing a ***TypeOfThingRef*** as an attribute on ***Thing***; it contains the version and identifier of ***TypeOfThing***.

— The 'part of' relationship between a THING and a parent THING is implemented by adding a ***ParentRef*** to ***Thing,*** of type ***ThingRef***; it contains the version and identifier of a ***Thing***.

— The aggregation of ***ChildThing*** is a one-to-many composition and is implemented as a ***children*** attribute to hold a collection of ***ChildThing*** instances on ***Thing.***

— The many-to-many association between THING and SOMETHING is implemented as a ***somethingElses*** attribute that holds a collection of ***SomethingElseRef*** references ***each*** identifying a version and identifier of a ***SomethingElse***.

Simple attributes will be assigned an XML data type. For example ***Thing*** / ***Name*** is a *MultilingualString*. ***Thing*** / **ThingProperty1** is of type *decimal*. For classifiers that are restricted to a fixed list of values, an enumeration can be used. For example, ***ThingType*** is of type *ThingPropertyEnumeration.*

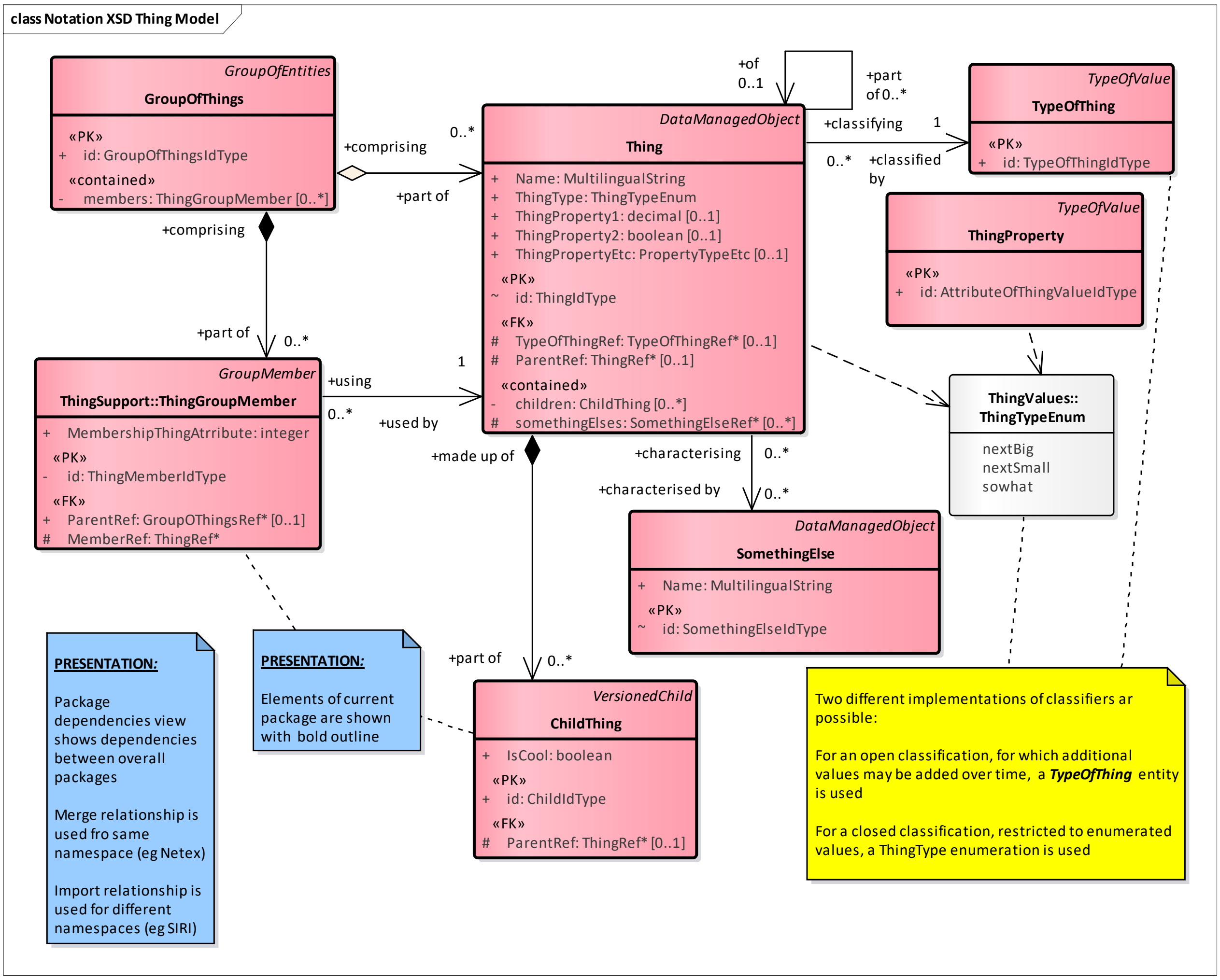


Figure 14 — Mapping Example – Thing Physical Model – Details

6.13.2.4 Thing UML Physical Thing Model – Concise view

An alternative ‘concise’ view of the model suppresses the attributes that implement relationships so that only the functional properties of each entity are visible.

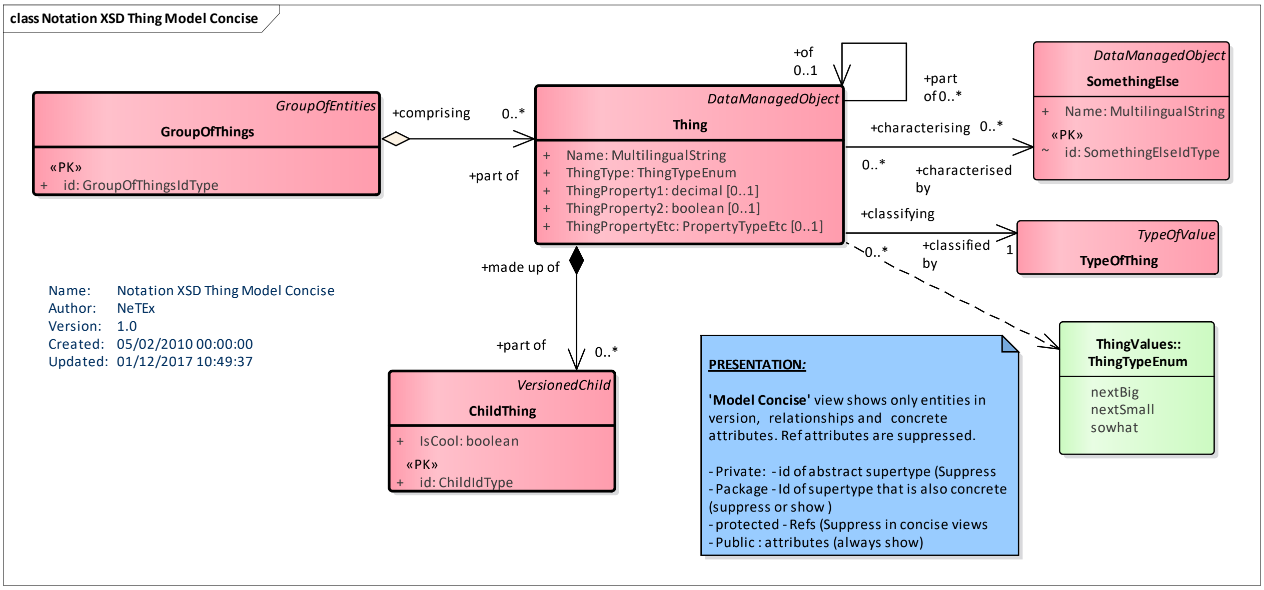


Figure 15 — Mapping Example – Thing UML Physical Model – Concise

6.13.2.5 Thing UML Physical Thing Model – Hierarchy

An alternative ‘Hierarchy view’ of the model shows the inheritance hierarchy. All NeTEx entities are descendants of ***EntityInVersion***, which provides identity and versioning attributes. First class objects are descendants of ***DataManagedObject*** which provides responsibility attributes. Dependent elements are descendants of ***VersionedChild.***

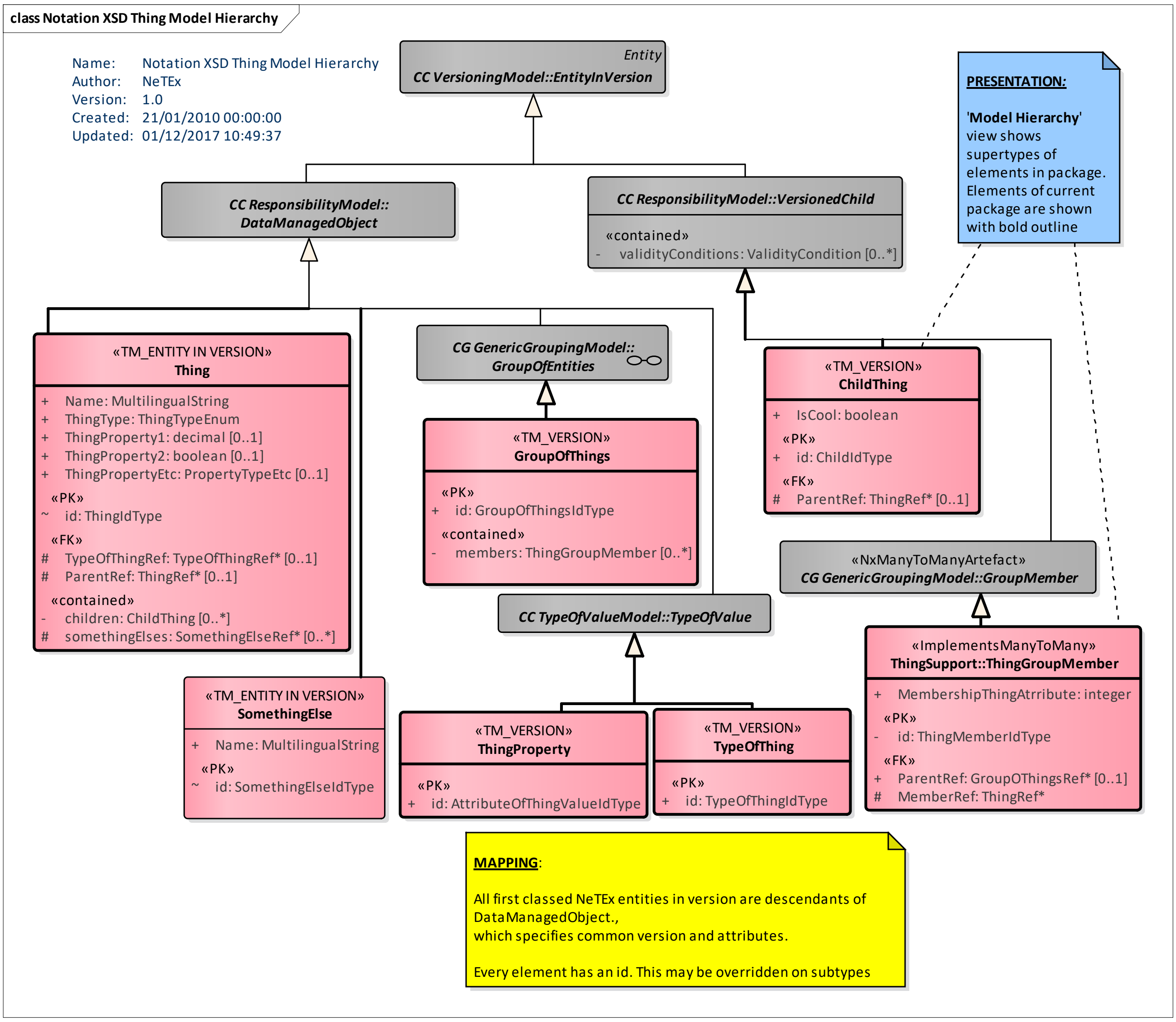


Figure 16 — Mapping Example – Thing UML Physical Model –Inheritance Hierarchy

6.13.2.6 Thing UML Physical Model – Inherited attributes

The result of the inheritance is to provide additional attributes (and behaviours) for the specialisations. For example as shown in the view below which expands out the inherited attributes of ***GroupOfThings, Thing*** and ***TypeOfThing*** to show the properties they inherit from ***GroupOfEntities***, ***DataManagedObject*** and ***TypeOfValue*** respectively.

Figure 17 — Mapping Example – Thing UML Physical Model – Inherited attributes

6.13.2.7 Thing UML Physical Model – Support

The Physical model defines the simple data types used for attributes. The identifiers of each element are specifically typed, for example *ThingIdType*, *SomethingElseIdType,* etc. Strong typing makes it possible to type check references automatically to ensure that the referenced object is of the correct type. For each element an element reference is created, for example ***ThingRef***, ***TypeOfThingRef***, ***SomethingElseRef***, containing the id and an optional version number. This is used to implement associations between elements in the example schema.

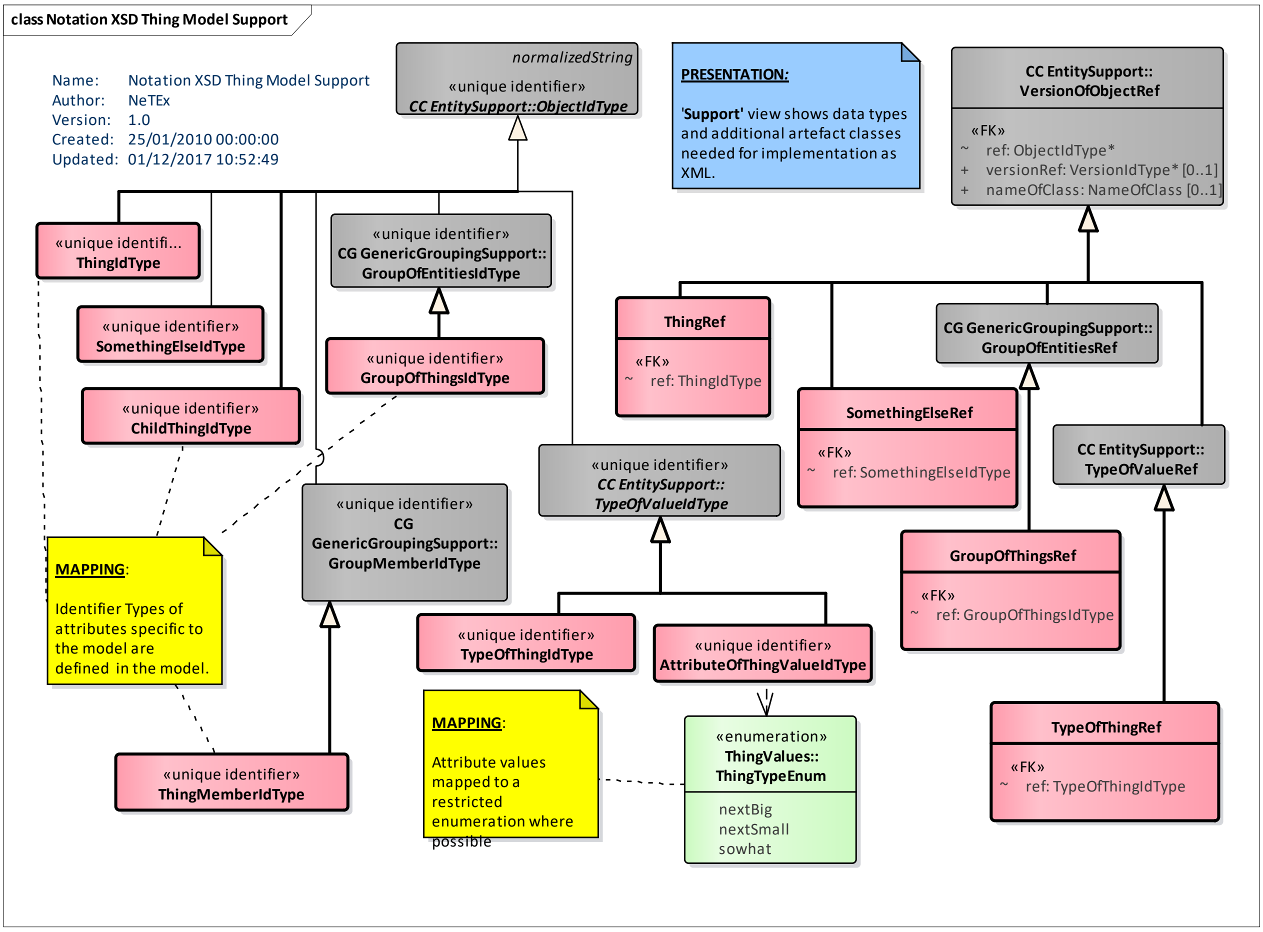


Figure 18 — Mapping Example – Thing UML Physical Model – Support

6.13.2.8 Thing XSD Physical Schema – Thing

The corresponding XSD for THING / Thing will have both inherited and specific attributes and elements.

Table 8 — *Thing – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	THING inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ProjectionIdType*	1:1	Identifier of THING.
	Name	*xsd:normalized-String*	0:1	Description of THING.
«enum»	***ThingType***	*ThingTypeEnum*	0:1	Type of THING as a set of enumerated values.
«FK»	***ParentThingRef***	*ThingRef*	0:1	Reference to a parent THING.
	Name	*xsd:normalizedString*	0:1	Description of THING.
«cntd»	***childThings***	*ChildThing*	0:*	One or more component CHILD THINGs.
«cntd»	***somethingsElse***	*SomeThingElseRef*	0:*	One or more instances of SOMETHING ELSE associated with this THING.

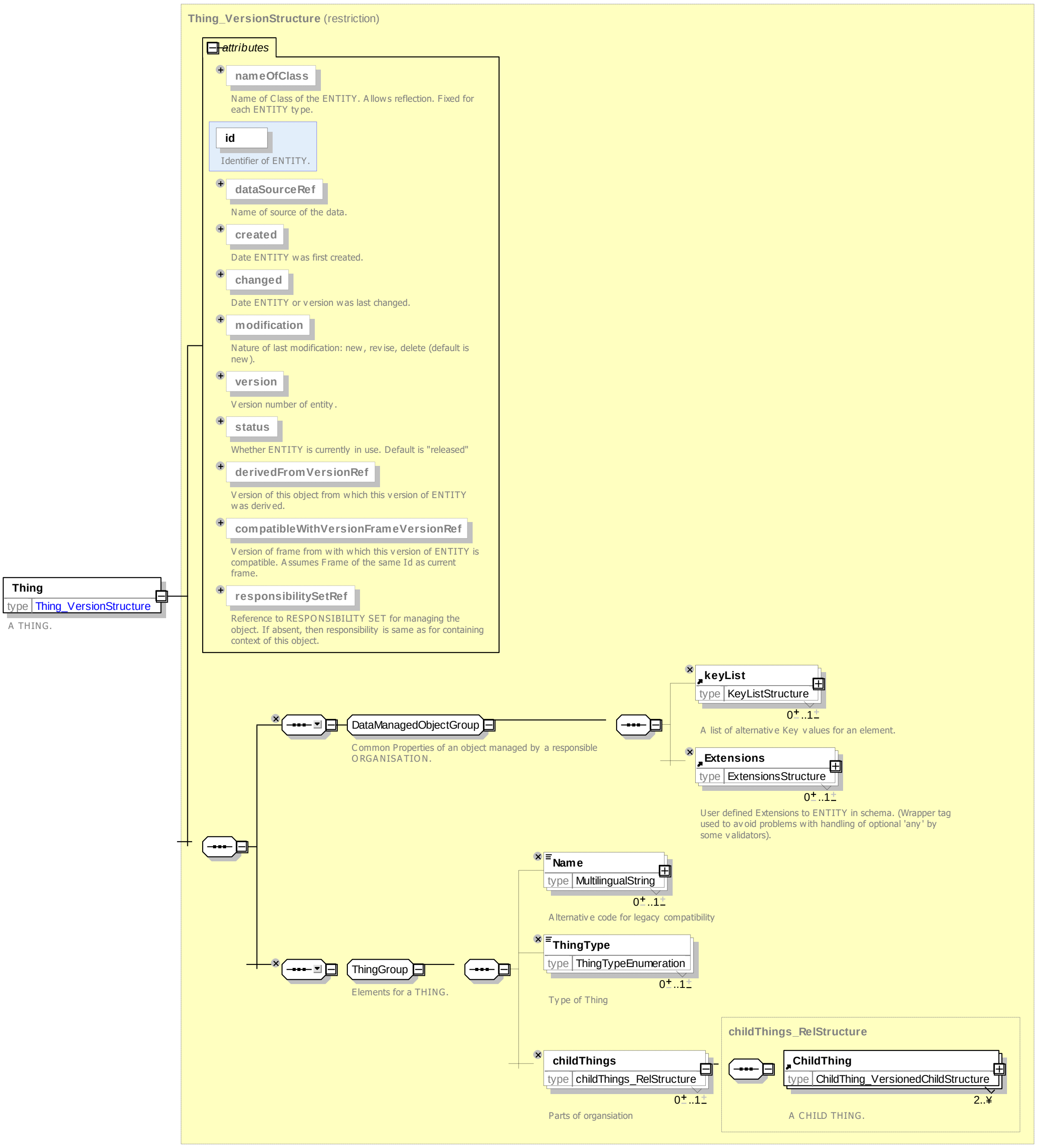


Figure 19 — XSD implementation of Thing

6.13.2.9 Thing XSD Physical Thing Schema – Thing Concise

Normally it is sufficient to show only the attributes specific to the entity. The properties of the parent will be separately documented.

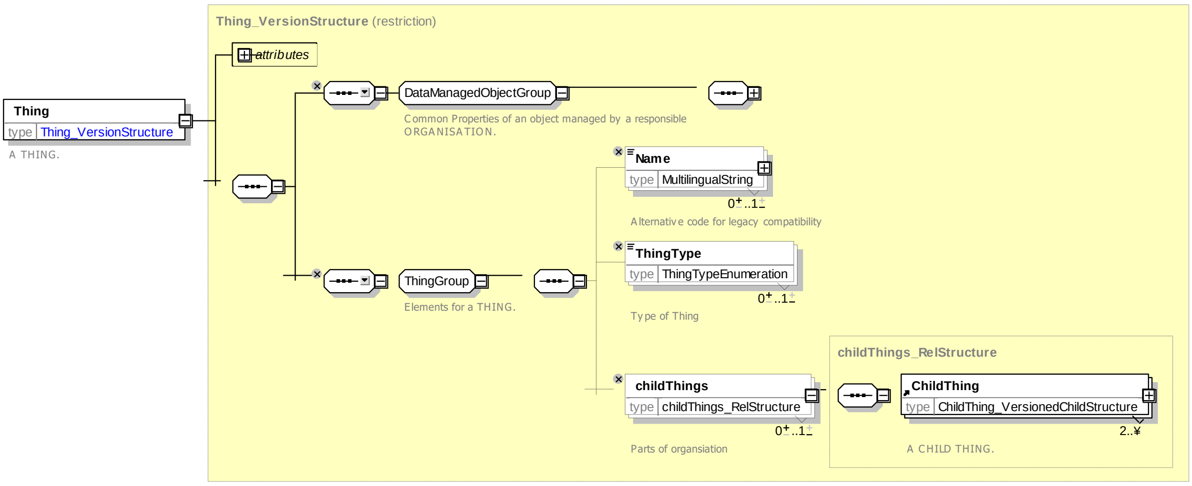


Figure 20 — XSD implementation of Thing – Concise

6.13.2.10 Thing – XML Example

The following XML fragment shows a sample instance of a ***Thing*** containing two ***ChildThing*** instances and classified with a ***TypeOfThing*** instance***.*** It has a parent ‘abc:43262’ and is associated with three other unvalidated instances of ***SomethingElse*** elements.

EXAMPLE

```
<Thing id="abc:3456" version="001">
    <Name lang="en">Lunch</Thing>
    <ThingType>nextBig</ThingType>
    <TypeOfThingRef ref=”abc:Val01” version=”any”/>
 <ParentThingRef ref=”abc:43262” version=”any”/>
    <ThingProperty2>bar</ThingProperty2>
    <children>
            <ChildThing id="abc:34561">
                    <IsCool>true</IsCool>
            </ChildThing>
            <ChildThing id=" abc: 34562">
                    <IsCool>false</IsCool>
            </ChildThing>
    </children>
 <somethingsElse>
    <SomethingElse ref="abc:66561" >
    <SomethingElse ref="abc:68752" >
    <SomethingElse ref="abc:21756" >
    </somethingsElse >
</Thing>
<TypeOfThing id=" abc:Val01" version="any">
    <Name lang="en">Good Thing</Thing>
</TypeOfThing>
```

6.13.3 Mapping Example – Handling Inheritance the SubThing Model

6.13.3.1 SubThing UML Conceptual MODEL

NeTEx makes extensive use of inheritance to factor out common properties of objects. This makes models more concise and more maintainable. The inheritance programming model of XML is more limited than the general model supported in UML.

In the Conceptual MODEL, multiple inheritance is possible (for example an element can be both a PLACE and a ZONE). In XML - and the physical model, only single class inheritance is possible, so multiple inheritance must be mapped to single inheritance.
For the most part mapping between model and schema is straight forward as a single hierarchy suffices for most elements. However in a few cases additional techniques are used in NeTEx.

We illustrate some of the design patterns for inheritance in NeTEx and their transformation between conceptual model and schema with a hypothetical example.

This example includes a "worst case" of threefold type ancestry in the conceptual model that cannot be directly implemented in the XML physical model, since XML supports only single inheritance.

— **Single Inheritance:** In the example Conceptual Model, SUB A is a subtype of only ANCESTOR A. Similarly, SUB SUB A is a subtype of SUB A. This is reflected in the Physical model in the class hierarchy of ***AncestorA***, ***SubA***, and ***SubSubA***.

— **Use of Single Inheritance to resolve Multiple Inheritance:** In the example Conceptual Model, SUB AB is a subtype of both ANCESTOR A, and ANCESTOR B. In the Physical model, by making ***AncestorB*** a subtype of ***AncestorA*** a single hierarchy can be used, thus ***SubAB***. Has both ***AncestorA*** and ***AncestorB's*** properties.

— **Use of a Group to implement multiple inheritance.** In the example Conceptual Model, SUB ABC is a subtype of both ANCESTOR A, ANCESTOR B and ANCESTOR C. Thus no single inheritance hierarchy can be created. In the Physical model, an ***AncestorC*** group is created to hold the properties of ***AncestorC***. This can be reused on multiple subtypes if needed. ***SubABC***, directly inherits from ***AncestorB*** (which is in turn a type of ***AncestorA***) and embeds the ***AncestorC*** group so it has properties of all its ancestors.

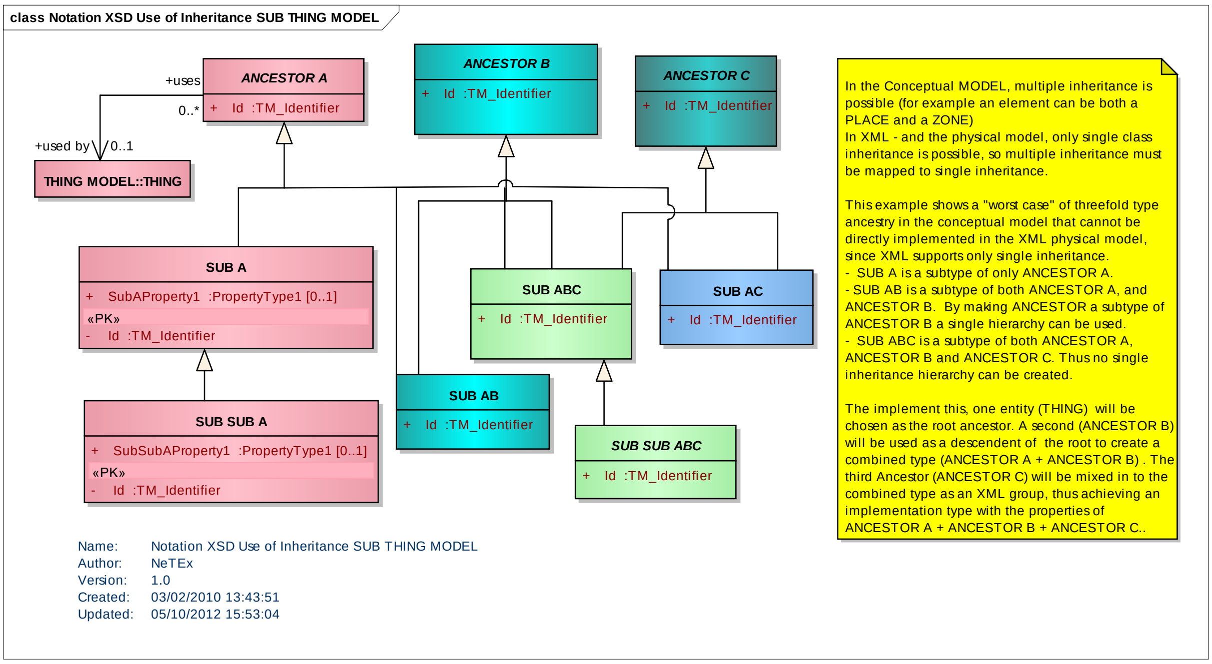


Figure 21 — Mapping Example – SubThing Conceptual MODEL

6.13.3.2 SubThing UML Physical Model – Inherited attributes

The following diagram introduces the Physical model discussed above.

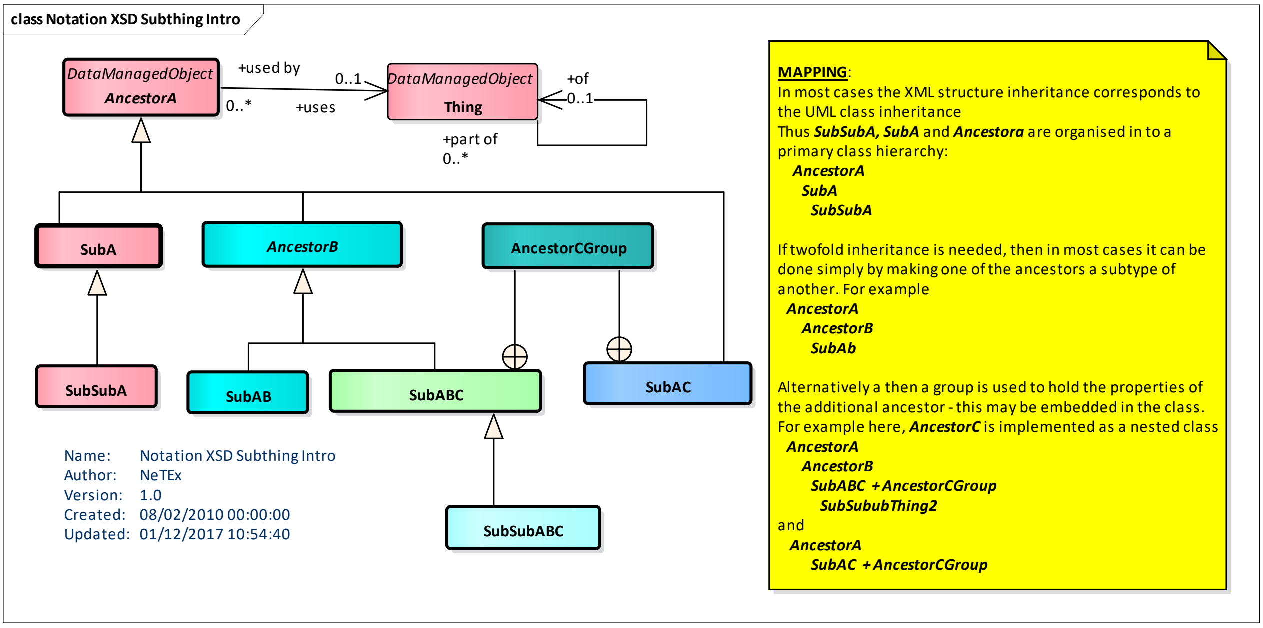


Figure 22 — Mapping Example – SubThing Physical Intro

The following diagram elaborates the Physical model to show attributes.

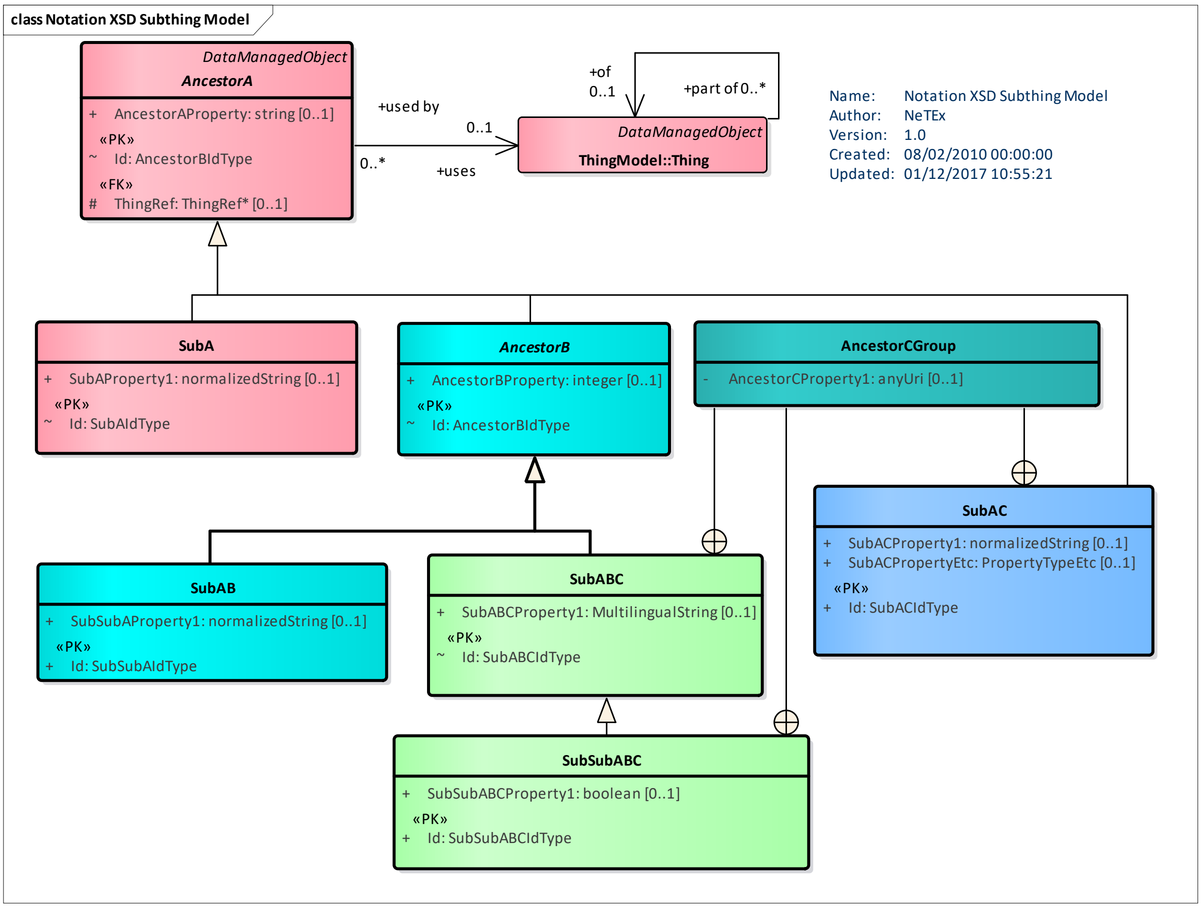


Figure 23 — Mapping Example – SubThing Physical Model

6.13.3.3 Subthing XSD Physical Schema – Ancestor A

The root element ***AncestorA*** inherits properties from ***DataManagedObject.***

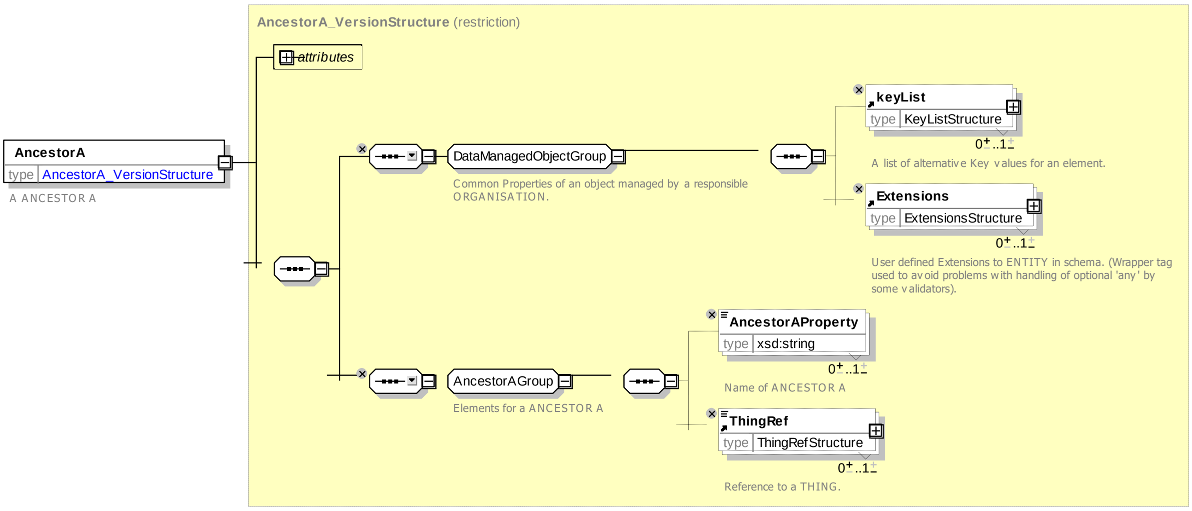


Figure 24 — XSD implementation of Ancestor A

6.13.3.4 Subthing XSD Physical Schema – Ancestor B

AncestorB inherits properties from ***AncestorA.***

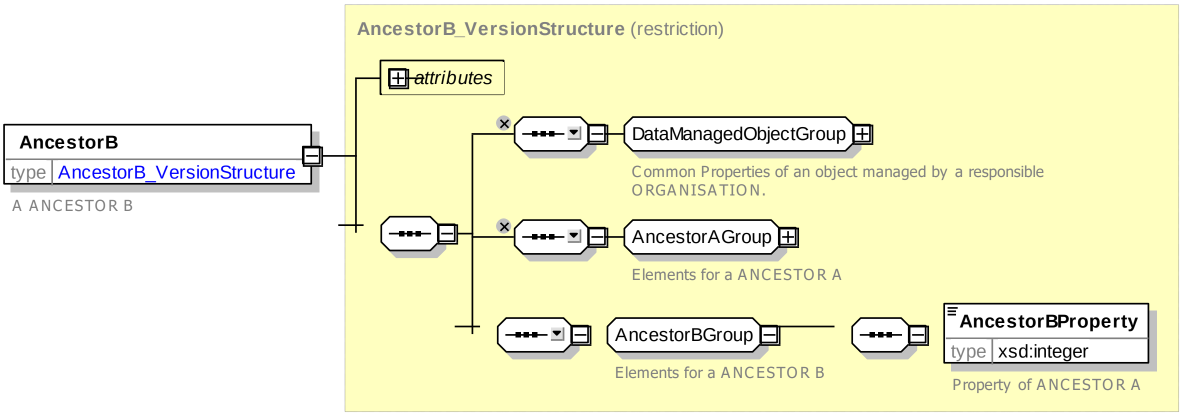


Figure 25 — XSD implementation of Ancestor B

6.13.3.5 Subthing XSD Physical Schema – Ancestor C

AncestorCGroup defines properties for ***AncestorC.***

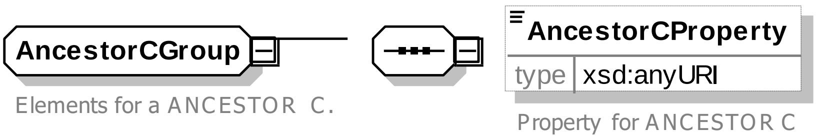


Figure 26 — XSD implementation of Ancestor C Group

6.13.3.6 Subthing XSD Physical Schema – SubA

SubA inherits just properties from ***AncestorA.***

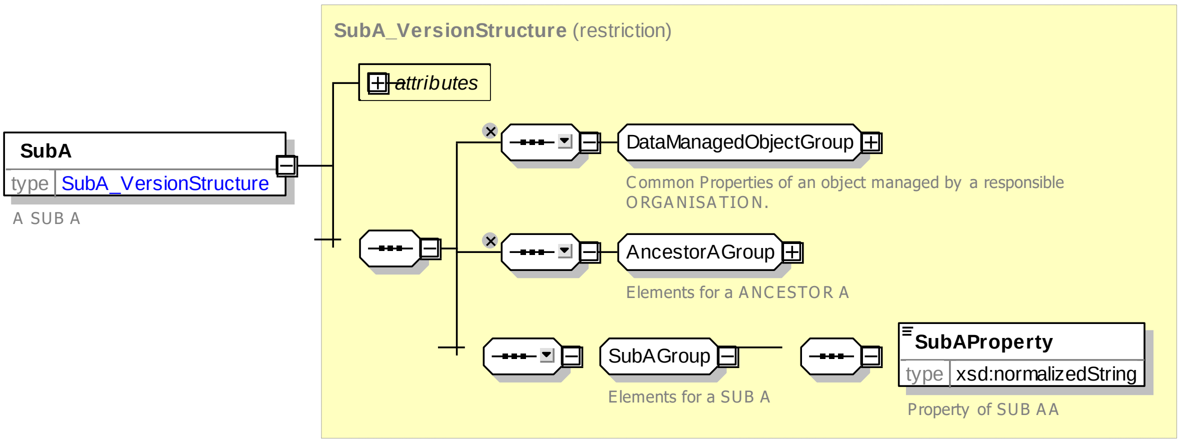


Figure 27 — XSD implementation of SubA

6.13.3.7 Subthing XSD Physical Schema – SubABC

SubABC inherits properties from ***AncestorB*** and embeds ***AncestorCGroup.***

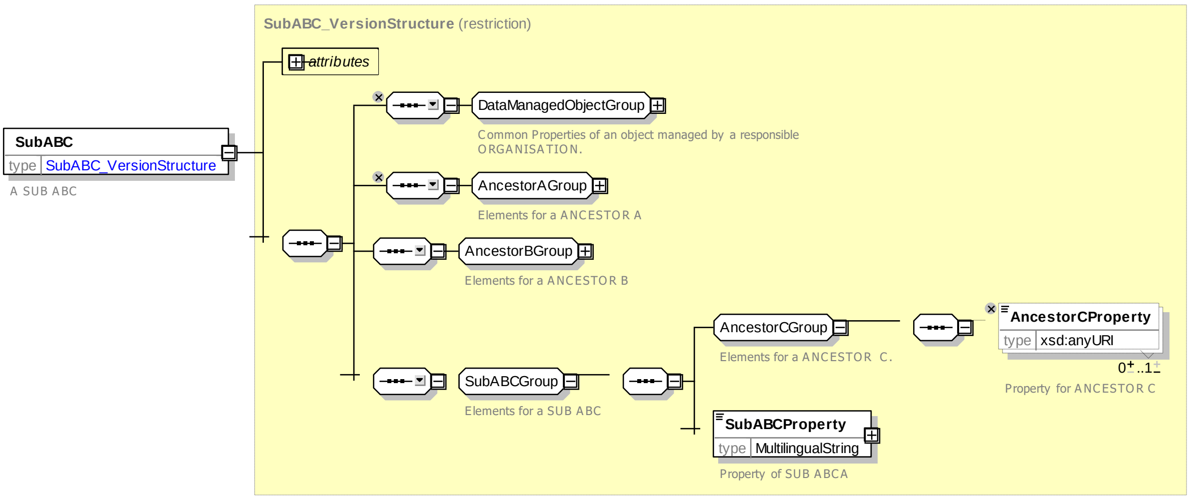


Figure 28 — XSD implementation of SubABC

7 NeTEx Framework - Conceptual and Physical data model

NOTE NeTEx representational model is divided in a number of separate submodels covering different aspects of the Network. For ease of understanding, the submodels are presented one at a time, each describing only a small set of related concepts. Each sub-model is described at three levels of abstraction:

1. A conceptual data model, mainly extracted from Transmodel reference model and sometimes extended with some NeTEx specific concepts
2. A physical data model, introducing how the conceptual model will be transformed into an expanded form suitable for an XSD implementation (this may include simplifications, merging, inheritance management strategy, etc.). It will also provide all the needed attributes.
3. The XSD schema which will be used to exchange data. Examples are given of the main elements.

7.1 Introduction

This section describes the common NeTEx Framework that is shared by all NeTEx functional parts. The framework has two different aspects.

Common framework mechanisms: The framework provides mechanisms for common aspects of all NeTEx objects that are needed for effective data exchange, such as versioning, validity, grouping, and responsibility tracking. The mechanisms, implemented through common supertypes and containers, and specialised in the various NeTEx functional modules, can be understood and implemented uniformly for all NeTEx components, rather than on an ad-hoc basis.

Reusable components: Certain common low level components, for example TRANSPORT MODES, CALENDARS, DAY TYPES, etc. are not specific to any particular functional part of NeTEx but are widely used in several different functional areas. Such components are defined centrally as part of the framework.

The NeTEx framework is split into four main submodels, each defined as a UML package.

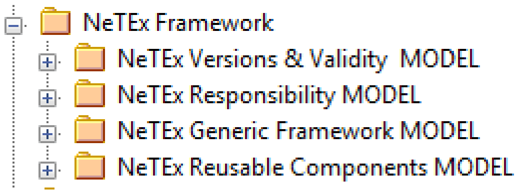


Figure 29 – NeTEx Framework packages

The Versions & Validity model: describes the successive versions of data elements and the conditions to be attached to elements to precisely know when they should be used

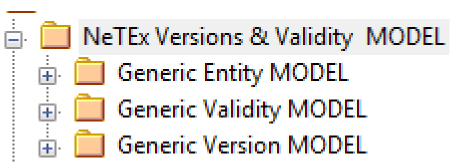


Figure 30 – NeTEx Versions & Validity MODEL packages

The Responsibility model: describes the stakeholder organisations related to the objects that possibly can be exchanged using NeTEx and expresses the type of responsibility or role they have with respect to these objects.

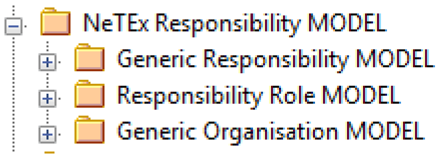


Figure 31 – NeTEx Responsibility MODEL packages

The Generic Framework model: describes a number of generic objects and representational mechanisms that are not specific to transport but which are specialised or used by NeTEx transport related objects.

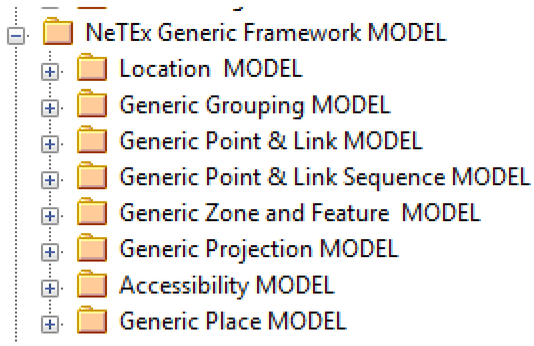


Figure 32 – NeTEx Generic Framework MODEL packages

NeTEx Reusable Components model: describes generic and reusable objects (through NeTEx parts 1, 2 and 3) specific to public transport.

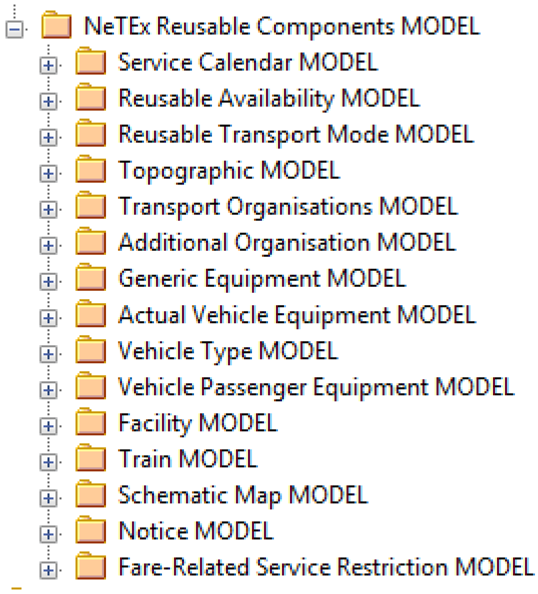


Figure 33 – NeTEx Reusable Components MODEL packages

7.2 Implementing Transmodel framework features in NeTEx

NOTE Transmodel includes generic mechanism for a number of commonly required features such as versioning and grouping. Because NeTEx describes a physical implementation of Transmodel in XML, defined using a W3C XSD schema, which requires strong static typing in order to have the benefits of the built in validation, it must chooses a specific concrete implementation of certain of the mechanisms which are done as generic types in Transmodel. This can be seen in particular in the following section on VALIDITY and VERSION FRAMEs. The Physical implementation is thus more specific and restrictive than the generalised conceptual formation, but has the advantage of being easier to implement and can be described more concretely, making it in some ways easier to understand. Thus rather than having just a general purpose frame to organise data elements for exchange, NeTEx by preference uses a number of explicit specialisations of the Frame, each of which is restricted to specific types of content.

7.3 Versions & Validity

7.3.1 Introduction

Information systems for public transport operation typically require the exchange of many different types of data, produced by different organisations or operating divisions, and are subject to a multistage lifecycle from planning through to production and realisation in real-time. These data are continuously evolving and are subject to a variety of different validity conditions as to when they are current, and as to which data is needed for a particular purpose. NeTEx includes uniform version and validity mechanism to address these requirements; the mechanisms are part of the NeTEx framework and that can be applied to all data elements throughout their various lifecycles.

The **versioning** model allows successive versions of data elements to be identified, allowing the fine grained identification of just those elements that have changed, and the auditing of changes. All references can also be versioned so that for composite data sets that comprise a number of related elements it is possible to be precise as to which version of each element is required. The versioning model also allows schemes where the responsibility for maintaining different parts of the data is split among several organisations and systems, each providing its partial data separately. In this case references to external data are not explicitly versioned, but instead the correct version of the different referenced entities are deducted from validity conditions when combining the data.

A **version frame** mechanism provides a versionable container that allows a coherent set of related elements to be exchanged. Since pragmatically actual systems that contain data to be exchanged differ in the sophistication of their support for versioning, the mechanisms are designed so that they may be used either just in a course grained manner at the level of the whole data set, or if support is available, in a more powerful way at the level of the individual data element.

The **validity** model allows conditions to be attached to elements as to when they are current or the circumstances in which they should be used. Validity conditions can be attached to specific elements and also, through version frames, to whole sets of objects so that it is possible to be explicit about the exact conditions governing the coherence and relevance of data. This makes it possible for systems to express the currency conditions for data they require and to describe the validity of data that is returned by a system.

7.3.2 Version & Validity – Model Dependencies

The versioning mechanisms are part of the core NeTEx framework, and are provided by common set of modules that are referenced by all other NeTEx modules. The fundamental models are shown in the following diagram and described in detail in the following sections.

— The ENTITY model describes the NeTEx basic object structure.

— The VERSIONING model adds in version control elements and attributes.

— The VERSION FRAME model provides VERSION FRAMES with which to group multiple instances of versions of entities that make up a coherent version set.

— The RESPONSIBILITY model adds in metadata for ENTITY ownership and roles for data management.

— The VALIDITY package defines generic validity conditions for use in the framework.

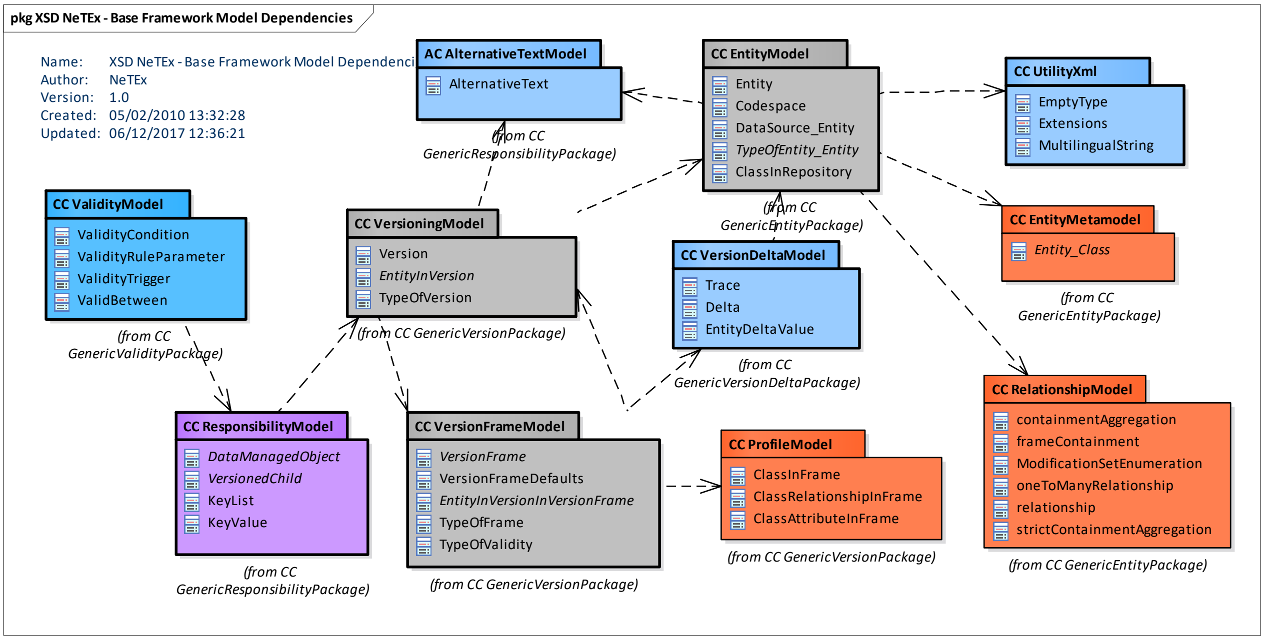


Figure 34 — Version & Validity – Model Dependencies (UML)

7.3.3 Generic Entity

The sections below describe the common base structure of the objects that can be exchanged with NeTEx.

7.3.3.1 Generic ENTITY – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

A NeTEx document containing exchanged data will comprise specific versions of objects that are passed between different systems. Each system shall hold its own representation of the object and may also retain a history of different versions.

The ENTITY represents an actual object instance of data present in an exchanged data set. An ENTITY may represent any instance of a CLASS IN REPOSITORY, corresponding to an instance of the object as stored in a specific database. All NeTEx objects are formal descendants of ENTITY.

An ENTITY is linked to a DATA SOURCE which is used to identify the system having produced the version of the ENTITY in the exchanged data set. Each ENTITY may have its own DATA SOURCE.

CLASSes IN REPOSITORY can be grouped into sets of coherent versions using a CLASS IN FRAME. CLASS IN REPOSITORY and CLASS IN FRAME are part of the Transmodel conceptual model and help to make clear the difference between an instance of an object for exchange (which is in effect always a particular version of an ENTITY) and an instance in a specific database (which may include multiple versions of an ENTITY). However since NeTEx is only concerned with the exchange model as serialised into an XML schema, and not with the repository structures, only the ENTITY and DATA SOURCE are realised in the NeTEx XSD schema.

The TYPE of ENTITY defines a set of sub categories that can be used to make arbitrary classifications of a specific ENTITY (thus it is really a “category of ENTITY” rather than a class or type). TYPE OF ENTITY is an abstract mechanism that is present in Transmodel to indicate the possibility of categorisation. Actual NeTEx objects generally have a more specific categorisation, e.g. TYPE OF POINT, TYPE OF SERVICE, etc. that specifies a category that is specific to the ENTITY type. TYPE OF ENTITY may be implemented in two different ways.

1) As a closed list of values as specified by an enumeration. The values are fixed and intended to be strictly standardised in the XML. In this case the XML element will be named xxxType, for example, ***StopPlaceType***, ***QuayType.***

2) As an open list of values to which an implementer may add additional values. For example, TYPE OF POINT, TYPE OF PLACE, TYPE OF SERVICE, etc. In this case the XML element will be named TypeOfXxxx for example ***TypeOfPoint, TypeOfPlace, TypeOfService.***

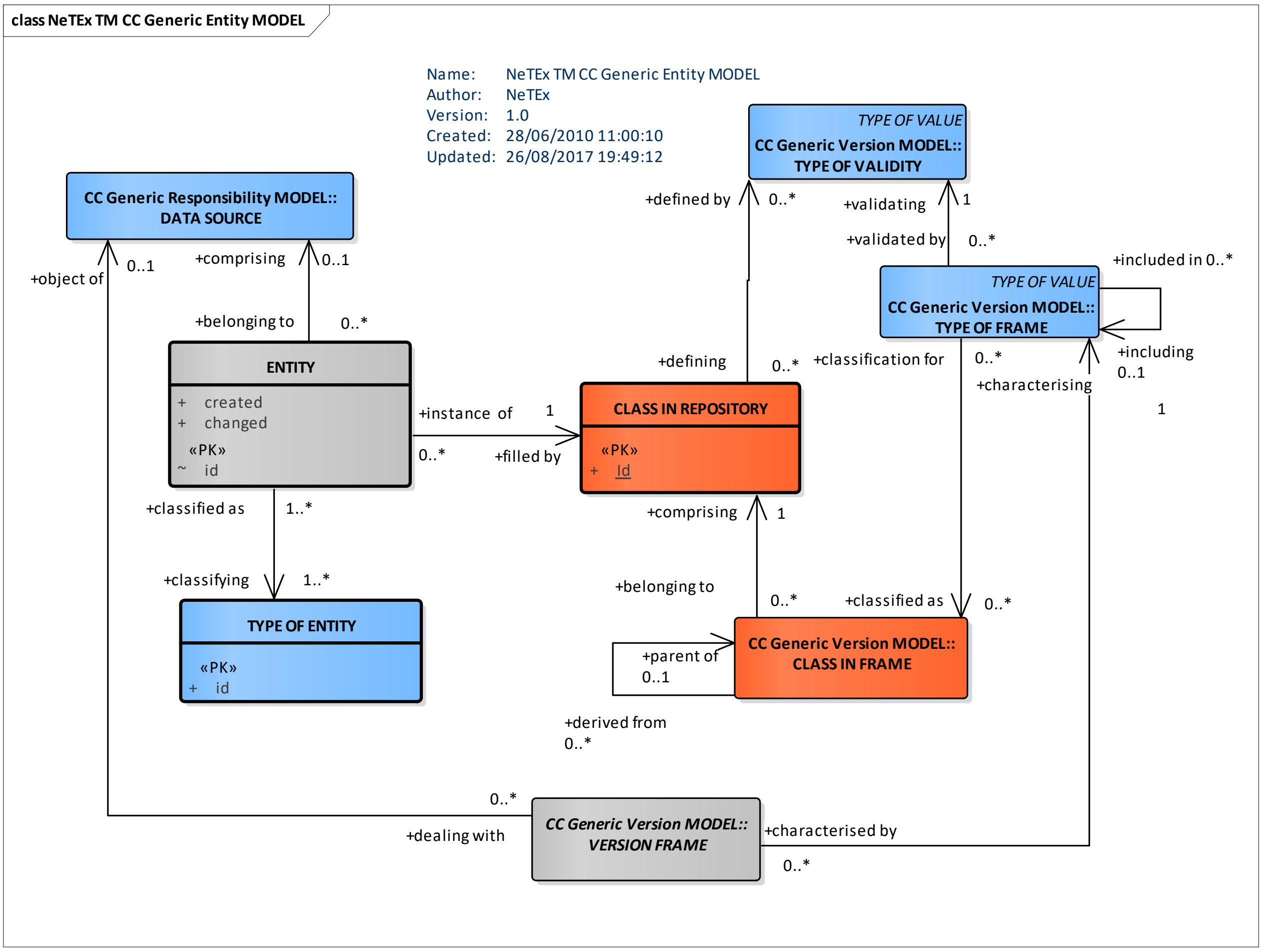


Figure 35 — Generic Entity – Conceptual MODEL (UML)

7.3.3.2 Generic Entity – Physical Model

The physical Generic Entity Model has classes corresponding to those of the conceptual model, but only the ENTITY and DATA SOURCE and CODESPACE elements are implemented in the XSD, since CLASS IN REPOSITORY etc. relate to the database of the participating system, not the exchange format.

An extra object is added in the physical model: the ***Codespace. Codespace*** is similar to XML concept of a *Namespace* that enables the names of elements and attributes in an XML schema to be unique, using the generic W3C domain mechanism. We use it also to ensure that the names of instances of elements in an XML document to be unique – see examples later below. This allows data elements from different countries, different regions, and different organisations to be combined without clashes. Having an explicit ***Codespace*** object makes it possible for applications to reflect on and compute over the **Codespaces** in a generic manner. The ***Codespace*** is used for two purposes; (i) to give a unique context within which to identify objects uniquely; (ii) to manage the distributed allocation of new identifiers to new instances of objects. See ADMINISTRATIVE ZONE later.

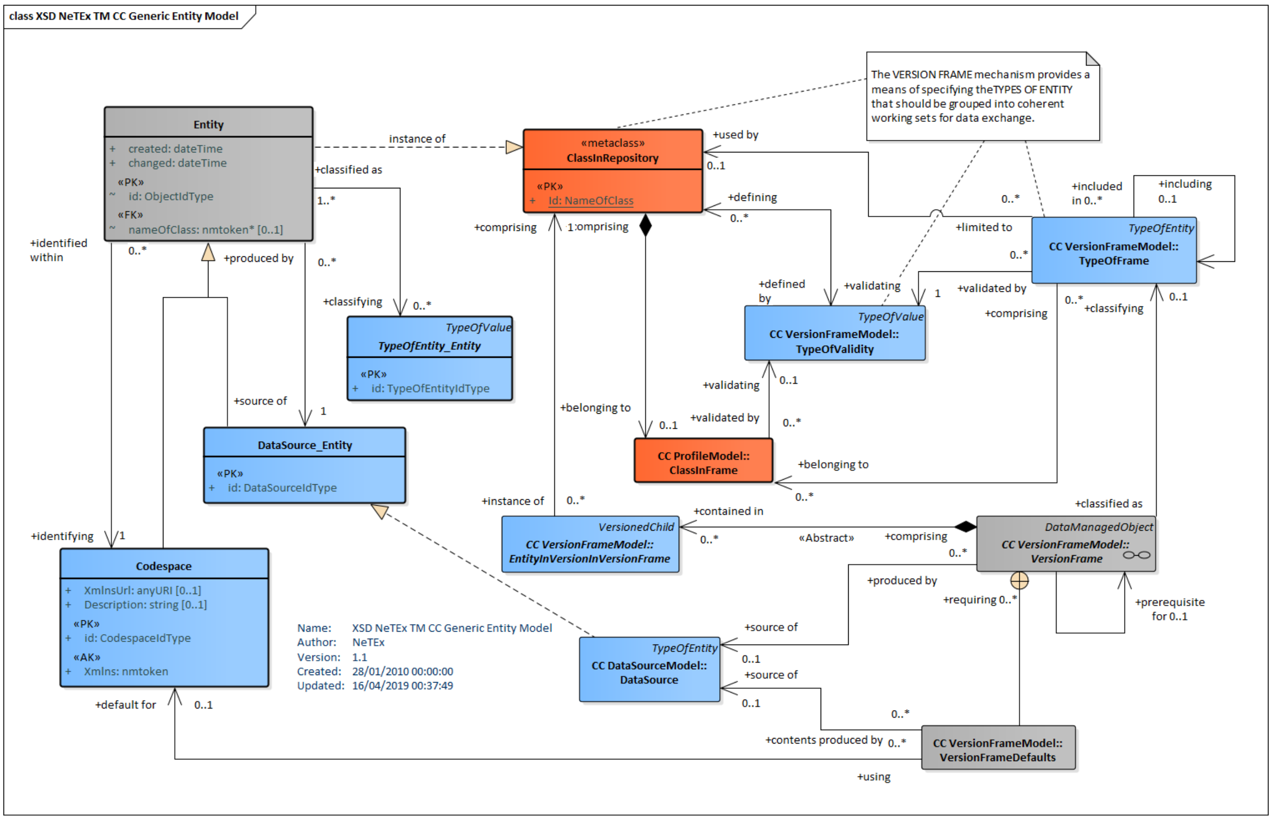


Figure 36 — Generic Entity – Physical Model (UML)

7.3.3.2.1 Codespace – Physical Model

The following diagram shows how a code space may be assigned to a specific ADMINISTRATIVE ZONE.

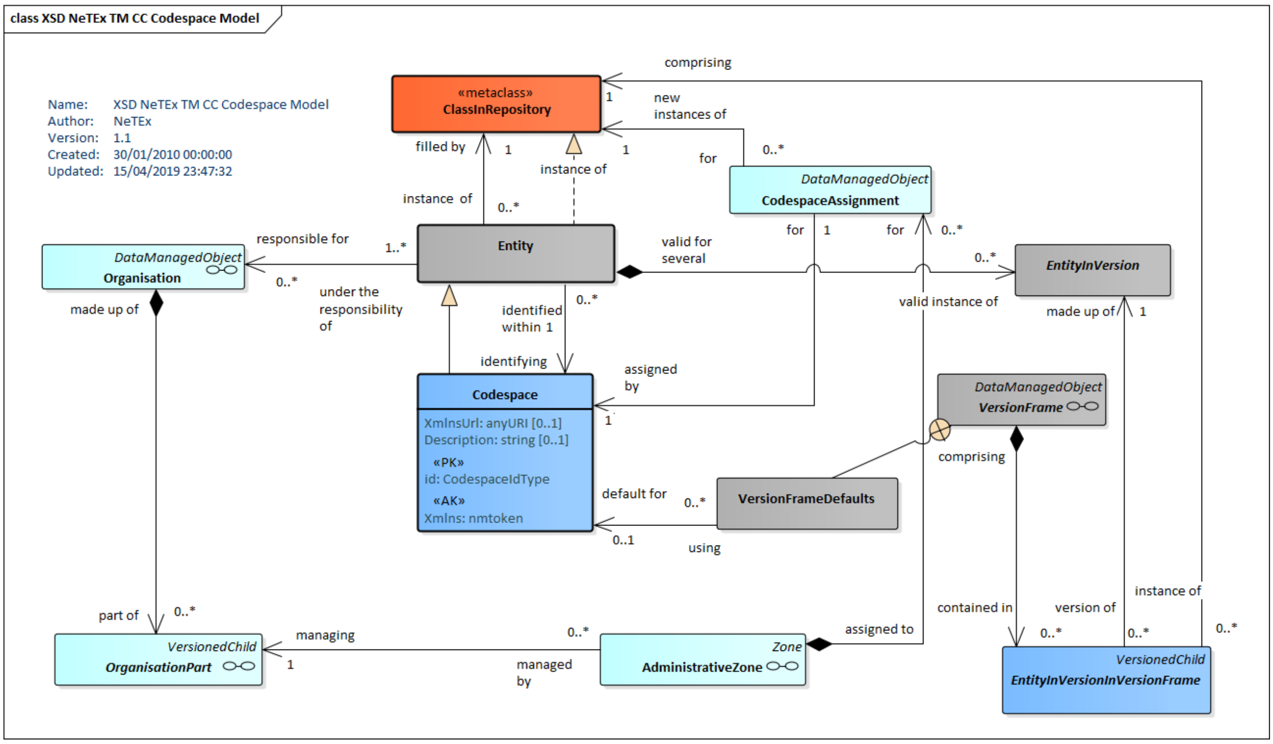


Figure 37 — Codespace – Physical Model (UML)

7.3.3.3 Generic Entity – Attributes and XSD

7.3.3.3.1 Entity – Model Element

An ENTITY is any data instance to be managed in an operational Version Management System. When several data sources coexist (multimodality and/or interoperability), an ENTITY has to be related to a given DATA SYSTEM in which it is defined

Table 9 — *Entity* – Element.

Classification	Name	Type	Cardinality	Description
«atr»	nameOfClass	*xsd:NMTOKEN*	0:1	Name of class of which this is an instance. Used for reflection.
«PK»	id	*ObjectIdType*	1:1	Identifier of ENTITY. Unique within CODESPACE.

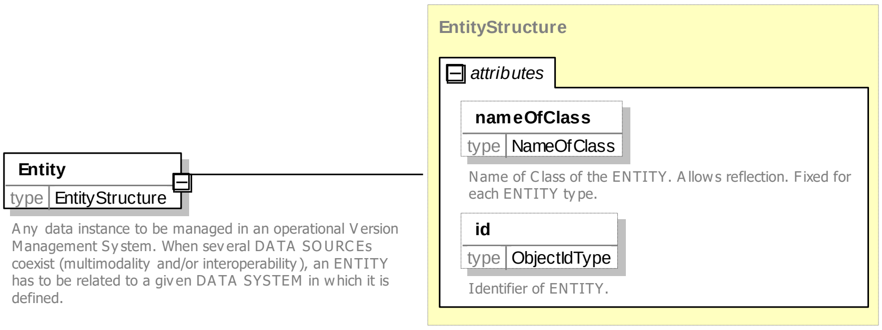


Figure 38 — *Entity – XSD*

7.3.3.3.2 Codespace – Model Element

A CODESPACE defines the context within which an identifier of an object in a document is unique. The use of CODESPACEs in NeTEx also allows an economical encoding of identifiers in an XML document so that a common prefix does not have to be repeated on each element in the document.

One or more CODESPACEs can also be associated with each ADMINISTRATIVE ZONE to reserve their use of that namespacae and a prefix or range value for the allocation of identifiers to new entities within that zone.

A CODESPACE is specified as a path expression by using internet domains, through IANA (Internet Assigned Numbers Authority), which provides a mechanism for registering global uniqueness. For example tfl.gov.uk, bahn.de, ratp.fr, foo.com, or sbb.de. These can be declared in a Codespace element **:xxx="path"**

Each Codespace has an identifier which may be used in the document using a ***Codespace*** declaration on a VERSION FRAME.

EXAMPLE CODESPACE declaration:

```
<CompositeFrame version="any" id="mybus:CompositeFrame:CF1">
        <!--- ======= CODESPACEs======== -->
        <codespaces>
            <Codespace id=" napt ">
                <Xmlns> napt </Xmlns>
                <XmlnsUrl>http://naptan.org.uk/naptan</XmlnsUrl>
```

```
            <Description>UK NaPTAN </Description>
        </Codespace>
        <Codespace id=" era ">
            <Xmlns> era </Xmlns>
            <XmlnsUrl>http://era.org.eu/</XmlnsUrl>
            <Description>European Rail AUthority</Description>
        </Codespace>
    </codespaces>
```

CODESPACE use:

id=napt:4701234567, ref= napt:4701234567
id=era:4501234345, etc

An application processing a document is expected to understand any rules peculiar to interpreting the identifers in a specific ***Codespace***. For example NaPTAN stop identifers in the naptan codespace have the structure '*999 0 XXXXXXXXX'* where '*999'* is an area prefix, '*0*' is fixed and '*XXXXXXXXX'* is a number unique within the area. The XML will not enforce the rule but an implementation that is aware of the codespace is able to do so.

Table 10 — *Codespace* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Entity*	::>	CODESPACE inherits from ENTITY.
«PK»	***id***	*CodespaceIdType*	1:1	Identifier of CODESPACE. Unique within document.
«AK»	***Xmlns***	xsd:*NMTOKEN*	1:1	Codespace prefix, unique within a given XML document e.g. 'napt'
	XmlnsUrl	*xsd:anyURI*	0:1	Codespace path. Globally unique. For example, http://naptan.org.uk/naptan or http:/vdv.de/vdv/haltstelle/
	Description	*xsd:string*	0:1	Description of CODESPACE.

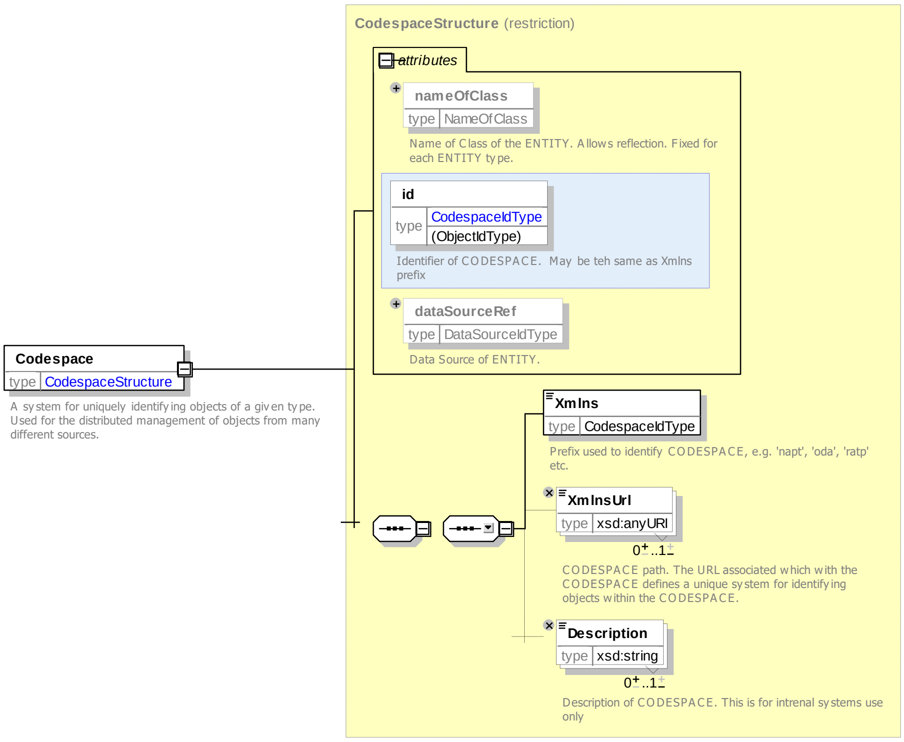


Figure 39 — *Codespace – XSD*

7.3.3.3.3 DataSource_Entity – Model Element

The DATA SOURCE ENTITY identifies the system that has produced the data.

References to a data source are useful in an distributed computer system. The ENTITY itself is a in-repository concept that is not implemented in the NeTEx schema – see DataSource for the versioned instance that is exchanged.

Table 11 — *DataSource_Entity* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	DATA SOURCE inherits from TYPE OF VALUE.
«PK»	***id***	*DataSourceIdType*	1:1	Identifier of DATA SOURCE.
	Email	*EmailAddressType*	1:1	Email address to contact DATA SOURCE organisation.

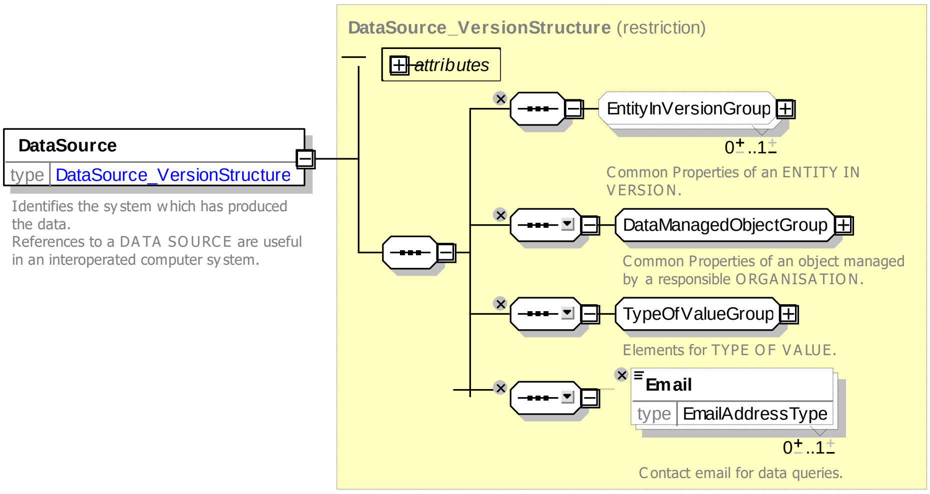


Figure 40 — *DataSource – XSD*

7.3.3.3.4 TypeOfEntity – Model Element

Classification of ENTITies, for instance according to the domain in which they are defined or used.

In the NeTEx physical model where there is a need to categorise an element specifically according to its type, a specialized "type of type" is provided, for example, ***TypeOfPlace*** implements the TYPE OF PLACE entity from the Conceptual model; ***TypeOfPoint*** implements the TYPE OF POINT entity from the Conceptual model**,** etc. In some cases the ***TypeOfXxx*** elementType is implemented as a closed list of enumerated values, in which case it will be named ***xxxType***, for exampleby ***StopPlaceType***.

Table 12 — *TypeOfEntity* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE OF ENTITY inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfEntityIdType*	1:1	Identifier of TYPE OF ENTITY.

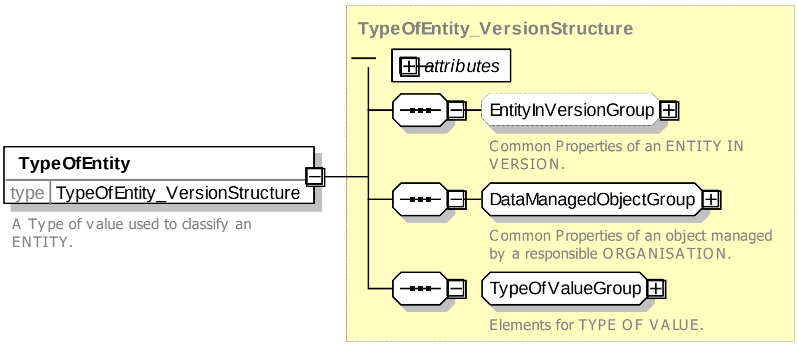


Figure 41 — *TypeOfEntity – XSD*

7.3.3.3.5 ClassInRepository – Model Element

Any ENTITY name belonging to the repository, e.g. DAY TYPE, PROPERTY OF DAY, TIME BAND, VEHICLE TYPE, DUTY, etc. are relevant instances of CLASS IN REPOSITORY in the context of Version Management.

This is a meta class that can be used to reference the available entity types.

Table 13 — *ClassInRepository* – Element

Classification	Name	Type	Cardinality	Description
«PK»	***id***	*NameOfClass*	1:1	Unique identifier of class of ENTITY. There will only be a single instance, which must be the class name, e.g. ***JourneyPattern***, ***StopPlace***, etc.

The ***ClassInRepository*** is not implemented in the NeTEx XSD Schema, however *nameOfClass* may be specified as an attribute in certain places to facilitate mapping to repositories.

7.3.3.4 XML Examples for generic Entities

7.3.3.4.1 Codespaces & Identifiers – XML Example Fragment

The following example shows the use of CODESPACEs to declare uniqueness. In the example the ROUTE LINKs come from one CODESPACE (*http://www.foo.eu/stuff*, using local prefix '*foo')*, and the ROUTE POINTs from two others (*http://www.bar.de*, using local prefix '*bar', and 'http://www.fum.ch*, with local prefix '*fum,'*). These can be used to give unique CODESPACEs for each identifier. Note that two of the points (A and C) have the same number '01', but are rendered unique by different CODESPACEs. The prefixes are arbitrary and local to the document. The urls are persistent and fixed.

EXAMPLE

```
<PublicationDelivery version="1.0" xsi:schemaLocation="http://www.netex.org.uk/netex /xml/NeTEx_publication.xsd" xmlns="http://www.netex.org.uk/netex"
xmlns:siri="http://www.siri.org.uk/siri" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:foo="http://www.foo.eu/stuff"
xmlns:bar="http://www.bar.de/"
xmlns:fum="http://www.fum.ch/">
    <PublicationTimestamp>2010-12-17T09:30:47.0Z</PublicationTimestamp>
    <dataObjects>
        <CompositeFrame version="1" id="bar:CF01">
                <ServiceFrame version="1" id="bar:SF01">
        <Name>Stops for Winter timetable for route 24 </Name>
        <routePoints>
            <RoutePoint version="any" id="bar:01">
                <Name>Point in Road near Stop A</Name>
                <Location>
                    <Longitude>53.0000</Longitude>
                    <Latitude>0.1000</Latitude>
                </Location>
            </RoutePoint>
            <RoutePoint version="any" id="bar:02">
                <Name>Point in Road near Stop B</Name>
                <Location>
                    <Longitude>53.0000</Longitude>
                    <Latitude>0.2000</Latitude>
                </Location>
            </RoutePoint>
            <RoutePoint version="any" id="fum:01">
                <Name>Point in Road near Stop C</Name>
                <Location>
                    <Longitude>53.0000</Longitude>
                    <Latitude>0.3000</Latitude>
```

```
            </Location>
        </RoutePoint>

    </routePoints>
            <routeLinks>
        <!--- ====== ROUTE LINKs ==OUTBOUND======== -->
        <RouteLink version="any" id="foo:01">
            <FromPointRef version="any" ref="bar:01"/>
            <ToPointRef version="any" ref="bar:02"/>
        </RouteLink>
        <RouteLink version="any" id="foo:02">
            <FromPointRef version="any" ref="bar:02"/>
            <ToPointRef version="any" ref="fum:01"/>
        </RouteLink>
::>
```

7.3.3.4.2 Data Source – XML Example Fragments

The following example shows the hierarchical use of DATA SOURCEs. In the example three different DATA SOURCEs are declared *'nptg:NaPTAN'* for stop data, *'tbd:XtraAccessibilityData'* for accessibility data, and *'mydata:DB770' for* the producer of the delivered frames that contains the assembled data. The outer ***CompositeFrame*** sets *'mydata:DB770'* as the default DATA SOURCE to use if no specific source is provided on an individual element. Each ***StopPlace*** is specifically declared as coming from *'nptg:NaPTAN'.* The accessibility data for each stop however has been added from a different source *'tbd:Xtra_Acc_Data'.*

EXAMPLE

```
<CompositeFrame version="any" id="tbd:cf01" dataSourceRef=" mydata:DB770">
    <ResourceFrame version="any" id="tbd:RS01" >
        <codespaces>
            <Codespace id=" napt ">
                <Xmlns>napt</Xmlns>
                <XmlnsUrl>http://www.naptan.org.uk/naptan</XmlnsUrl>
                <Description>UK NaPTAN Stop Place codes</Description>
            </Codespace>
            <Codespace id="tbd">
                <Xmlns>tbd</Xmlns>
                <XmlnsUrl>http://www.intdata.co.uk/data</XmlnsUrl>
                <Description>Other interchange DATA SOURCE </Description>
            </Codespace>
        </codespaces>
        <dataSources>
            <DataSource version="any" id="nptg:NaPTAN" dataSourceRef="nptg:NaPTAN" >
                <Name>DfT Naptan database </Name>
            </DataSource>
            <DataSource version="any" id="tbd:Xtra_Acc_Data" dataSourceRef=" tbd:Xtra_Acc_Data " >
                <Name>Additional accessibility Data </Name>
            </DataSource>
            <DataSource version="any" id="mydata:DB770" dataSourceRef=" mydata:DB770">>
                <Name>Wigan local data base </Name>
            </DataSource>
        </dataSources>
    </ResourceFrame>
::>….
    <SiteFrame version="any" id="tbd:SiteFrame:SVC01">
        <stopPlaces>
            <StopPlace  version="001"  created="2006-09-11T15:42:00"  changed="2009-02-26T15:47:00
modification="revise"
                    dataSourceRef="nptg:NaPTAN" id="napt:910GWIMBLDN">
                <Name>Wimbledon Rail Station</Name>
                <ShortName>Wimbledon Station</ShortName>
                <types><TypeOfZoneRef ref="napt:GRLS"/></types>
                <AccessibilityAssessment version="any" id="tbd:910GWIMBLDN_Ac1"
                    dataSourceRef=" tbd:Xtra_Acc_Data " >
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation modification="new" dataSourceRef="tbd:Xtra_Acc_Data"
                    created="2010-05-17T09:30:47Z">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
```

```
            </StopPlace>
::>.
        <responsibilitySets>
            <!-- Generalisation of NPTG ability to associate data with an area and Equivalent to
NPTG areas = -->
            <ResponsibilitySet version="001" created="2010-05-17T09:30:47Z" modification="revise"
changed="2010-05-17T09:30:47Z" responsibilitySetRef="napt:RS_nptg" id="napt:RS_82">
                <roles>
                    <ResponsibilityRoleAssignment version="001" created="2010-05-17T09:30:47Z"
id="napt:RS_82_01">
                        <Description>490 London - Managed centrally</Description>
                        <DataRoleType>collects</DataRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TL001"/>
                        <ResponsibleAreaRef version="any" ref="napt:82"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
            <ResponsibilitySet            version="001"            created="2010-05-17T09:30:47Z"
responsibilitySetRef="napt:RS_nptg" id="napt:RS_110">
                <roles>
                    <ResponsibilityRoleAssignment version="001" created="2010-05-17T09:30:47Z"
id="napt:RS_110_01">
                        <Description>910 UK Rail - Managed centrally</Description>
                        <DataRoleType>collects</DataRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TD002"/>
                        <ResponsibleAreaRef version="any" ref="nptg:110"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
            <ResponsibilitySet            version="001"            created="2010-05-17T09:30:47Z"
responsibilitySetRef="napt:RS_147" id="napt:RS_147">
                <roles>
                    <ResponsibilityRoleAssignment version="001" created="2010-05-17T09:30:47Z"
id="napt:RS_147_01">
                        <Description>940 UK Tram - Managed centrally</Description>
                        <DataRoleType>collects</DataRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TL001"/>
                        <ResponsibleAreaRef version="any" ref="nptg:147"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
            <ResponsibilitySet            version="001"            created="2010-05-17T09:30:47Z"
responsibilitySetRef="napt:RS_nptg" id="napt:RS_nptg">
                <roles>
                    <ResponsibilityRoleAssignment version="001" created="2010-05-17T09:30:47Z"
id="napt:RS_nptg">
                        <Description>NPTG Coordination of area codes UK London</Description>
                        <DataRoleType>collects validates distributes</DataRoleType>
                        <StakeholderRoleType>Operation</StakeholderRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TD002"/>
                        <ResponsibleAreaRef version="any" ref="nptg:UK"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
            <ResponsibilitySet            version="001"            created="2010-05-17T09:30:47Z"
responsibilitySetRef="tbd:RS_123" id="napt:RS_de">
                <roles>
                    <ResponsibilityRoleAssignment version="001" created="2010-05-17T09:30:47Z"
id="napt:RS_123_01">
                        <Description>Extra STOP PLACE data </Description>
                        <DataRoleType>all</DataRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TD002"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
        </responsibilitySets>
```

7.3.4 Generic Version

The sections below describe the versioning mechanism for objects exchanged with NeTEx.

NOTE The modelling of versions in NeTEx is designed to fulfil the versioning needs of a data exchange mechanism exchanging complex linked objects. It is in effect a version description model, not a model of a version management system. However, it is allows for fine grained versioning, and uses a uniform and generic approach that can be used for any time of complex data object. This versioning mechanism is available on all NeTEx elements, but not mandatory thus allowing legacy systems without any versioning mechanism to use NeTEx simply by omitting the versioning attributes. In practice versiong will often just done at an aggregate level and not that of the indviudal data instance.

Public transport data are in a permanent process of evolution; schedule and operational data typically undergoes a regular cycle of planning, distribution and execution, whilst reference data describing the network, such as stop and line data, will change if network or physical environment is modified. It is therefore necessary to be able to organise data elements to support such a lifecycle, with multiple versions of a given element being in use concurrently, and different assemblies of data referencing different versions for different purposes. This is achieved in NeTEx with VERSIONs and VERSION FRAMEs.

7.3.4.1 Generic VERSION – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

Each state of an object, or a set of objects, is called a VERSION. VERSIONs of an object may be consecutive or competitive. *Consecutive* VERSIONs describe the successive states of an object, whilst *competitive* VERSIONs describe an alternative version to use in particular circumstances (as say described by a VALIDITY CONDITION). For example, there may be for a single LINE at the same time competitive versions of the officially scheduled LINE; a simulated LINE (for planning work or for study), and the LINE with alternative ROUTEs (for disruption management for example).

The VERSION describes the identifier and purpose of a version state. The actual version state is described by an instance of ENTITY IN VERSION. Thus in a given repository or documents there will be a single instance of each Transmodel ENTITY and one or multiple instances of ENTITY IN VERSIONs for that ENTITY; these will be tried together by a common identifier and differentiated by distinct VERSION identifiers. For example an instance of the entity VEHICLE JOURNEY may have multiple VEHICLE JOURNEY IN VERSION instances, a STOP PLACE may have multiple STOP PLACE IN VERSION instances, etc.

The purpose of the VERSION may be categorised with an arbitrary classification using a TYPE OF VERSION, for example planning, scheduled, operational, etc.

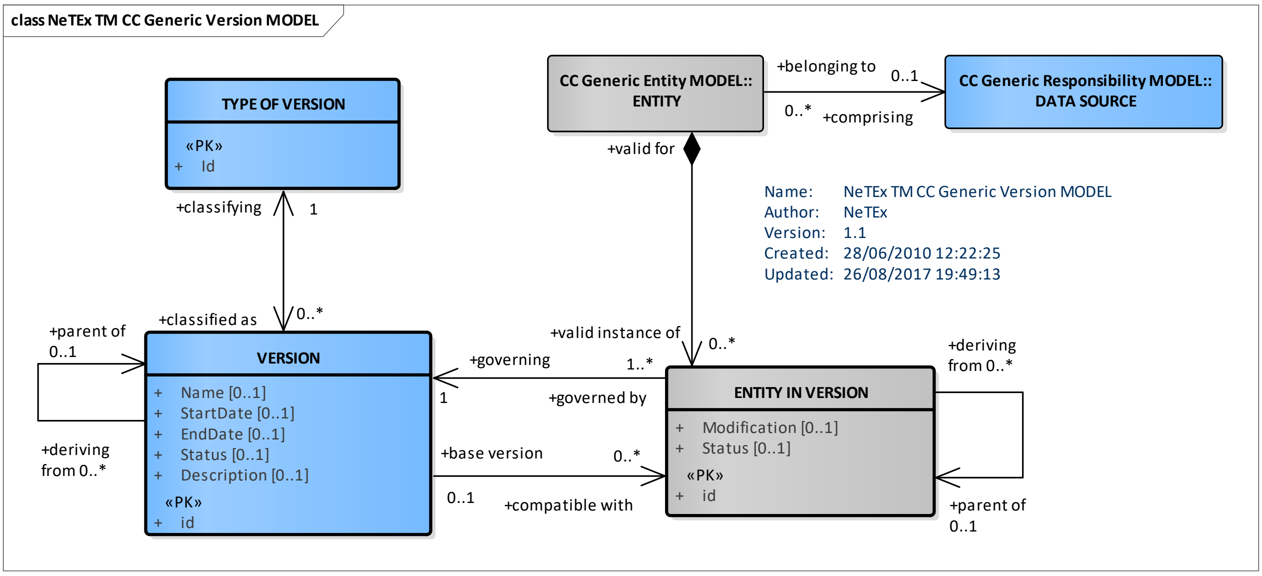


Figure 42 — Generic Version – Conceptual MODEL (UML)

7.3.4.2 Generic Version – Physical Model

The physical Generic Version Model has classes corresponding to those of the conceptual model. In the physical model ENTITY IN VERSION is abstract; no instance of ***EntityInVersion*** is possible, instead there is a specific subclass of it for each type of concrete NeTEx ENTITY that is exchanged. For example STOP PLACE IN VERSION, VEHICLE JOURNEY IN VERSION, etc. For brevity the 'IN VERSION' is ommitted from the names of model elements. Thus ***Place*** is short for ***PlaceInVersion***, ***VehicleJourney*** is short for ***VehicleJourneyInVersion***, etc

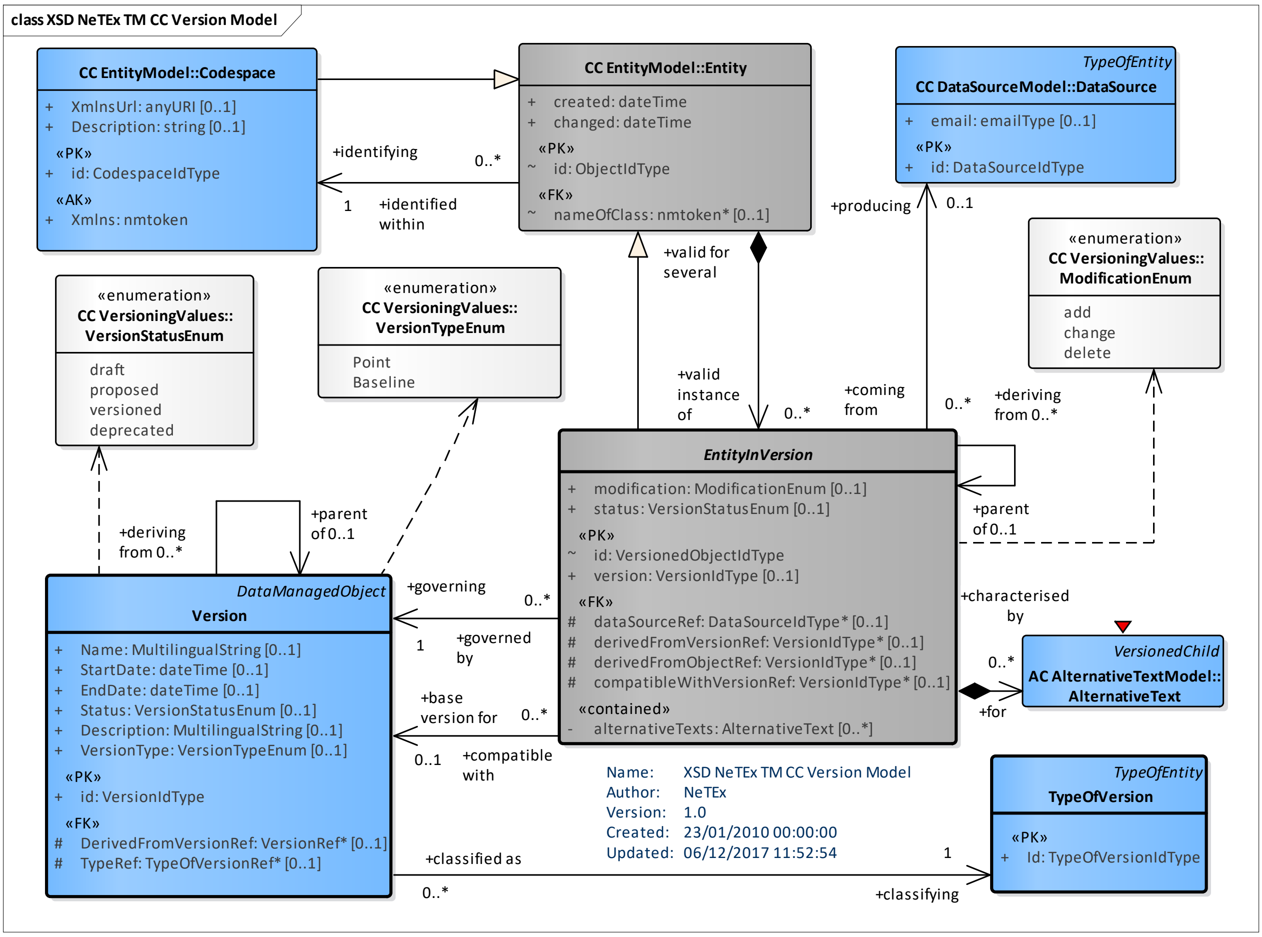


Figure 43 — Generic Version – Physical Model (UML)

7.3.4.2.1 Naming Entities & Entities in Version in the NeTEx Physical Model

IMPORTANT NOTE Data exported to an XML document from a repository represents a snapshot of the state of a particular version of the data at a particular moment in time. ***The NeTEx XSD is therefore primarily concerned with instances of ENTITY IN VERSION.*** That is to say, data elements in a NeTEx XML document represent a particular version of each ENTITY present and not a simple unversioned 'current' instance of the ENTITY. Even if a basic implementation of a repository holds only a representation of the current state of an ENTITY, rather than its entire history of versions, every time it makes an export it is in effect creating an ENTITY IN VERSION for the time of export; if the current version is changed, two successive exports would result in different states, i.e. different instances of the appropriate ENTITY IN VERSION.

Accordingly, the NeTEx elements of interest in the Physical Model are actually specialisations of ENTITY IN VERSION, not of ENTITY, but for brevity and simplicity we follow a naming convention such that the physical model element for a specific ENTITY IN VERSION is called simply the name of the corresponding ENTITY. Thus for example, the physical element for VEHICLE JOURNEY IN VERSION is called simply ***VehicleJourney***, TIMING LINK IN VERSION is called ***TimingLink***, etc.

In the NeTEx Physical model, all attributes specific to an entity type are held on the respective specialisation of ***EntityInVersion***. Attributes common to several Entities are held on an abstract supertype. NeTEx Physical model and schema do not actually include separate subclasses of each ENTITY (***Line_Entity***, ***VehicleJourney_Entity***, etc.). There is however a single concrete XML ***Entity _Entity*** element that may be used to group versions of a given elements of any type within a given NeTEx document but does not otherwise hold data attributes.

NOTE W3C XSD schema do not support generic types so the *Entity_Entity* is a workaround used to achieve the equivalent mechanism of a container for ENTITY IN VERSION instances of a specific ENTITY type. Although the NeTEx physical model and XSD schema do not include specific elements for each specialisation of ENTITY, but if they did the names of such classes would by convention be qualified with "_ENTITY" in order to distinguish them from the corresponding ENTITY IN VERSION. For example ***VehicleJourneyEntity, VehicleJourney_Entity, TimingLink_Entity,*** etc**.** as shown in the following table.

Table 14 — Examples of ENTITY & ENTITY IN VERSION Names

ENTITY			***ENTITY IN VERSION***	
Name in Conceptual Model	***Name in Physical Model *****		***Name in Conceptual Model***	***Name in Physical Model***
ENTITY	***Entity_Entity***		*ENTITY IN VERSION*	***EntityInVersion***
[XXX[	***Xxx_Entity***		*[XXX] IN VERSION*	***Xxx***
LINE	***Line_Entity***		*LINE IN VERSION*	***Line***
VEHICLE JOURNEY	***VehicleJourney_Entity***		*VEHICLE JOURNEY IN VERSION*	***VehicleJourney***
TIMING LINK	***TimingLinkEntity***		*TIMING LINK IN VERSION*	***TimingLink***
TIMETABLE FRAME	***TimetableFrame_Entity***		*TIMETABLE FRAME IN VERSION*	***TimetableFrame***
Etc.				

7.3.4.2.2 Version States and the Edit / Version / Release Cycle

The NeTEx implementation assumes a conventional Edit / Release / Version / Release lifecycle for versioning each data element within a repository.

— When an element is created or modified a new open version of it is created (status = *draft*). Further changes may be made to the state of the object within the local repository but it may not be exported.

— If it is part of a containing aggregation, for example, a Version Frame, then a new open edition of the containing aggregation must also be created (status = *draft*) and the new version of the contained element must be released to the new version of the container.

— Once all changes are complete the element is versioned (status = *versioned*), after which no further changes can be made. Only versioned elements can be exchanged. Both contained and container elements are versioned. Contained elements must be versioned before their container is versioned.

Data instances exported as NeTEx data elements should have states consistent with these rules.

7.3.4.3 Version – Attributes and XSD

7.3.4.3.1 Version – Model Element

A VERSION is a group of operational data instances which share the same VALIDITY CONDITIONs. A version belongs to a unique VERSION FRAME and is characterised by a unique TYPE OF VERSION. E.g. 'NETWORK VERSION for Line 12 starting from 2000-01-01'.

An instance of a VERSION can be used to describe the reason for a change or set of changes and record further details about it. Often however explicit instances of VERSION will not be created, but rather just be implied by the use of version numbers on instances of ENTITies IN VERSION.

Table 15 — *Version* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VERSION inherits from of DATA MANAGED OBJECT
«PK»	***id***	*VersionIdType*	1:1	Identifier of VERSION.
	StartDate	*xsd:dateTime*	0:1	Start date of validity of VERSION.
	EndDate	*xsd:dateTime*	0:1	End date of validity of VERSION.
«enum»	***Status***	*VersionStatusEnum*	0:1	Status of VERSION. See allowed values below.
	Description	*MultilingualString*	0:1	Description of VERSION.
«enum»	***VersionType***	*VersionTypeEnum*	0:1	Fixed version type. See allowed values below.
«FK»	***TypeOfVersionRef***	*TypeOfVersionRef*	0:1	Reference to an open TYPE OF VERSION. +v1.1
«FK»	***DerivedFromVersionRef***	*VersionRef*	0:1	Reference to a VERSION from which this VERSION is derived. The ***VersionRef*** type holds a ***Ref*** being of ***VersionIdType***, thus being a version number.

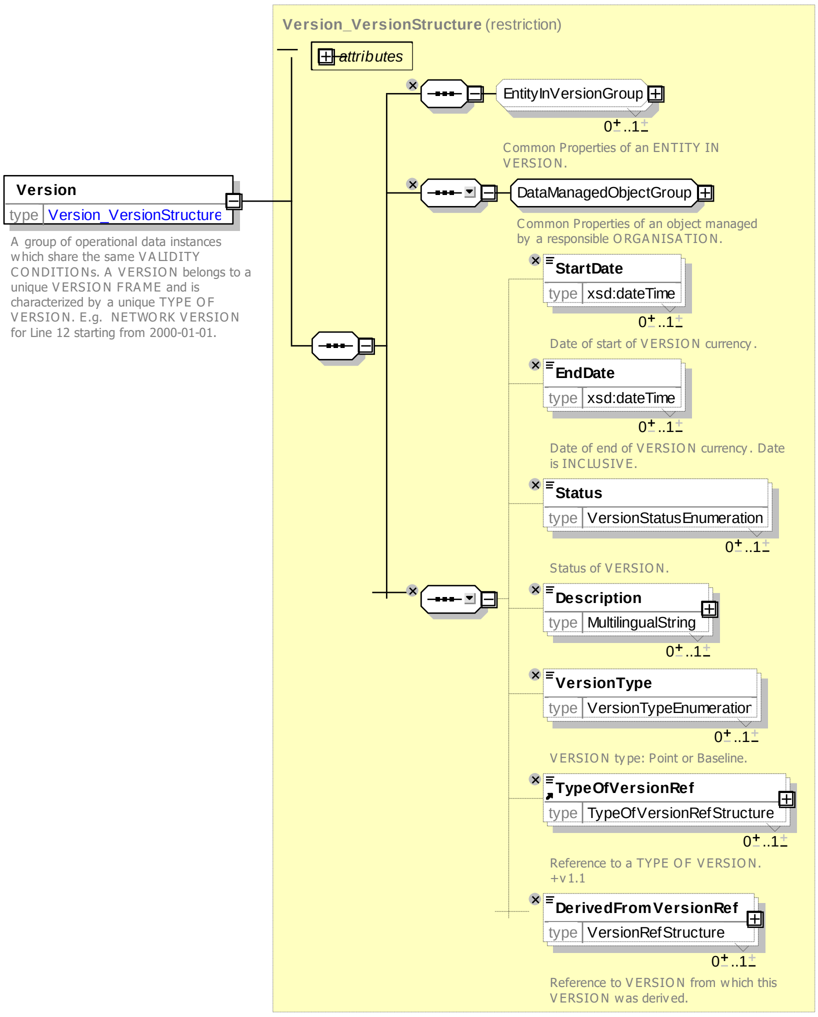


Figure 44 — *Version – XSD*

7.3.4.3.1.1 VersionType – Allowed Values

The following table shows the allowed values for ***VersionType (****VersionTypeEnumeration***).**

Table 16 — *VersionType*– Allowed Values

Value	Description
baseline	This version is comprehensive and comprises all the needed objects to use it.
point	This version is an update of a baseline version and may only contain a subset of the object from the baseline.

7.3.4.3.2 EntityInVersion – Model Element

An ENTITY IN VERSION is an ENTITY associated with a given VERSION. ENTITY IN VERSION is restricted by ENTITY IN FRAME.

Table 17 — *EntityInVersion* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Entity*	::>	ENTITY ON VERSION inherits from ENTITY.
«FK»	***dataSourceRef***	*DataSourceIdType*	0:1	Data system which originated data instance.

«atr»	***created***	*xsd:dateTime*	0:1	Date and time of creation of ENTITY.
«atr»	***changed***	*xsd:dateTime*	0:1	Date and time of last change to ENTITY.
«atr», «enum»	***modification***	*ModificationEnum*	0:1	Nature of modification. Enumerated value: new | revise | delete | unchanged | delta.
«atr». «FK»	***version***	*VersionIdType*	0:1	VERSION number of this instance of the ENTITY IN VERSION.
«atr», «enum»	***status***	*VersionStatusEnum*	0:1	Status of Entity in VERSION. Enumerated value.
«atr», «FK»	***derivedFrom-VersionRef***	*VersionIdType*	0:1	Reference to VERSION from which this VERSION of the ENTITY was derived. This is a *VersionIdType,* the same type as vesion, thus being a version number.
«atr», «FK»	***compatibleWithVersionFrameVersionRef***	*VersionIdType*	0:1	Version of frame from with which this version of ENTITY is compatible. Assumes Frame of the same Id as current frame.
«atr», «FK»	***derivedFromObjectRef***	*ObjectIdType*	0:1	Identity of object from which this object of ENTITY was derived. Normally the same.
		CHOICE	0:1	Common Properties of an ENTITY IN VERSION.
«cntd»	***a*** ***validityConditions***	*ValidityCondition*	1:1	VALIDITY CONDITIONs conditioning entity.
«cntd»	***b*** ***ValidBetween***	*ValidBetween*	1:*	OPTIMISATION. Simple version of a VALIDITY CONDITION. Comprises a simple period. NO UNIQUENESS CONSTRAINT.
«cntd»	***alternativeTexts***	*AlternativeText*	0:*	ALTERNATIVE TEXTs associated with ENTITY.

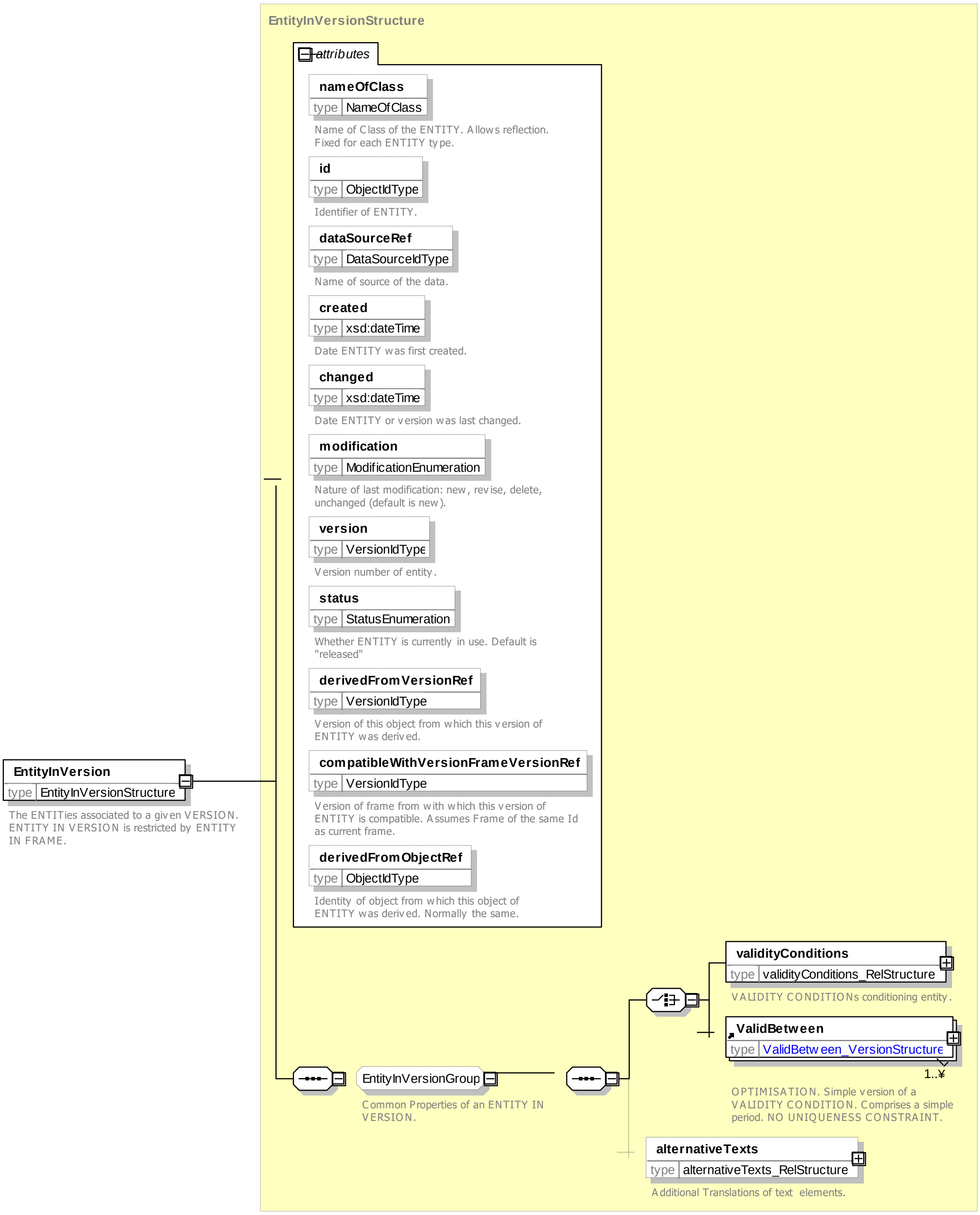


Figure 45 — *EntityInVersion* – XSD

7.3.4.3.2.1 Modification – Allowed Values

Modification explains why this ENTITY version was distributed. The following table shows the allowed values for ***Modification (****ModificationEnumeration****).***

Table 18 — *Modification* – Allowed Values

Value	Description
new	This is the first version of a new object.
revise	This is an update of a previously defined object.
delete	This version is to point out that the object was deleted.
unchanged	This object has not changed since the previous version.
delta	This version contains only the changes to the previous version of the object.

7.3.4.3.2.2 VersionStatus – Allowed Values

The following table shows the allowed values for ***Version / Status (****StatusEnumeration****).***

Table 19 — *VersionStatus* – Allowed Values

Value	Description
draft	This version is a draft: it is currently under construction and should not be used for operational purpose.
proposed	This version is comprehensive but not yet validated.
versioned	This is a finalized, version frozen from further release.
deprecated	This is an old version, and it should not be used anymore for operational purpose.

7.3.4.3.3 EntityInVersionReferences (VersionOfObjectRef)– Model Element

A relationship to another an ENTITY in VERSION is represented by a ***VersionOfObjectRef***, that is, a reference to a specific version of an entity. e.g. ***LineRef***, ***ServiceJourneyRef***. Where the same type of entity is referenced by more than one relationship from an ENTITY, the names of the attributes used to represent the relationship need be qualified, for example a LINK may have both a ***FromPointRef***, and a ***ToPointRef***, to for its start and end POINTs.

Note that the *created* and *changed* attributes of a ***VersionOfObjectRef*** are for the *reference* instance (that is, the ***VersionOfObjectRef*** element itself), **not** for the *referenced* object. If a reference is created, or changed, these should be updated.

Table 20 — *VersionOfObjectRef* – Element

Classification	Name	Type	Cardinality	Description
	NameOfRefClass	*NameOfClass*	0:1	Reference to object type of referenced ENTITY that is in the specified VERSION FRAME
«atr»	***created***	*xsd:dateTime*	0:1	Date and time of creation of reference to ENTITY, i.e. when it was added to frame.
«atr»	***changed***	*xsd:dateTime*	0:1	Date and time of last change to reference to ENTITY.
«FK»	***version***	*VersionRef*	0:1	Reference to version of ENTITY that is in the specified VERSION FRAME. Will be checked by XML validty checker.
«FK»	***ref***	*ObjectIdType*	1:1	Identifier of a referenced ENTITY.
«FK»	***versionRef***	*VersionRef*	0:1	Alternate Reference to version of ENTITY that is in the specified VERSION FRAME. Will not be checked by XML validty checker. +v1.1

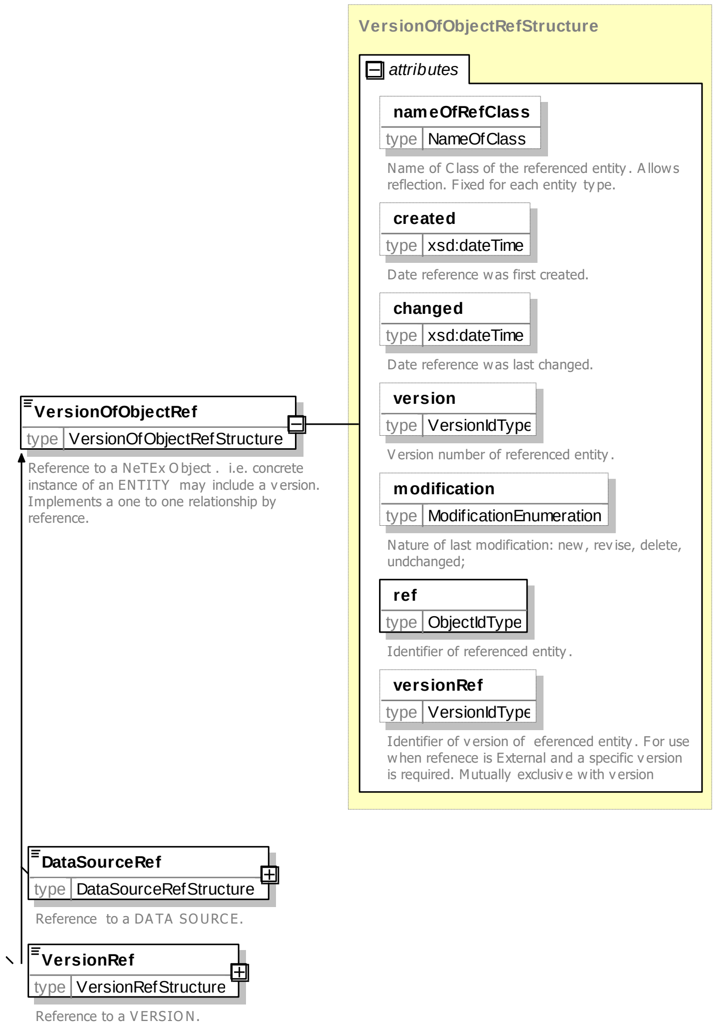


Figure 46 — *EntityInVersionInVersionFrame* (*VersionOfObjectRef*) – *XSD*

7.3.4.3.4 Entity_Entity – Model Element

The ***Entity_Entity*** element provides a general purpose implementation of the ENTITY entity that can be used to group the versions (i.e. ENTITY in VERSIONs) of an ENTITY. All of the child versions should be of the same class as the ENTITY. See below for an example.

Table 21 — *Entity_Entity* – Element

Classification	Name	Type	Cardinality	Description
«atr»	***nameOfClass***	*xsd:NMTOKEN*	0:1	Name of class of which this is an instance. Used for reflection.
«PK»	***id***	*ObjectIdType*	1:1	Identifier of ENTITY. Unique within CODESPACE.

«atr»	***created***	*xsd:dateTime*	0:1	Date and time of creation of ENTITY.
«atr»	***changed***	*xsd:dateTime*	0:1	Date and time of last change to ENTITY.
«cntd»	***versions***	*(EntityInVersion)*	0:*	Versions of the ENTITY. In a given instance of ***Entity_Entity***, The ***EntityInVersion*** instances must all be of the same type as the value of the ***nameOfClass*** attribute. For example if the ***nameOfClass*** is *'ServiceLink'*, then the contained versions must all be instanses of ***ServiceLink***.

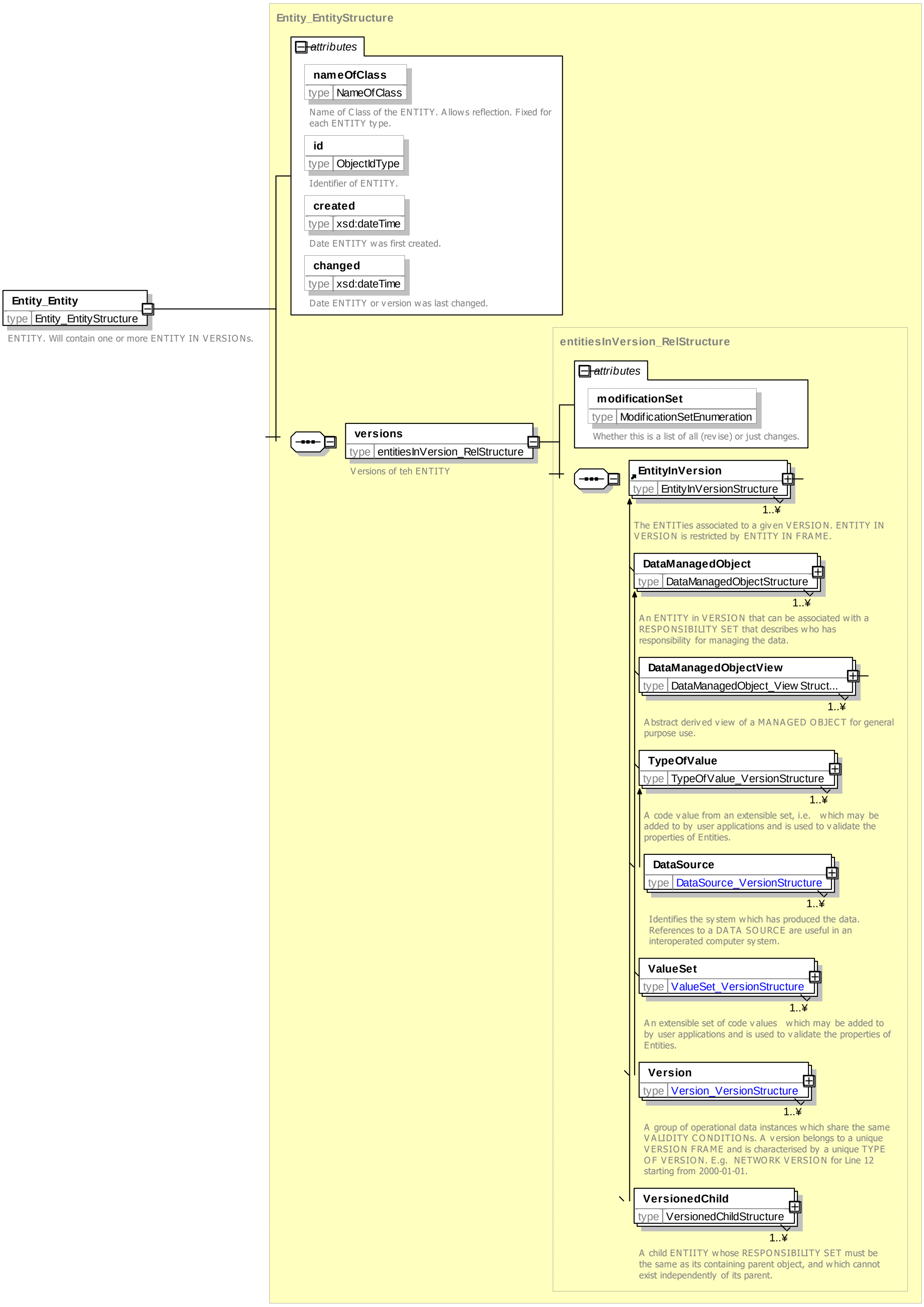


Figure 47 — *Entity_Entity* – XSD

7.3.4.3.5 TypeOfVersions– Model Element

Classification of a VERSION, according to purpose.

Table 22 — *TypeOfVersion* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE OF VERSION inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfVersionIdType*	1:1	Identifier of TYPE OF VERSION.

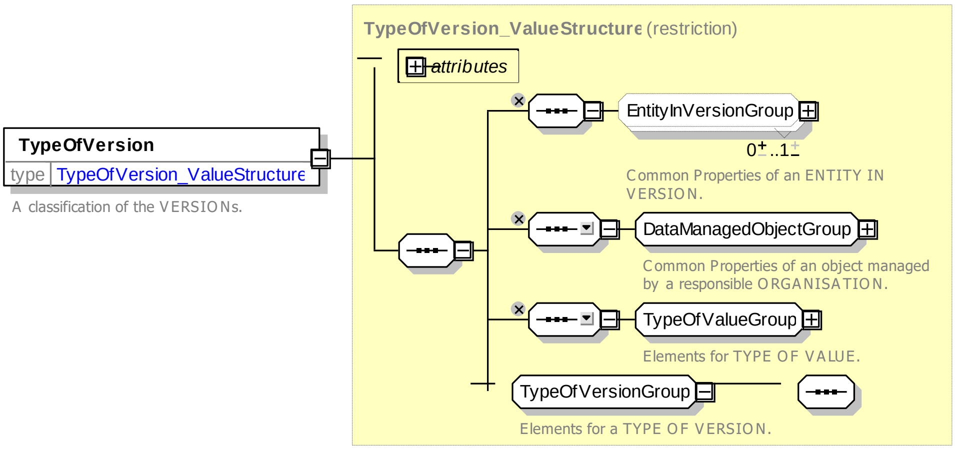


Figure 48 — *TypeOfVersion – XSD*

7.3.4.4 XML Examples of use of Versions

7.3.4.4.1 Version Instances

The following example XML fragment shows the use of explicit version numbers. The example uses separate version numbers for the SCHEDULED STOP POINT (version="022") and the SERVICE FRAME (version="001").

EXAMPLE

```
    <ServiceFrame version="001" created="2010-05-17T09:30:47.0Z" id="mybus:ntwkf001">
        <Name>My Network (V1) </Name>
        <scheduledStopPoints>
            <ScheduledStopPoint  version="022"  created="2010-05-17T09:30:47.0Z"  changed="2011-06-17T09:30:27.0Z" id="mybus:SSP0001A">
                <Name>Haltstelle A</Name>
                <Description>Stop A</Description>
            </ScheduledStopPoint>
        </scheduledStopPoints>
    </ServiceFrame>
```

7.3.4.4.2 Including Version Instances

Information about the VERSIONs can be included along with the actual ENTITY in VERSION instance, for example the following example XML fragment shows the use of VERSION descriptions for the versions of the SCHEDULED STOP POINT and SERVICE FRAME shown in the previous example.

EXAMPLE

```
    <ServiceFrame version="001" created="2010-05-17T09:30:47.0Z" id="mybus:ntwkf001">
        <Name>My Network (V1) </Name>
        <versions>
            <Version created="2010-05-17T09:30:47.0Z" id="mybus:ntwkf001" version="001">
                <Status>versioned</Status>
                <Description>Version 1 of Service Frame </Description>
                <VersionType>baseline </VersionType>
            </Version>
            <Version created="2011-06-17T09:30:27.0Z " version="022">
                <Status>versioned</Status>
                <Description>Version 22 of Stop Point SSP001A </Description>
            </Version>
        </versions>
        <scheduledStopPoints>
            <ScheduledStopPoint  version="022"  created="2010-05-17T09:30:47.0Z"  changed="2011-06-
17T09:30:27.0Z" id="mybus:SSP0001A">
                <Name>Haltstelle A</Name>
                <Description>Stop A</Description>
            </ScheduledStopPoint>
        </scheduledStopPoints>
    </ServiceFrame>
```

7.3.4.4.3 Use of an Entity with Versions of that Entity In Version – XML Example fragment

It is possible for the same document to hold more than one version of the same ENTITY, that is to say multiple instances of a specific ENTITY IN VERSION. The following example XML fragment shows the use of an *Entity_Entity* to group two sets of successive versions (i.e. ENTITY IN VERSIONs) of an ENTITY, one for two versions (‘*001’* and ‘*002*’) of a Scheduled Stop Point (‘*mybus:SSP0001A*’), and the other for two versions of a Service Pattern (‘*007’* and ‘*012*’).

EXAMPLE

```
<GeneralFrame version="any" id="mybus:ServiceFrame:SF1">
    <members>
        <Entity_Entity           created="2010-05-17T09:30:47.0Z"            id="mybus:SSP0001A"
nameOfClass="ScheduledStopPoint">
            <versions>
                <ScheduledStopPoint         version="001"         created="2010-05-17T09:30:47.0Z"
modification="new" " id="mybus:SSP0001A">
        <Name>Haltstelle A</Name>
        <Description>Version one of stop A</Description>
                </ScheduledStopPoint>
                <ScheduledStopPoint  version="002"  created="2010-05-17T09:30:47.0Z"  changed="2010-
05-18T09:30:47.0Z" modification="revise" derivedFromVersionRef="001" id="mybus:SSP0001A">
        <Name>Haltstelle A - Museum</Name>
        <Description>Version two of stop A. Name is Changed and code added</Description>
        <PrivateCode>mycodeA</PrivateCode>
                </ScheduledStopPoint>
            </versions>
        </Entity_Entity>

    ::>::::
        <!-- ===ServicePattern_Entity id="mybus:ServicePattern:SP_001" -->
        <Entity_Entity       created="2010-05-21T10:30:51.0Z"       changed="2010-05-22T10:30:51.0Z"
id="mybus:SP_001" nameOfClass=" ServicePattern ">
            <versions>
                <ServicePattern  version="007"  created="2010-05-21T10:30:51.0Z"  modification="new"
id="mybus:SP_001">
        <Name>From A to C, version 1</Name>
        <pointsInSequence>
            <StopPointInJourneyPattern version="001" id="mybus:SP_001_01" order="1">
                <ScheduledStopPointRef version="002" ref="mybus:SSP0001A"/>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="001" id="mybus:SP_001_02" order="2">
                <ScheduledStopPointRef version="002" ref="mybus:SSP0002B"/>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="001" id="mybus:SP_001_03" order="3">
                <ScheduledStopPointRef version="001" ref="mybus::SSP0003C"/>
            </StopPointInJourneyPattern>
        </pointsInSequence>
                </ServicePattern>
                <ServicePattern  version="012"  created="2010-05-21T10:30:51.0Z"  changed="2010-05-
22T10:30:51.0Z" modification="new" id="mybus:SP_001">
        <Name>From A to D, version 2</Name>
```

```
        <pointsInSequence>
            <StopPointInJourneyPattern version="002" id="mybus:SP_001_01" order="1">
                <ScheduledStopPointRef version="001" ref="mybus:SSP0001A"/>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="002" id="mybus:SP_001_02" order="2">
                <ScheduledStopPointRef version="002" ref="mybus:SSP0002B"/>
                <ForAlighting>true</ForAlighting>
                <ForBoarding>false</ForBoarding>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="002" id="mybus:SP_001_03" order="3">
                <ScheduledStopPointRef version="001" ref="mybus:SSP0003C"/>
            </StopPointInJourneyPattern>
        </pointsInSequence>
                </ServicePattern>
            </versions>
        </Entity_Entity>
    </members>
</GeneralFrame>
```

7.3.5 Implementing relationships in NeTEx

7.3.5.1.1 Entity identifiers and uniqueness

All NeTEx entities can be given a unique identifier, the ***id*** attribute. Identifiers are of type *ObjectIdType.*

Specific entities will have a specific subtype, for example *PointIdType* is a subtype of *ObjectIdType*.

Identifier values should be unique within object type within a document. The NeTEx main schema includes integrity constraints to enforce uniqueness.

7.3.5.1.2 Associations and Entity References

Where an association between two entities in the conceptual model needs to be implemented, normally an element at one end will be chosen in the physical model to hold the reference as a foreign key, i.e concrete attribute that holds the identifier fo the associated object. The choice of end will be according to the semantics of the model. For example *GroupOfEntities* has an association with *PurposeOfGrouping* to specify a purpose for the group*,* and so has an attribute *PurposeOfGroupingRef* which holds a value of *PurposeOfGroupingIdType.*

Most relationships can be versioned, that is a specific version of the referenced entity may be specified on the reference. To implement versioned references a ***VersionOfObJectRef*** is used as the attribute type, specialised to the specific type of the referenced entity, for example ***PointRef, DayTypeRef,*** *etc*. Thus instead of a simple reference to simple type (e.g. *PointIdType)* being held as the referencing attribute, a complex type (a *PointRefStructure*) is used that also specifies a version. For example, the following references version '*001'* of point '*SSP0001A';* `<FromPointRef ref="mybus:SSP0001A" version="001"/>`. See below for further XML examples.

Versioned object references are provided as reusable components for all NeTEx DATA MANAGED OBJECTS. For example, in the following figure ***DataSourceRef*** is a type of ***VersionOfObjectRef*** that holds both a version id and a reference value of type *DataSourceIdType*.

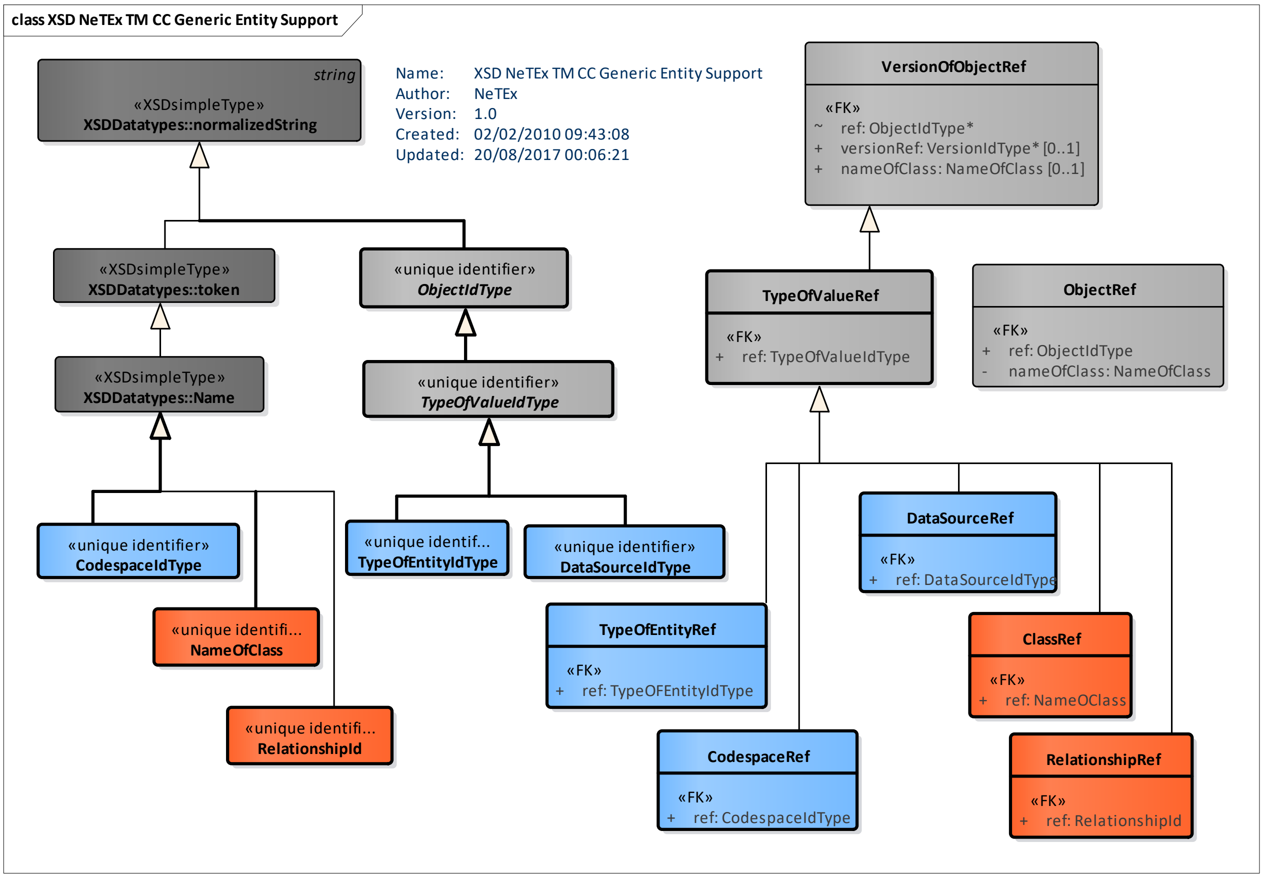


Figure 49 — Entity References – Physical Model (UML)

The version on a reference is optional - if no version is specified integrity checking will be omitted in the XML. A dummy version number can be used to invoke integrity checking against an indeterminate version – this mechanism can be used to ensure that the referenced entity is present in the document regardless of version number. The three levels of enforcement possible are summarised with examples in the following table.

Table 23 — Versioned References

Reference Nature	**Example Reference**	**Example id & version number of referenced entity in same document.**	**Referential Integrity Checking**
Exact version	*<**PointRef** version=”22” ref=”foo:xxx” />*	*<**Point** version=”22” id=”foo:xxx”>*	Yes
Indeterminate version	*<**PointRef** version=”any” ref=”foo:xxx”/>*	*<**Point** version=”any” id=”foo:xxx”*	Yes
No version	*<**PointRef** ref=”foo:xxx”/>*	A instance of Point with identifier foo does not need to be present in the same XML document. If it is present any value is allowed for version.	No

— For an *n-to-one* relationship a simple version reference will be used.

— For an *n-to-many* relationship a set of simple version references will be used, contained within a relationship class. See below.

Generally the physical attribute used to hold the reference on the referencing entity will by convention be named *xxxRef* where xxx is the name of the referenced element, or in some cases a qualified name of the referenced element. For example ***PointRef*** refers to a POINT and uses an instance of *PointIdType;* ***ScheduledStopPoint*** refers to a SCHEDULED STOP POINT and uses an instance of *ScheduledStopPointIdType.* Qualifiers are used when it is necessary or helpful to to distinguish references.For example FromPointRef and ToPointRef on link; or on ***Zone**, **ParentZoneRef*** refers to another ZONE *The attribute name* is qualified with "Parent" to distinguish it from the identifier of the zone itself and has a type of *ZoneRefSructure.*

Certain metadata relationships that are implemented as attributes are not versionable.

— ***Entity*** →dataSourceRef →***DataSource***

— ***Entity*** →responsibilitySetRef → ***ResponsibilitySet***

— **Entity** →version →Version

— ***Entity*** →derivedFromVersionRef → ***Version***

— ***VersionFrame*** →compatibleWIthBaseVersionRef →***Version***

7.3.5.1.3 Relationship metaclasses

The NeTEx relationship classes provide a concrete implementation of the metamodel in NeTEx, making it possible for implementations to reflect over the elements in order to provide automated validation and mapping, etc.

There is a specific relationship class for each one-to-many relationship implemented in NeTEx; all of them are concrete specialisations of the NeTEx framework ***oneToMany*** relationship and by convention are named *xxxx_RelStructure*. For example, for the relationship '*STOP PLACE contains QUAYs*' the ***StopPlace*** element will have a ***quays*** attribute of type *quays_RelStructure.*

Table 24 — NeTEx Schema relationship types

Relationship Type	Description	Example
oneToMany_Rel	The child elements are by reference only	***Organisation***: 0:*→*typeOfOrganisationRefs_Rel:*→ ***TypeOfOrganisationRef***
containmentAggregation_Rel	The child elements may variously be by reference or inline.	***StopPlace***: →*quays_Rel:*→ 0:* ***Quay | QuaysRef***
strictContainmentAggregation_Rel	The child elements must be included in line and there is a compositional semantic	***CompoundTrain*** →*components_Rel:*→ 1:* ***TrainInCompoundTrain***
frameRelationship_Rel	The child elements must be present inline but there is no compositional semantic	***SiteFrame***: →*stopPlacesInFrame_Rel:*→ 0:* ***StopPlace***

The following figure shows the abstract relationship types used in NeTEx.

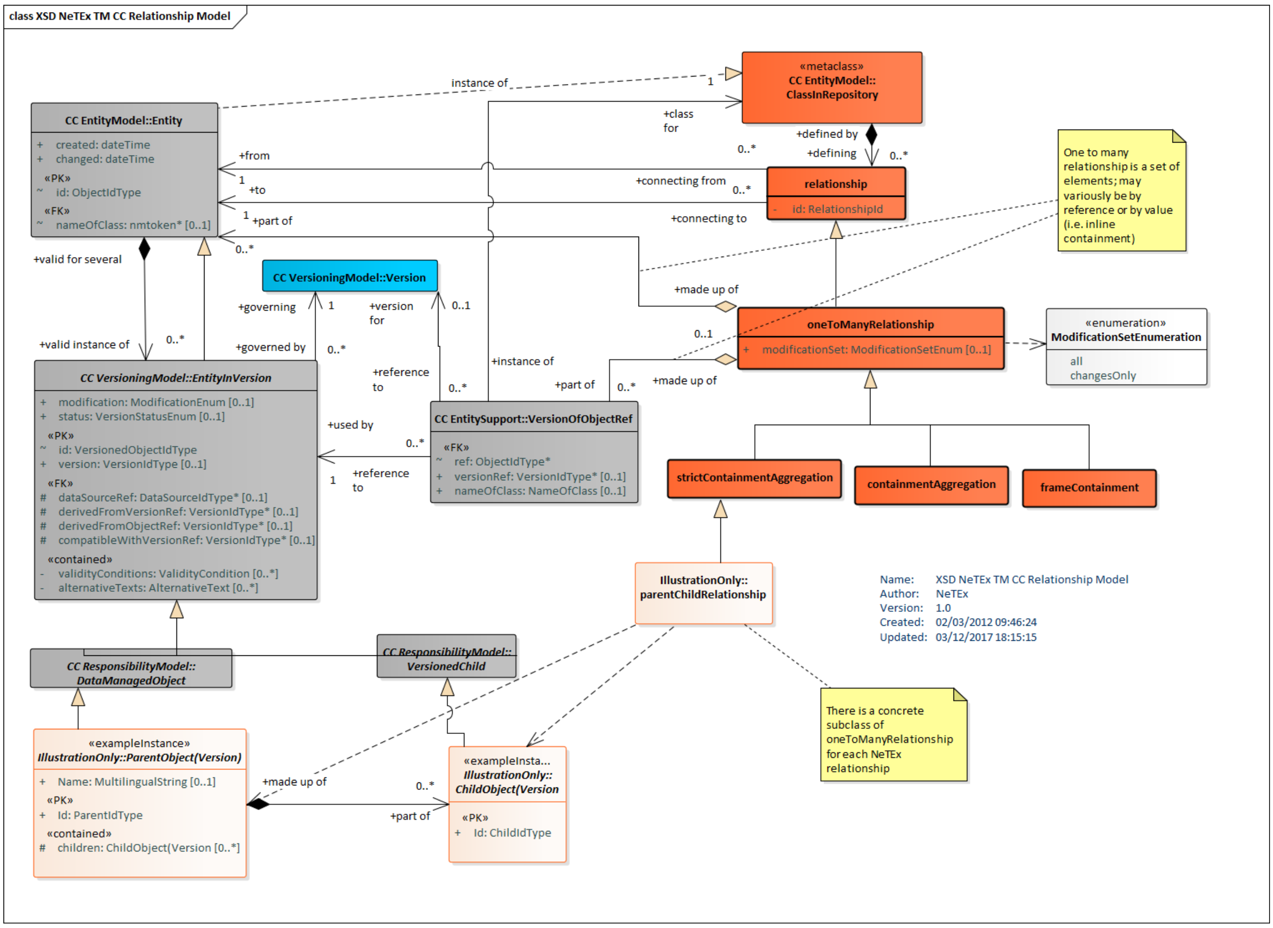


Figure 50 — NeTEx Relationships – Physical Model (UML)

7.3.5.1.4 XML Examples of Relationships

Depending on the use case, an application may need to exchange a complete data set as a single coherent document, or it may exchange static and reference data as separate documents as part of a more complex workflow. In the former case it is desirable to be able to perform automatic integrity checking to ensure that all references ar satisfied; in the latter case, only certain references should be checked. The NeTEx schema supports both approaches by means of the *version* and *versionRef* attributes. Four different scenarios are found.

(i) Full integrity checking of a reference using an actual version.

(ii) Integrity checking of a reference using a dummy version.

(iii) No Integrity Checking.

(iv) No Integrity checking but version cited.

7.3.5.1.4.1 **Use of versions & Version references with Integrity checking – XML fragment**

The following example shows the use of versions and references. The SERVICE LINK is versioned, and connects two SCHEDULED STOP POINTs. It explicitly references which version of each point should be used. The XML will be validated to check that the referenced elements are present in the document.

EXAMPLE

```
<scheduledStopPoints>
        <ScheduledStopPoint   version="001"   created="2010-05-17T09:30:47.0Z"   changed="2010-05-18T09:30:47.0Z" modification="revise" id="mybus:SSP0001A">
            <Name>Haltstelle A - Museum</Name>
            <Description>Version one of stop A. Name is Changed</Description>
            <PrivateCode>mycodeA</PrivateCode>
        </ScheduledStopPoint>
        <ScheduledStopPoint   version="002"   created="2010-05-18T09:30:47.0Z"   changed="2010-06-18T09:30:47.0Z" modification="revise" id="mybus:SSP0002B">
            <Name>Haltstelle B</Name>
            <Description>Version two of stop B</Description>
            <PrivateCode>mycodeB</PrivateCode>
        </ScheduledStopPoint>
</scheduledStopPoints>
<serviceLinks>
        <ServiceLink     version="002"     created="2010-05-19T10:30:47.0Z"     changed="2010-05-21T10:30:51.0Z" modification="revise" id="mybus:SL_AtoB01">
            <Name>Version one of Link from A to B</Name>
            <Distance>1.12</Distance>
            <FromPointRef version="001" ref="mybus:SSP0001A"/>
            <ToPointRef version="002" ref="mybus:SSP0002B"/>
        </ServiceLink>

<serviceLinks>
```

7.3.5.1.4.2 Use of dummy versions & references with Integrity checking – XML fragment

The following example shows the use of dummy versions with references. The SERVICE LINK connects two SCHEDULED STOP POINTs. It explicitly references which version of each point should be used. However a dummy value (by convention ‘any’) is used as the version number. The XML will be validated to check that the referenced elements are present in the document with the dummy version number - but no specifc version is actually intended.

EXAMPLE

```
<scheduledStopPoints>
        <ScheduledStopPoint   version="any"   created="2010-05-17T09:30:47.0Z"   changed="2010-05-18T09:30:47.0Z" modification="revise" id="mybus:SSP0001A">
            <Name>Haltstelle A - Museum</Name>
            <Description>Version two of stop A. Name is Changed</Description>
            <PrivateCode>mycodeA</PrivateCode>
        </ScheduledStopPoint>
        <ScheduledStopPoint   version="any"   created="2010-05-18T09:30:47.0Z"   changed="2010-06-18T09:30:47.0Z" modification="revise" id="mybus:SSP0002B">
            <Name>Haltstelle B</Name>
            <Description>Version two of stop B</Description>
            <PrivateCode>mycodeB</PrivateCode>
        </ScheduledStopPoint>
</scheduledStopPoints>
<serviceLinks>
        <ServiceLink     version="002"     created="2010-05-19T10:30:47.0Z"     changed="2010-05-21T10:30:51.0Z" modification="revise" id="mybus:SL_AtoB01">
            <Name>Version one of Link from A to B</Name>
            <Distance>1.12</Distance>
            <FromPointRef version="any" ref="mybus:SSP0001A"/>
            <ToPointRef version="any" ref="mybus:SSP0002B"/>
        </ServiceLink>

<serviceLinks>
```

7.3.5.1.4.3 Use of unversioned references – XML fragment

The following example shows the use of references without a version. The SERVICE LINK is versioned, and connects two SCHEDULED STOP POINTs. However it does not specify which version of each point should be used. The XML will not be validated to check that the referenced elements are present in the document.

EXAMPLE

```xml
<scheduledStopPoints>
      <ScheduledStopPoint version="001" created="2010-05-17T09:30:47.0Z" changed="2010-05-18T09:30:47.0Z" modification="revise" id="mybus:SSP0001A">
         <Name>Haltstelle A - Museum</Name>
         <Description>Version one of stop A. Name is Changed</Description>
         <PrivateCode>mycodeA</PrivateCode>
      </ScheduledStopPoint>
      <ScheduledStopPoint version="002" created="2010-05-18T09:30:47.0Z" changed="2010-06-18T09:30:47.0Z" modification="revise" id="mybus:SSP0002B">
         <Name>Haltstelle B</Name>
         <Description>Version two of stop B</Description>
         <PrivateCode>mycodeB</PrivateCode>
      </ScheduledStopPoint>
</scheduledStopPoints>
<serviceLinks>
      <ServiceLink version="002" created="2010-05-19T10:30:47.0Z" changed="2010-05-21T10:30:51.0Z" modification="revise" id="mybus:SL_AtoB01">
         <Name>Version one of Link from A to B</Name>
         <Distance>1.12</Distance>
         <FromPointRef ref="mybus:SSP0001A"/>
         <ToPointRef ref="mybus:SSP0002B"/>
</ServiceLink>
```

7.3.5.1.4.4 Use of version references without integrity checking – XML fragment

The following example shows the use of references with a version, however the version is merely cited (the *versionRef* keyword rather than *version* is used) and will not be checked by the XML validator (but might still be used by application program code). The SERVICE connects two SCHEDULED STOP POINTs. The XML will not be validated to check that the referenced elements are present in the document.

EXAMPLE

```xml
<scheduledStopPoints>
        <ScheduledStopPoint version="001" created="2010-05-17T09:30:47.0Z" changed="2010-05-18T09:30:47.0Z" modification="revise" id="mybus:SSP0001A">
            <Name>Haltstelle A - Museum</Name>
            <Description>Version two of stop A. Name is Changed</Description>
            <PrivateCode>mycodeA</PrivateCode>
        </ScheduledStopPoint>
        <ScheduledStopPoint version="002" created="2010-05-18T09:30:47.0Z" changed="2010-06-18T09:30:47.0Z" modification="revise" id="mybus:SSP0002B">
            <Name>Haltstelle B</Name>
            <Description>Version two of stop B</Description>
            <PrivateCode>mycodeB</PrivateCode>
        </ScheduledStopPoint>
</scheduledStopPoints>
<serviceLinks>
        <ServiceLink version="002" created="2010-05-19T10:30:47.0Z" changed="2010-05-21T10:30:51.0Z" modification="revise" id="mybus:SL_AtoB01">
            <Name>Version one of Link from A to B</Name>
            <Distance>1.12</Distance>
            <FromPointRef versionRef="001" ref="mybus:SSP0001A"/>
            <ToPointRef versionRef="002" ref="mybus:SSP0002B"/>
        </ServiceLink>

<serviceLinks>
```

7.3.6 Generic Version Frame

7.3.6.1 Generic VERSION FRAME – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

Usually when exchanging data, only a subset of all the possibly available NeTEx classes are present in an exchanged data set. This subset is defined by the VERSION FRAME mechanism. For example, there might be a TIMETABLE FRAME containing all vehicle schedules of '*Line 12*' services, grouping together the LINE, ROUTEs, JOURNEY PATTERNs, SCHEDULED STOP POINTS, VEHICLE JOURNEYs, etc. for a specific validity period.

VERSION FRAMEs allow data to be managed and exchanged as a coherent version, that is a set of instances of different entity types that are consistent and correct as to referential integrity and other business semantics and so are suitable for use without extensive consistency checking by the importing appplication. A VERSION FRAME contains a list of specific versions of an entity, that is, instances of ENTITY IN VERSION.

To be useful a data set must be both ***correct*** (satisfy integrity checks that all required references are satisfied and all types correct) and ***complete*** (all necessary elements must be present). Note that this does not imply that a single VERSION FRAME must contain all the elements that are referenced in that VERSION FRAME. A VERSION FRAME can refer to external elements defined in other VERSION FRAMEs. If *versioned* references are used then these other VERSION FRAMES must be present in the same XML document. However, if *unversioned* references are used then the other VERSION FRAMES do not have to be present at all and it is assumed they are exchanged separately. In practice different use cases require different sets of data to be assembled and have different criteria for completeness.

A VERSION FRAME may pe a prerequisite for another VERSION FRAME.

7.3.6.1.1 Types of Frame – Conceptual MODEL

The possibilities for including specific types of ENTITies in VERSION in a FRAME are limited by the generic rules set by a corresponding CLASS IN FRAME. All the classes that are allowed to be present in the frame are defined by the CLASS IN FRAME, and each frame is defined by its TYPE OF FRAME. Note that validation to check that only permitted classes are present has to be carried out by the importing application and cannot be done by XML validation alone.

VERSION FRAMEs may have common properties as regards validity. This is described by the TYPE OF FRAME entity (e.g. vehicle schedules, network description for line versions, etc.). The main property of a TYPE OF FRAME is the purpose it is designed for.

A TYPE OF FRAME may be associated with a particular TYPE OF VALIDITY, which expresses a general validity environment. The TYPE OF VALIDITY will apply to any VERSION FRAMEs of that type. For instance, if the schedules designed for day types are to be distinguished from schedules planned for a particular operating day, different TYPEs OF VALIDITY, which will serve as a basis to select general validity rules, may specify this difference. Similarly, certain VERSION FRAMEs may be designed only for simulation purposes and be distinguished from production data, this classification being expressed with a different TYPE OF VALIDITY;

A particular rule (filter) for selecting VERSIONs may be attached to a TYPE OF FRAME, in particular in situations where several VERSIONs of the same VERSION FRAME may be in competition at the same time. The valid VERSION in a given context will be chosen by application of a specific request filters, comparing this context with parameter values. For instance, production vehicle schedules will be selected using the filters which state the PROPERTies OF DAY to be considered by an algorithm, whereas simulation schedules may be selected using another filter. This aspect is useful when rules for selecting versions are more complex than the application of a simple criterion such as “the most recent valid version”.

A TYPE OF FRAME may include other TYPEs OF FRAME, for which the validity rules and processes may be different. This is represented by a circular relationship on TYPE OF FRAME.

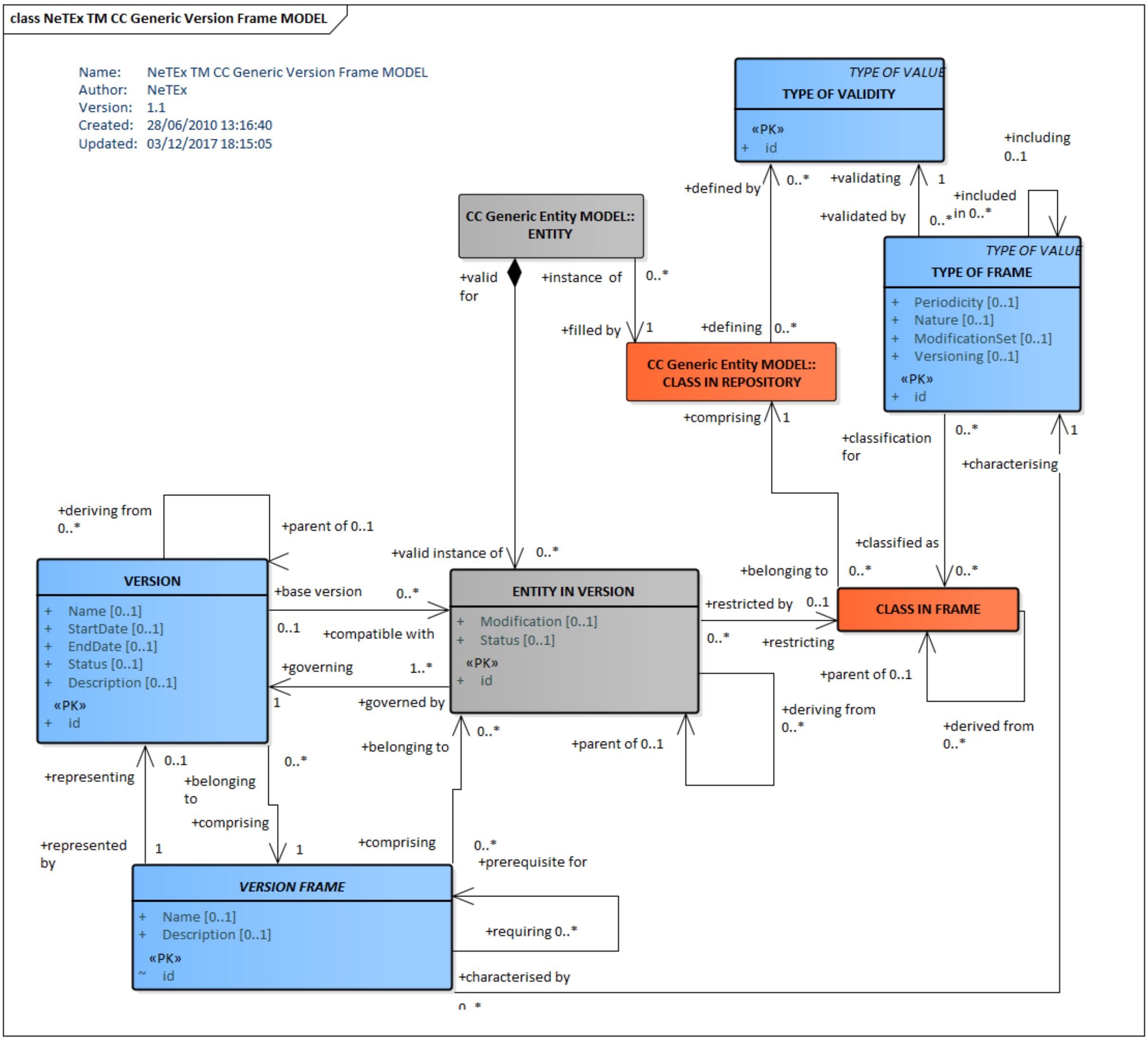


Figure 51 — Generic Version Frame – Conceptual MODEL (UML)

A particular VERSION FRAME, defined according to a TYPE OF FRAME, is usually limited by operational parameters: For example VERSION FRAME for network description of “*area West*”, for vehicle schedule parameters in “*depot D*”, for fare versions on “*tramway lines*”, etc. When these limiting parameters are actual instances of entities stored in the database, this may be described by instances of entity VALIDITY CONDITION, related to the VERSION FRAME. For instance, a VALIDITY CONDITION may represent a DEPOT, a LINE or a GROUP OF LINEs, a particular OPERATOR, etc.

7.3.6.1.2 Version Frames and Versioning

The VERSION FRAME itself is versioned, so that if any change is made to the contents of a frame to add, change or delete its entities, then a new version of the frame must be created; see 'Version States' above. Only versioned Frames (i.e. with status of ‘*version*’) may be exchanged;. Typically in a given repository a new version will be created with a status of ‘draft’, be worked on locally till complete and consistent, and then be ‘frozen’ and versioned for export.

For a defined group of object instances, there may be several (consecutive or competitive) VERSIONs of a VERSION FRAME. For example, the contents of a frame containing the SCHEDULED STOP POINTs for a town will change as they are added, updated or deleted, so there will be successive versions of the same frame), or there may be successive instances of a given TIMETABLE FRAME reflecting successive changes to a given schedule. In Summary:

— A given aggregation may undergo successive versions as the data evolves through its lifecycle, so there may be several *consecutive* VERSIONs of a VERSION FRAME.

— A given aggregation may represent an alternative to be used in particular conditions, so there may be several *competitive* VERSIONs of a VERSION FRAME in which case a VALIDITY condition must be attached to the frame to discriminate the conditions for use.

7.3.6.1.3 General Frames and Explicit Frames

In the original Transmodel model the required contents of a given frame can be specified dynamically (i.e. at run time) using the CLASS IN FRAME mechanism which in effect provides metadata that an importing program can use to check that all of the necessary elements are present. Such a dynamic validation mechanism cannot be enforced in XML and so to obtain the benefits of XML's built in type checking in NeTEx, the general frame mechanism is complemented by a more specific set of "Explicit" VERSION FRAMES that specify sets of data elements appropriate for a particular use case or set of related use cases; for example, INFRASTRUCTURE FRAME, SITE FRAME, TIMETABLE FRAME, etc. Each of these represents a predefined combination of data types that are commonly exchanged together as part of the data management processes of organisations concerned with transport data.

Sometimes data elements from more than one type of explicit frame is needed; a COMPOSITE FRAME can be used to group a coherent set of explicit frames. A given frame may indicate which other frames are prerequisites.

The explicit frames correspond to various parts of NeTEx and in most cases are described in the appropriate section along with their contents. In most cases a given NETEX element appears only in one explicit frame. For example, INFRASTRUCTURE LINK is found only in the INFRASTRUCTURE FRAME. The following table summarises the frames.

Table 25 — Explicit Frames

	Name	Primary contents
Framework	***CompositeFrame***	Frame to group other VERSION FRAMEs.
	GeneralFrame	Any ENTITY or ENTITY IN VERSION.
	ResourceFrame	General purpose components such as ORGANISATIONs, VEHICLE TYPEs and code values.
Part1	***InfrastructureFrame***	INFRASTRUCTURE POINTs, LINKs & RESTRICTIONs.
	SiteFrame	SITESs, STOP PLACEs, POINTS OF INTEREST and other fixed objects.
	ServiceFrame	Network description elements such as LINEs, ROUTEs, etc. Tactical Planning elements such as SCHEDULED STOP POINTs, JOURNEY PATTERNs, etc. pattern.
Part2	***TimetableFrame***	Timetable elements : SERVICE JOURNEYs with timings.
	VehicleScheduleFrame	VEHICLE SCHEDULEs: BLOCKs and BLOCK PARTs.
	DriverScheduleFrame	DRIVER SCHEDULEs: DUTies and DUTY plans.
Part3	***FareFrame***	Fare related elements: TARIFF STRUCTUREs, FARE PRODUCTs, FARE PRICEs, SALES OFFER PACKAGES, etc.

	SalesTransactionFrame	Data generated by the sale of fare products: CUSTOMERS, CUSTOMER ACCOUNTS, FARE CONTRACTs, SALES TRANSACTIONs, TRAVEL DOCUMENTsetc.

VersionFrame itself is abstract and cannot be used in an XML document, but a concrete general purpose frame ***GeneralFrame*** is provided that can contain any type of ENTITY or ENTITY in VERSION. This can be used to handle any ad-hoc assembly of data.

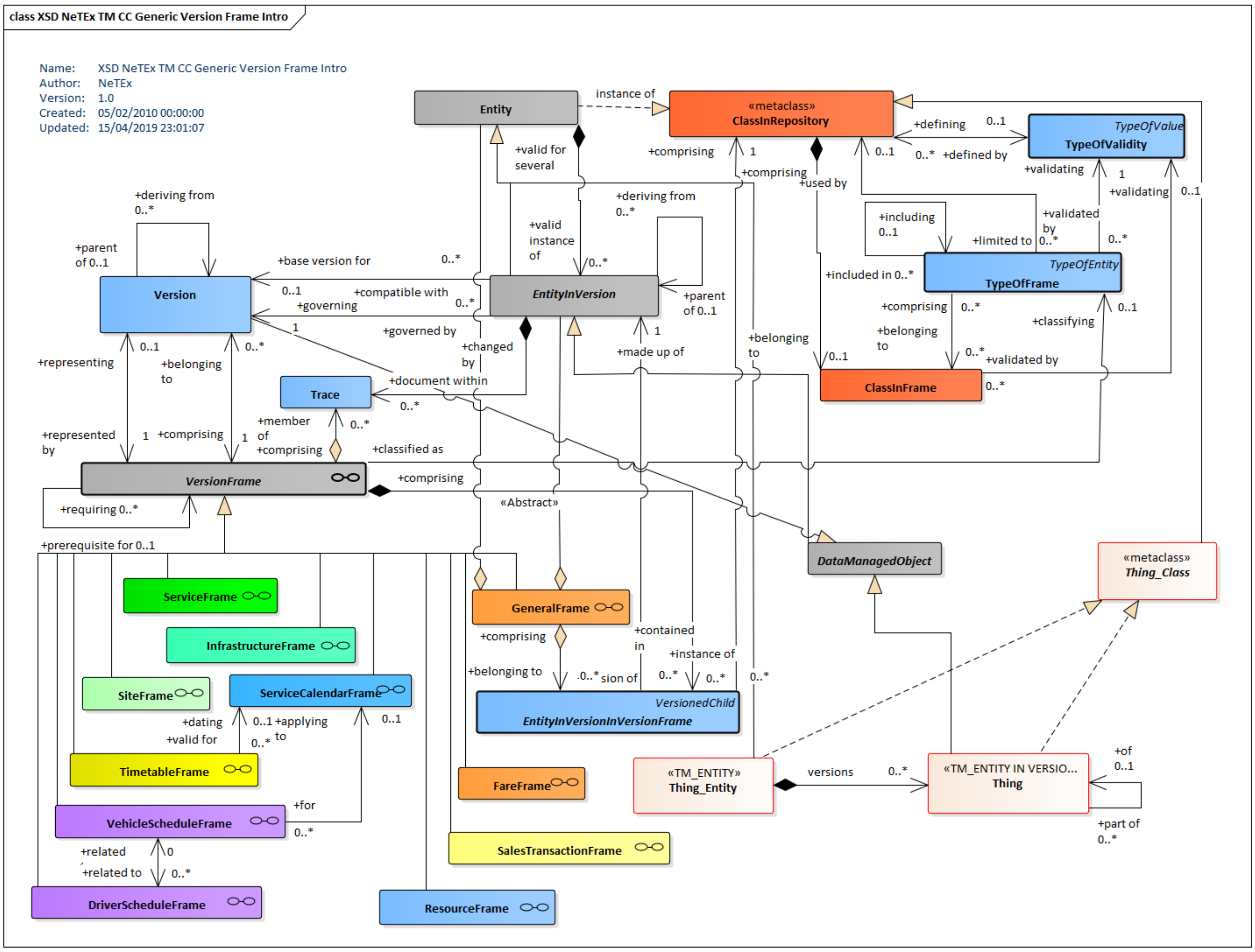


Figure 52 — Explicit Version Frames – Overview (UML)

7.3.6.1.4 Using Version Frames to Requests

In NeTEx a request and filtering mechanism (at XSD level) is used to specify the data to be returned. The contents are then returned in a VERSION FRAME. The type of frame depends on the request. The data to be requested can be specified in several different ways as shown in the following table

The filtering mechanism takes into account the VALIDITY CONDITIONs attached to the data. This means that, for example, the same NeTEx request may provide different results, depending on when the request is submitted and what filter criteria are specified.

Table 26 — Requests & Version Frames

Request Type	Description	Results Returned

Version Frame + Version	Specific version of a specific known version frame. E.g. Version 002 of ***TimetableFrame*** ABC001.	Exact requested version of Version frame returned.
Version Frame + Validity Criteria	Specific known version frame that meets some validity criteria. E.g. ***TimetableFrame*** ABC001 for Summer 2013.	Instance of version frame that meets validity condition.
Scope Data criteria _ Validity Criteria	Criteria for selecting data. E.g. "Stops for the Munich Area, now"".	Ad hoc version frames returned to hold matching data.

7.3.6.1.5 Types of Version frame

VERSION FRAMEs may have common properties as regards validity. This is described by the TYPE OF FRAME entity (e.g. vehicle schedules, network description for line versions, etc.). The main property of a TYPE OF FRAME is the purpose for which it is designed.

A TYPE OF FRAME may be associated with a particular TYPE OF VALIDITY, which expresses a general validity environment. The TYPE OF VALIDITY will apply to any VERSION FRAMEs of that type. For instance, if the schedules designed for day types are to be distinguished from schedules planned for a particular operating day, different TYPEs OF VALIDITY, which will serve as a basis to select general validity rules, may specify this difference. Similarly, certain VERSION FRAMEs may be designed only for simulation purposes and be distinguished from production data, this classification being expressed with a different TYPE OF VALIDITY.

A particular rule (filter) for selecting VERSIONs may be attached to a TYPE OF FRAME, in particular in situations where several VERSIONs of the same VERSION FRAME may be in competition at the same time. The valid VERSION in a given context will be chosen by application of a specific request filters, comparing this context with parameter values. For instance, production vehicle schedules will be selected using the filters which state the PROPERTies OF DAY to be considered by an algorithm, whereas simulation schedules may be selected using another filter. This aspect is useful when rules for selecting versions are more complex than the application of a simple criterion such as "the most recent valid version".

A TYPE OF FRAME may include other TYPEs OF FRAME, for which the validity rules and processes may be different. This is represented by a circular relationship on TYPE OF FRAME.

A particular VERSION FRAME, defined according to a TYPE OF FRAME, is usually limited by operational parameters: for example, VERSION FRAME instances; for network description of "*area West*"; for vehicle schedule parameters in "*depot D";* for fare versions on "*tramway lines*", etc. When these limiting parameters are actual instances of entities stored in the database, this may be described by the entity VALIDITY CONDITION, related to VERSION FRAME. For instance, a VALIDITY CONDITION may represent a DEPOT, a LINE or a GROUP OF LINEs, a particular OPERATOR, etc.

7.3.6.2 Generic Version Frame – Physical Model

7.3.6.2.1 Version Frame – Physical Model

The following diagram shows the Physical model for the NeTEx versioning entities. A VERSION FRAME contains instances of ENTITY IN VERSION IN VERSION FRAME (i.e. the normal NeTEx elements).

In NeTEx, VERSION FRAME is specialised to create actual frame instances, for example SITE FRAME, RESOURCE FRAME, etc., each containing a restricted set of ENTITIEs for a particular business purpose.

Normally there will only be one VERSION of each of the top level ENTITIes in a given frame, reflecting a single coherent set of data elements. However it is possible to include multiple versions of the same ENTITY if appropriate.

A VERSION FRAME may be given certain VERSION FRAME DEFAULTs which describe properties that apply to all elements in the VERSION FRAME unless explclity overridden, for example LOCALE and DATA SOURCE.

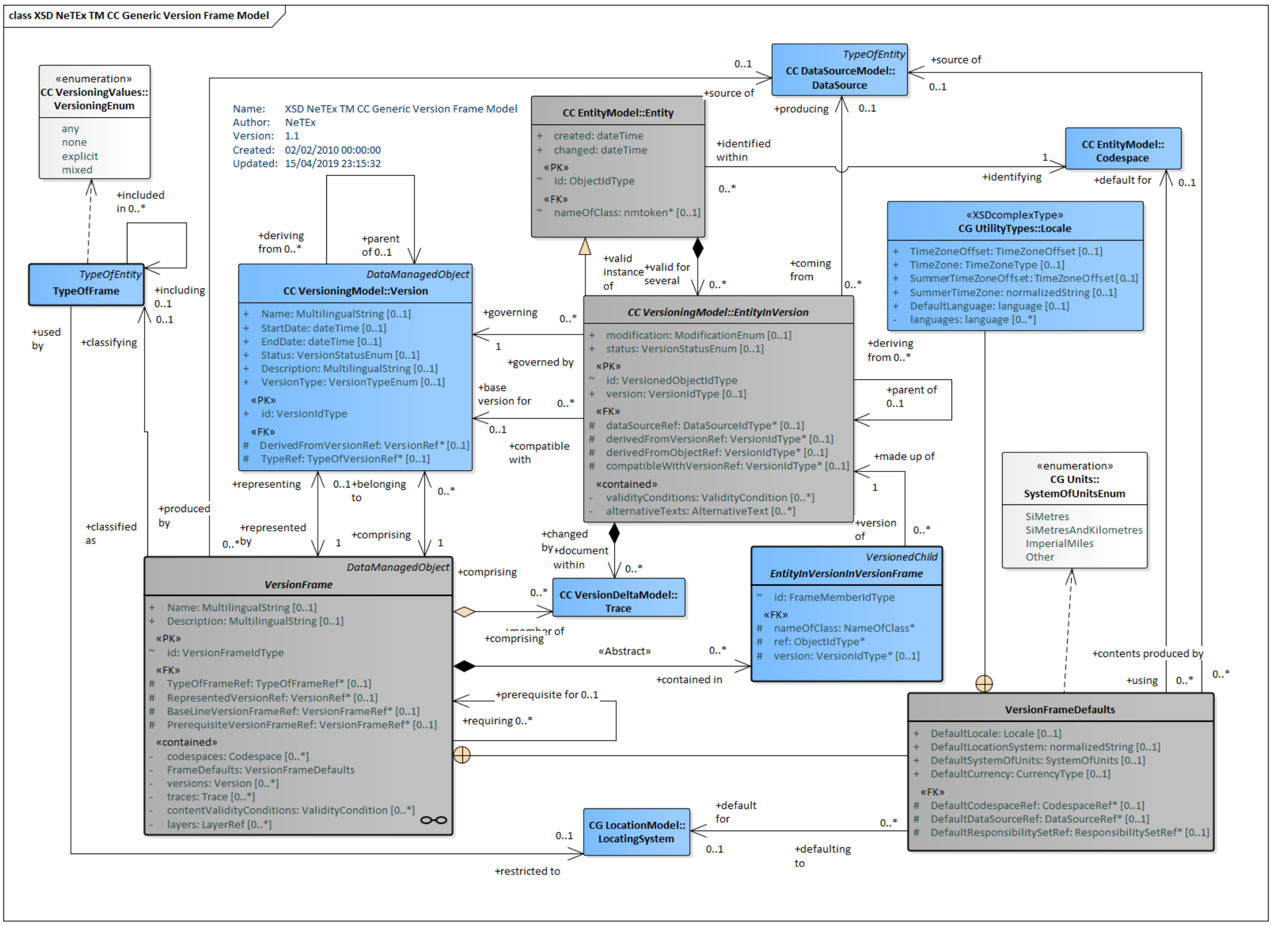


Figure 53 — Version Frame – Physical Model (UML)

7.3.6.2.2 Type of Version Frame – Physical Model

The following diagram shows the Physical model for the NeTEx TYPE OF VERSION FRAME entities.

A TYPE OF FRAME can be used to define the properties of a profile describinghow to to use for a particular purpose. A TYPE OF FRAME DEFINITION can include CLASS IN FRAME elements to specify which actual elements must be present.

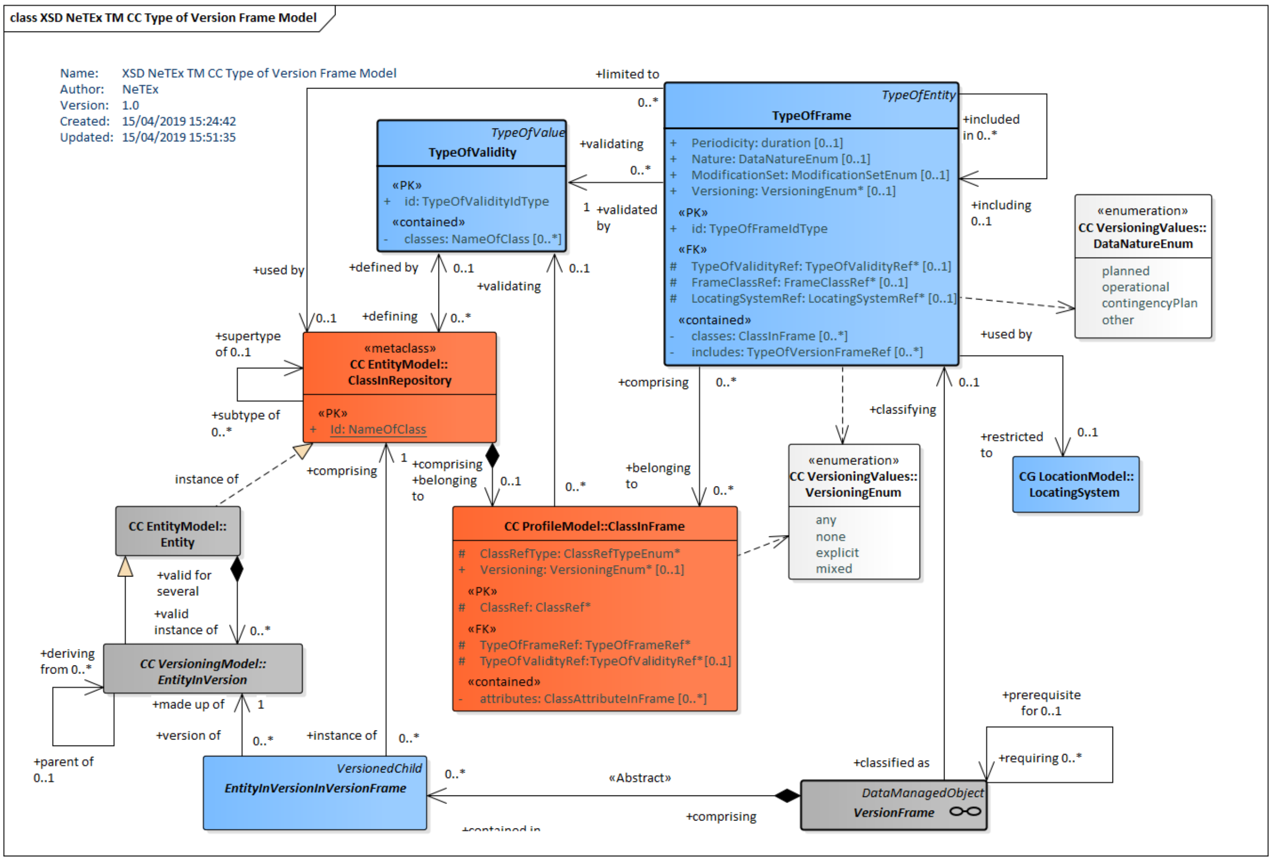


Figure 54 — Type of Version Frame – Physical Model (UML)

7.3.6.2.3 Profile Model – Physical Model

The following diagram shows the use of a TYPE OF VERSION FRAME to specify a profile. This provides in effect a form of simple metadata allowing an implementor to make elements and attributes mandatory.

A TYPE OF FRAME can indicate a ***ClassInRepository*** to specify the class of VERSION FRAME to be used for the profile (e.g. *TimetableFrame*, *FareFrame*, etc). It may also specify a set of ***ClassInFrame*** elements specifying which ENTITIEs must be present or absent. Each ***ClassInFrame*** may furthermore specify ***ClassAttributeInFrame*** and ***ClassRelationshipInFrame*** telements to specify which XML attribute elements for the ENTITY must be present or absent.

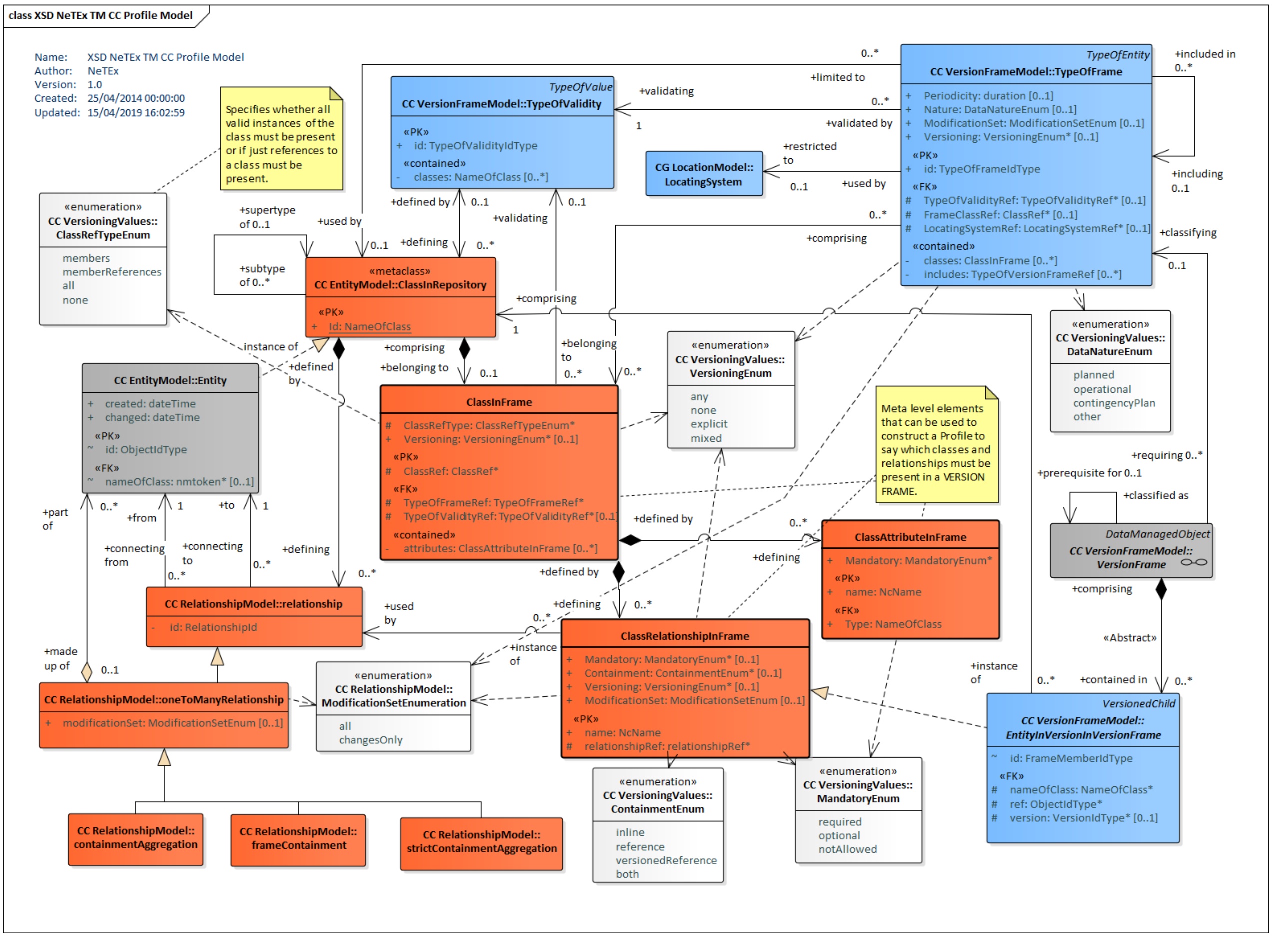


Figure 55 — Profile Model – Physical Model (UML)

7.3.6.3 Version Frame – Attributes and XSD

7.3.6.3.1 VersionFrame – Model Element

A set of VERSIONs referring to a same DATA SYSTEM and belonging to the same TYPE OF FRAME. A FRAME may be restricted by VALIDITY CONDITIONs.

The frame may include default values that apply to all elements so that they do not specify a specific value of their own – see ***FrameDefaults***.

Table 27 — *VersionFrame* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VERSION FRAME inherits from DATA MANAGED OBJECT.
«PK»	***id***	*VersionFrameIdType*	1:1	Identifier of VERSION FRAME.
	Name	*MultilingualString*	0:1	Description of VERSION FRAME.
	Description	*MultilingualString*	0:1	Description of VERSION FRAME.
«FK»	***TypeOfFrameRef***	*TypeOfFrameRef*	0:1	Reference to TYPE of VERSION FRAME.

«FK»	***BaselineVersion-FrameRef***	*VersionRef*	0:1	Previous baseline version frame that obejcts in this frame require.
«cntd»	***codespaces***	*Codespace*	0:*	CODESPACES used in this frame. Normally there will be at least one. A default may be specified in the Frame defaults.
«cntd»	***FrameDefaults***	*FrameDefaults*	0:1	Set of default values to assume for values in frame if not explicitly stated on individual elements.
«cntd»	***versions***	*Version*	0:*	Formal definitions of VERSIONS included in the FRAME.
«cntd»	***prerequisites***	*VersionFrameRef*	0:*	Prerequisite VERSION FRAMEs containing elements that this frame depends on. +v1.1
«cntd»	***traces***	*Trace*	0:*	TRACEs recording changes to ENTITIES in FRAME.
«cntd»	***contentValidity-Conditions***	*ValidityCondition*	0:*	Shared VALIDITY CONDITIONs used by elements in the FRAME.
«cntd»	***layers***	*LayerRef*	0:*	Layers included in VERSION FRAMEs. +v1.1

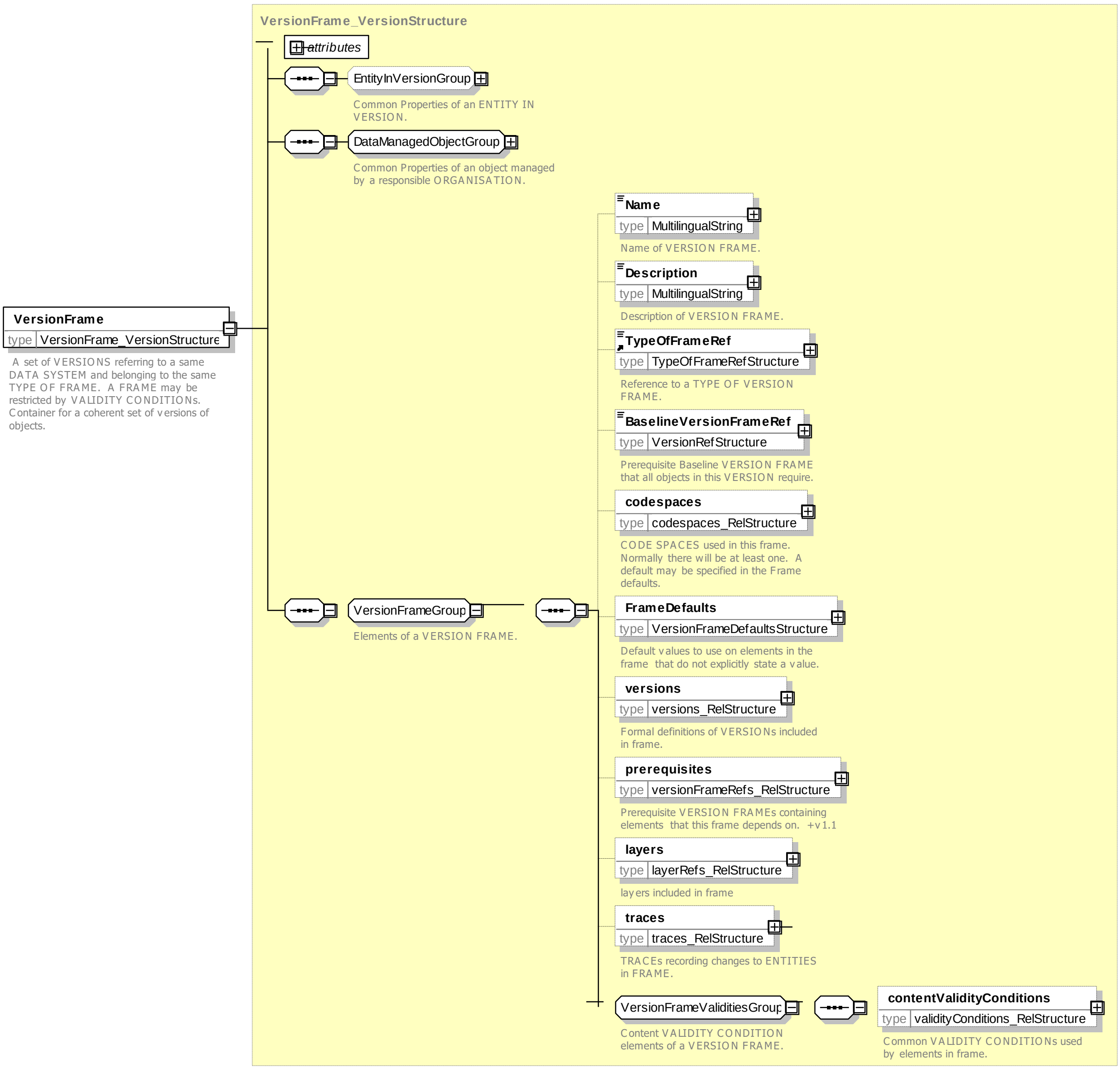


Figure 56 — *VersionFrame – XSD*

7.3.6.3.1.1 FrameDefaults – Wrapper Element

The *FrameDefaults* element specifies default values for certain common properties of elements in the frame, such as DATA SOURCE, time-zone etc., to be applied to elements in the frame for which an explicit value has not been specifed. The use of defaults can both simplify export and reduce the size of documents.

Table 28 —*FrameDefaults* – Element

Classification	Name	Type	Cardinality	Description
«FK»	***DefaultCodeSpace-Ref***	*CodeSpaceRef*	0:1	Default CODESPACE to assume for an identifiers that do not have an explicit CODESPACE specified.
«FK»	***DefaultDataSource-Ref***	*DataSourceRef*	0:1	DATA SOURCE to use for elements in the frame which do not have a DATA SOURCE specified.

«FK»	***Default-ResponsibilitySetRef***	*ResponsibilitySetRef*	0:1	RESPONSIBILITY SET to use for elements in the frame which do not have a RESPONSIBILITY SET specified.
«cntd»	***DefaultLocale***	*Locale*	0:1	Default LOCALE to use to provide attribute values for elements in the frame which do not have a LOCALE element specified, for example language time zone, etc.
	DefaultLocation-System	*xsd:normalizedString*	0:1	Default LOCATION SYSTEM to use for locations in the frame which do not have a LOCATION SYSTEM specified.
«enum»	***DefaultSystemOf-Units***	*SystemOfUnits*Enum	0:1	Default *SystemOfUnits* to use for measurable attributes in the frame for which no units have been specified. See allowed values.
«enum»	***DefaultCurrency***	*CurrencyType*	0:1	Default CURRENCY to use for amount attributes in the frame for which a currency has not been specified.

Figure 57 — *VersionFrameDefaults – XSD*

7.3.6.3.2 EntityInVersionInVersionFrame – Model Element

Nearly all of the data elements exchange by NeTEx are instances of ENTITY IN VERSION IN FRAME; they have the attributes of an ENTITY IN FRAME, and exist in the context of a specifc VERSION FRAME.

7.3.6.3.3 TypeOfFrame – Model Element

A TYPE OF FRAME is a classification of VERSION FRAMEs according to a common purpose. E.g. line descriptions for line versions, vehicle schedules, operating costs. A TYPE OF FRAME is ruled by a unique TYPE OF VALIDITY.

A TYPE OF FRAME may be used to classify a profile for using NeTEx for a specific purpose; see profile model below.

Table 29 — *TypeOfFrame* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	*::>*	TYPE OF VALIDITY inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfFrameIdType*	1:1	Identifier of TYPE OF FRAME.
«FK»	***TypeOfValidity-Ref***	*TypeOfValidityRef*	0:1	Reference to a TYPE OF VALIDITY.
«FK»	***FrameClassRef***	*ClassRef*	0:1	Refence to a XML class of a type of VERSION FRAME (***ResourceFrame, SiteFrame, TimetableFrame,*** etc.), that is, the XML class of frame that VERSION FRAMEs having this specific TYPE OF FRAME should to use to hold elements.
«cntd»	***classes***	*ClassInFrameRef*	0:*	Classes to be present in frames conforming to the TYPE OF FRAME.
«cntd»	***typesOfEntity***	*TypeOfEntityRef*	0:*	TYPEs OF ENTITY to include in FRAME. + v1.1
«cntd»	***includes***	*TypeOfFrame*	0:*	TYPEs OF FRAME contained in TYPE OF FRAME. Must not be cyclic.
«FK»	***LocatingSystem Ref***	*LocatingSystemRef*	0:1	Locating system to use for frames of this type.
«enum»	***ModificationSet***	*ModificationSetEnum*	0:1	Nature of contents in set: See allowed value s below.

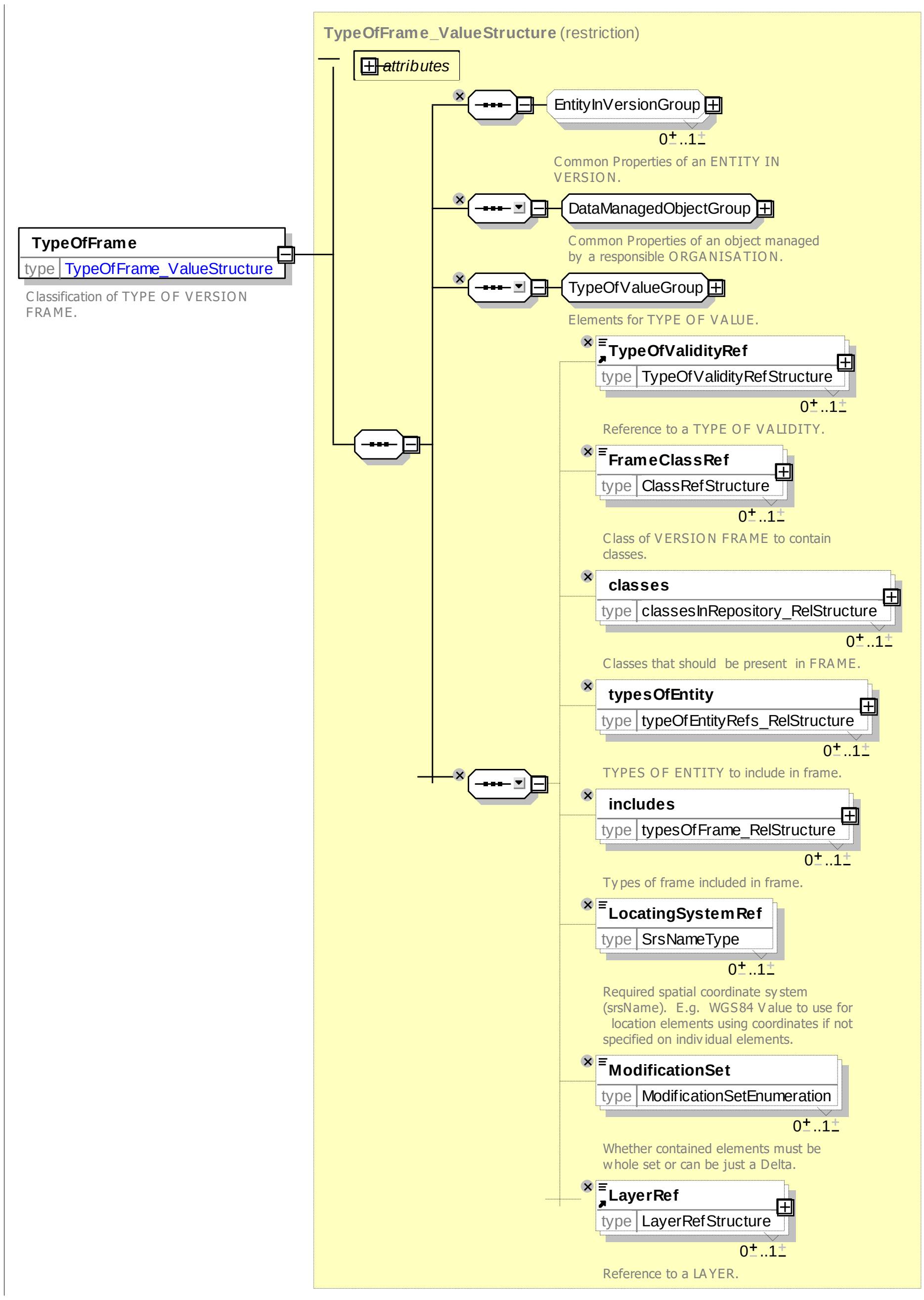


Figure 58 — *TypeOfFrame – XSD*

7.3.6.3.3.1 ModificationSet – Allowed Values

Allowed values for ***ModificationSet*** (*ModifcationSetEnumeration)*.

Table 30 *ModificationSet* – Allowed values

Value	Description
all	Include all properties (attributes and subcomponents).
changesOnly	Include only those properties which have changed. Required attributes will always be included.

7.3.6.3.4 TypeOfValidity – Model Element

A classification of the validity of a TYPE OF FRAME, e.g. VERSION FRAMEs for schedules designed for DAY TYPEs, SITEs, dated schedules, etc.

Table 31 - *TypeOfValidity* – Element

Classification	**Name**	**Type**	**Cardinality**	**Description**
::>	::>	*TypeOfValue*	::>	TYPE OF VALIDITY inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfValidityIdType*	1:1	Identifier of Validity type. Unique within repository.
	Periodicity	*xsd:duration*	0:1	How long frames of this content are valid for by default.
«enum»	***Nature***	*DataNatureEnum*	0:1	Nature of validity data. See allowed values below.
«cntd»	***classes***	*ClassRef*	0:*	Reference to the XML class used to implementof an ENTITY.

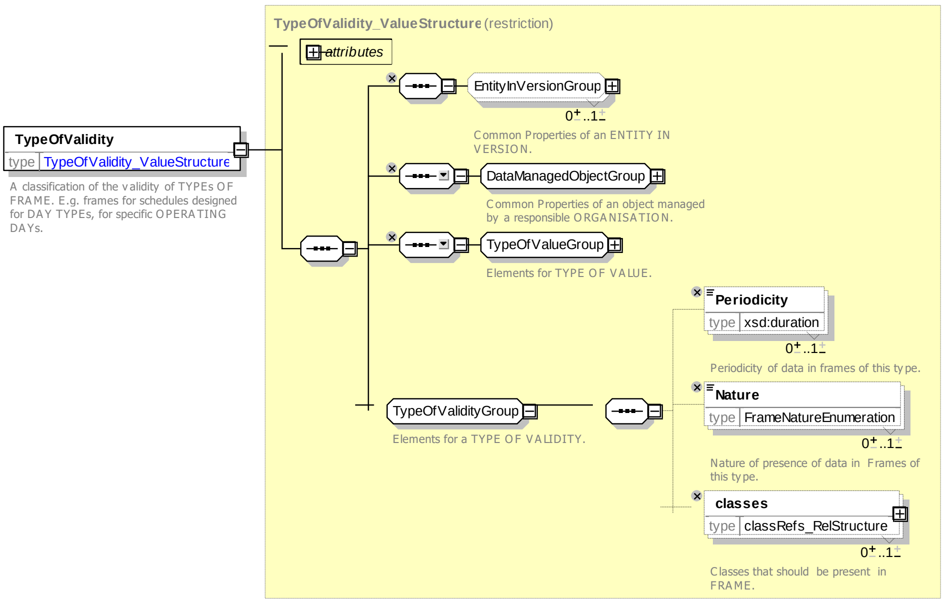


Figure 59 — *TypeOfValidity – XSD*

7.3.6.3.4.1 DataNature – Allowed Values

Allowed values for the nature of a TYPE OF VALIDITY (*FrameNatureEnumeration).*

Table 32 *TypeOfValidity* – allowed values

Value	**Description**
planned	Data is for planning
operational	Operational data
contingencyPlan	Data is for contingency plan to be used in certain circumstances
other	Other

7.3.6.3.5 ClassInFrame– Model Element

Requirements for use of an XML Class representing a given entity in a VERSION FRAME. This is a metaclass that allows services to specify whether an XML class for an ENTITY must or must not be present in a given

VERSION FRAME (Most NeTEx elements are otherwise optional in a VERSION FRAME, as are their attributes). Furthermore, it can indicate whether indirect references should also be included.

For example:

- To state that a ***ServiceFrame*** *must* contain **ServiceLinks**, the ***TypeOfFrame*** associated with the ***ServiceFrame*** would have a ***ClassinFrame*** with '***nameOfClass***="*ServiceLink"*, ***Mandatory***='true' and '***classRefType***="*members"*.

- To state that the ***ServiceFrame*** must include any ***ScheduledStopPoints*** referenced by each ***ServiceLink*** (but not unreferenced stop points), the ***TypeOfFrame*** would have a ***ClassinFrame*** of '***nameOfClass***="*ScheduledStopPoint*"' and '***classRefType***="*memberReferences"*.

Table 33 *ClassInFrame* – Element

Classification	Name	Type	Cardinality	Description
«PK»	***nameOfClass***	*NameOfClass*	1:*	Reference to the XML Class of the ENTITY.
«enum»	***classRefType***	*ClassRefTypeEnum*	0:1	Type of XML Class reference. The default is '*members*'. See allowed values below.
	Mandatory	*xsd:booean*	0:1	Whether any instances of class must be included in Frame
«cntd»	***attributes***	*ClassAttributeInFrame*	0:*	Requirements for an attribute of the XML class used to implement an ENTITY.
«cntd»	***relationships***	*ClassRelationshipInFrame*	0:*	Requirements for a attribute of the XML class used to implement a relationship of an ENTITY.

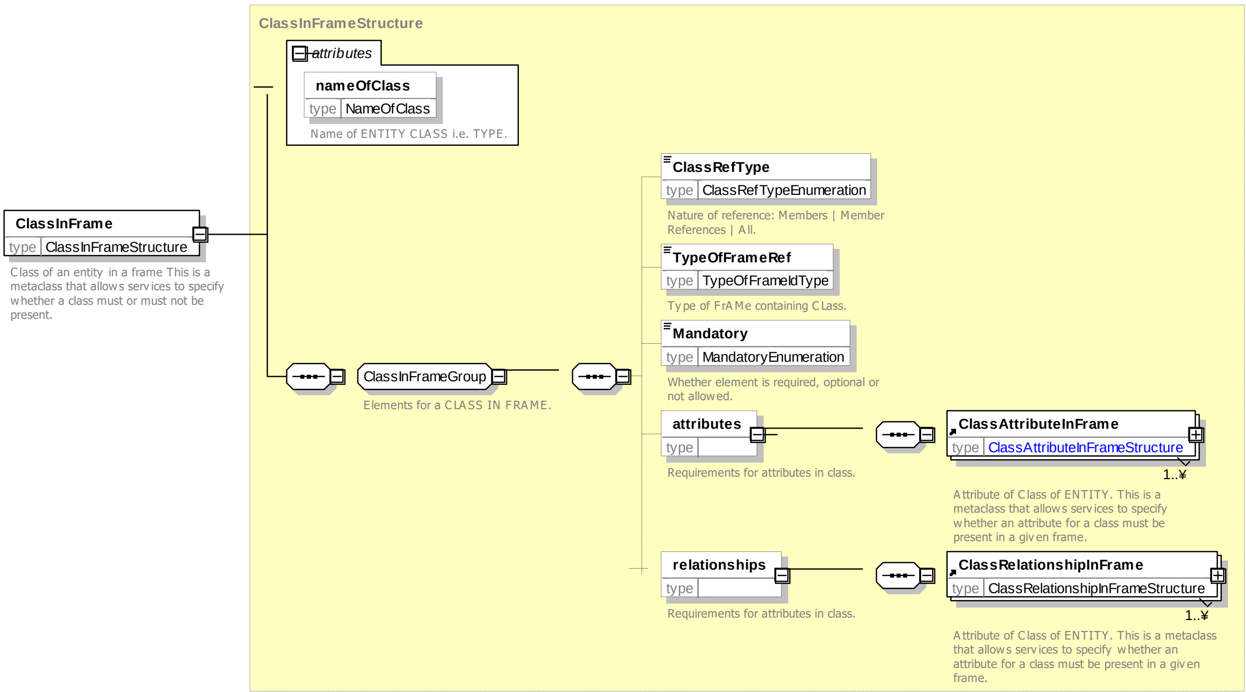


Figure 60 — *ClassInFrame – XSD*

7.3.6.3.5.1 ClassRefType – Allowed Values

Allowed values for class references (*ClassRefTypeEnumeration)*.

Table 34 *ClassRefType* – Allowed values

Value	Description
all	Include all known instances of class that meet validity conditions
members	Include all instances of class that are direct components of another class.
memberReferences	Only include ancillary class instance if referenced by another instance

7.3.6.3.6 ClassAttributeInFrame– Model Element

Requirements for use of an XML Class representing a given entity in a VERSION FRAME. This is a metaclass that allows services to specify whether an XML class for an ENTITY must or must not be present in a given VERSION FRAME (Most NeTEx elements are otherwise optional in a VERSION FRAME, as are their attributes). Furthermore, it can indicate whether indirect references should also be included.

For example;-

- To state that a ***ServiceFrame*** *must* contain **ServiceLinks**, the ***TypeOfFrame*** associated with the ***ServiceFrame*** would have a ***ClassinFrame*** with '***nameOfClass***="*ServiceLink*", ***Mandatory***='true' and '***classRefType***="*members*".

- To state that the ***ServiceFrame*** must include any ***ScheduledStopPoints*** referenced by each ***ServiceLink*** (but not unreferenced stop points), the ***TypeOfFrame*** would have a ***ClassinFrame*** of '***nameOfClass***="*ScheduledStopPoint*"' and '***classRefType***="*memberReferences*".

Table 35 *ClassAttributeInFrame* – Element

Classification	Name	Type	Cardinality	Description
«atr», «PK»	***nameOfClass***	*NameOfClass*	0:1	Reference to the XML Class of the ENTITY.
«PK»	***Type***	*QName*	0:1	Name of XML attribute of an ENTITY.
	Mandatory	*xsd:boolean*	0:1	Whether any instances of class must included attribute.

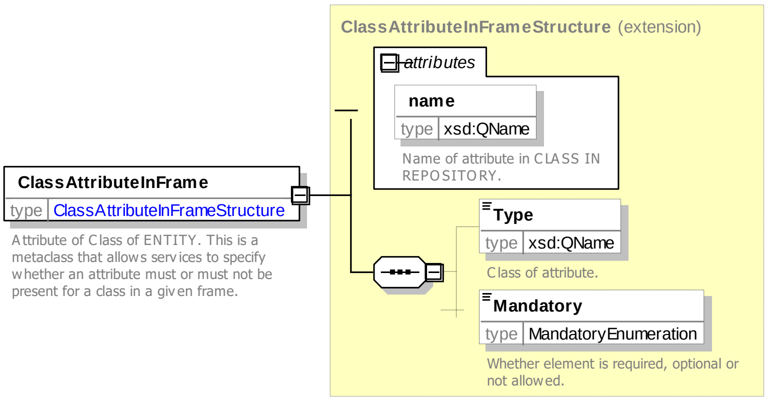


Figure 61 — *ClassAttributeInFrame – XSD*

7.3.6.3.7 ClassRelationshipInFrame– Model Element

Requirements for use of an XML Class representing a given entity in a VERSION FRAME. This is a metaclass that allows services to specify whether an XML class for an ENTITY must or must not be present in a given VERSION FRAME (Most NeTEx elements are otherwise optional in a VERSION FRAME, as are their attributes). Furthermore, it can indicate whether indirect references should also be included.

For example;-

- To state that a ***ServiceFrame*** *must* contain **ServiceLinks**, the ***TypeOfFrame*** associated with the ***ServiceFrame*** would have a ***ClassinFrame*** with '***nameOfClass***="*ServiceLink"*, ***Mandatory***='true' and '***classRefType***="*members"*.

- To state that the ***ServiceFrame*** must include any ***ScheduledStopPoints*** referenced by each ***ServiceLink*** (but not unreferenced stop points), the ***TypeOfFrame*** would have a ***ClassinFrame*** of '***nameOfClass***="*ScheduledStopPoint"'* and '***classRefType***="*memberReferences"*.

Table 36 *ClassRelationshipInFrame* – Element

Classification	Name	Type	Cardinality	Description
«atr», «PK»	***name***	*Xsd:QName*	0:1	Reference to the XML Class of the ENTITY.
«FK»	***RelationshipRef***	*RelationshipRef*	0:1	Reference to a relationship in metamodel.
	Mandatory	*xsd:boolean*	0:1	Whether the relationship mandatory.
«enum»	***Containment***	*ContainmentEnum*	0:1	Nature of containment of attribute. See allowed values below.
«enum»	***ModificationSet***	*ModificationSetEnum*	0:1	Whether contained elements must be whole set or can be just a Delta. See allowed values under DATA MANAGED OBJECT.

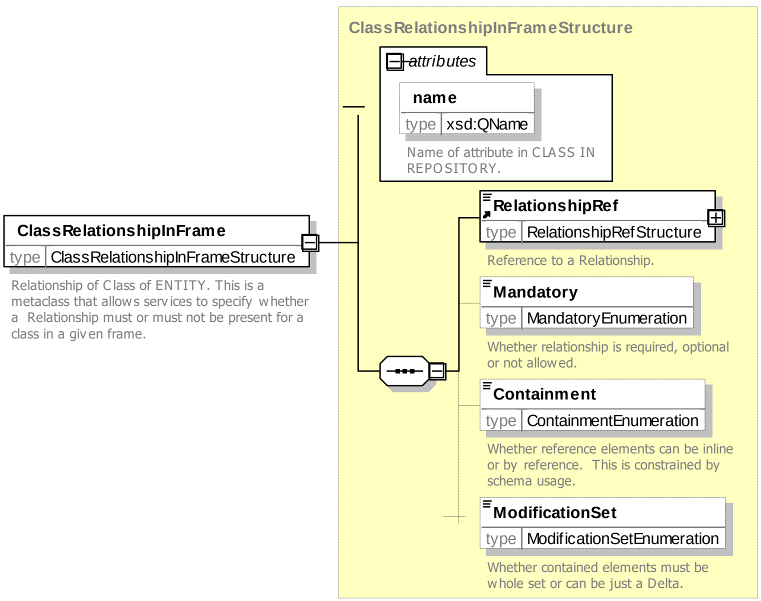


Figure 62 — *ClassRelationshipInFrame – XSD*

7.3.6.3.8 Class (ClassRef) – Model Element

An XML class implementing a model ENTITY. (A meta reference to an ENTITY).

Table 37 — *ClassInFrame* – Element

Classification	Name	Type	Cardinality	Description
«PK»	***nameOfClass***	*NameOfClass*	1:1	Identifier of XML element used to implement ENTITY that is available in VERSION FRAME

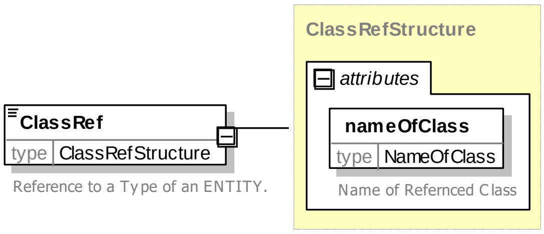


Figure 63 — ClassRef – *XSD*

7.3.6.3.9 Relationship (RelationshipRef) – Model Element

A specific relationship between two ENTITIEs in the model of ENTITies. (A meta reference).

Table 38 — *RelationshipInFrame* – Element

Classification	Name	Type	Cardinality	Description
«PK»	***nameOfClass***	RelastionshipId	1:1	Identifier of relationship

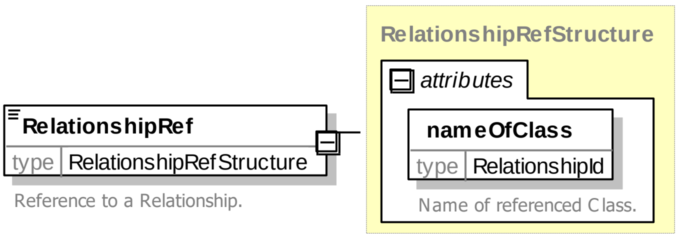


Figure 64 — Relationship (*RelationshipRef) – XSD*

7.3.6.4 XML Examples of Version Frames

7.3.6.4.1 Simple Version Frame – XML fragment

The following XML fragment shows a simple Service VERSION FRAME with two SCHEDULED STOP POINTs in it and a SERVICE LINK that connects them.

EXAMPLE

```
<ServiceFrame version="any" " id="mybus:ntwkf001">
    <Name>My Network (V1) </Name>
    <frameValidityConditions>
```

```xml
            <AvailabilityCondition id="mybus:VC002_mf">
                <Description>Current </Description>
                <FromDate>2010-05-17T00:00:00Z</FromDate>
                <ToDate>2011-05-17T00:00:00Z</ToDate>
            </AvailabilityCondition>
            </frameValidityConditions>
        <scheduledStopPoints>
            <ScheduledStopPoint version="any" id="mybus:SSP0001A">
                <Name>Haltstelle A</Name>
                <Description>Stop A</Description>
            </ScheduledStopPoint>
            <ScheduledStopPoint version="any" id="mybus::SSP0002B">
                <Name>Haltstelle B</Name>
                <Description>Version one of stop B</Description>
            </ScheduledStopPoint>
        </scheduledStopPoints>
        <serviceLinks>
            <ServiceLink version="any" id="mybus:SL_AtoB01">
                <Name>Version one of Link from A to B</Name>
                <Distance>1.12</Distance>
                <FromPointRef version="002" ref="mybus::SSP0001A"/>
                <ToPointRef version="002" ref="mybus::SSP0002B"/>
            </ServiceLink>
        </serviceLinks>
    </ServiceFrame>
```

7.3.6.4.2 Simple Version Frame with Condition – XML fragment

The following XML fragment shows a CompositeFrame VERSION FRAME with a VALIDITY CONDITION.

EXAMPLE

```xml
        <CompositeFrame version="1" id="hde:CompositeFrame:CF_1">
            <frameValidityConditions>
                <ValidityRuleParameter version="1" id="hde:ValidityRuleParameter:VR_1">
                    <Name>Applies to area west</Name>
                    <Description>Name </Description>
                    <ConditionedObjectRef version="1" ref="hde:CompositeFrame:CF_1"/>
                    <RuleObjectRef version="any" ref="hde:TopographicPlace:TP_1"/>
                    <isValid>true</isValid>
                </ValidityRuleParameter>
            </frameValidityConditions>
            <frames>

            </frames>
        </CompositeFrame>
```

7.3.6.4.3 Use of Type Of Frame – XML fragment

The following example shows a frame definition for the weekly export of Neptune data elements.
EXAMPLE

```xml
<ResourceFrame version="any" id="Neptune:FrameProfile01">
    <Name>RATP--Neptune-Line Profile </Name>
    <typesOfValue>
        <TypeOfValidity version="01" id="Neptune:Weekly_planned">
            <Name>Weekley export of Neptune Planned data</Name>
            <Periodicity>P7D</Periodicity>
            <Nature>planned</Nature>
        </TypeOfValidity>
        <TypeOfFrame version="01" id="Neptune:Composite_frame">
            <Name>Neptune Profile: Composite Frame </Name>
            <Description>RATP--Neptune-Line-orderx  frames  will  be  composite  frame  containing
Service, Timetable (and Service Calendar) and Resource frames with all the usual NEPTUNE attributes
filled in, for a given line order.</Description>
            <TypeOfValidityRef version="01" ref="Neptune:Weekly_planned"/>
            <FrameClassRef nameOfClass="CompositeFrame"/>
            <classes>
                <ClassInContextRef                          nameOfClass="AvailabilityCondition"
classRefType="memberReferences"/>
            </classes>
            <typesOfFrame>
                <TypeOfFrame version="01" id="Neptune:Time_table_frame">
```

```xml
                    <Name>Neptune Profile Timetable Frame </Name>
                    <FrameClassRef nameOfClass="TimeTableFrame"/>
                    <classes>
                        <ClassInContextRef classRefType="members" nameOfClass="TimeTableFrame"/>
                        <ClassInContextRef classRefType="members" nameOfClass="ServiceJourney"/>
                        <ClassInContextRef classRefType="members" nameOfClass="JourneyPart"/>
                        <ClassInContextRef classRefType="members" nameOfClass="JourneyPart"/>
                    </classes>
                </TypeOfFrame>
                <TypeOfFrame version="01" id="Neptune:ServiceFrame">
                    <Name>Neptune Profile: Service Frame </Name>
                    <FrameClassRef nameOfClass="ServiceFrame"/>
                    <classes>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="ScheduledStopPoint"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="ServicePattern"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="Network"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="RoutePoint"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="RouteLink"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="Line"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="Route"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="Direction"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="TariffZone"/>
            <ClassInContextRef                              classRefType="memberReferences"
nameOfClass="PassengerStopAssignment"/>
                    </classes>
                </TypeOfFrame>
                <TypeOfFrame version="01" id="Neptune:Site_frame">
                    <Name>Neptune Profile: Site Frame </Name>
                    <FrameClassRef nameOfClass="SiteFrame"/>
                    <classes>
                    <ClassInContextRef classRefType="memberReferences" nameOfClass="StopPlace"/>
                    </classes>
                    </TypeOfFrame>
                <TypeOfFrame version="01" id="Neptune:Service_calendar_frame">
                    <Name>Neptune Profile: Frame </Name>
                    <FrameClassRef nameOfClass="ServiceCalendarFrame"/>
                    <classes>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="ServiceCalendar"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="DayType"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="DayTypeAssignment"/>
                    </classes>
                </TypeOfFrame>
                <TypeOfFrame version="01" id="Neptune:Resource_frame">
                    <Name>Neptune Profile: Resource Frame </Name>
                    <FrameClassRef nameOfClass="ResourceFrame"/>
                    <classes>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="TypeOfFrame"/>
            <ClassInContextRef classRefType="memberReferences" nameOfClass="Operator"/>
            ClassInContextRef classRefType="memberReferences" nameOfClass="Authority"/>
                    </classes>
                </TypeOfFrame>
            </typesOfFrame>
        </TypeOfFrame>
    </typesOfValue>
</ResourceFrame/>
```

The following example shows a Frame definition for a frame to hold data elements to provide a GTFS timetable export.

EXAMPLE

```xml
<TypeOfFrame version="1.5" changed="2011-01-20T00:00:00" id="gtfs:GTFS">
    <Name>GTFS Version Frame definition: A GTFS timetable has the followinbg classes </Name>
    <classes>
        <ClassInContextRef nameOfClass="AvailabilityCondition" classRefType="memberReferences"/>
    </classes>
    <typesOfFrame>
        <TypeOfFrame version="1.5" changed="2011-01-20T00:00:00" id="gtfs:Gtfs_Resource">
            <FrameClassRef nameOfClass="ResourceFrame"/>
            <classes>
                <ClassInContextRef nameOfClass="Authority" classRefType="memberReferences"/>
            </classes>
        </TypeOfFrame>
        <TypeOfFrame version="1.5" changed="2011-01-20T00:00:00" id="gtfs:Gtfs_Service_Calendar">
            <FrameClassRef nameOfClass="ServiceCalendarFrame"/>
            <classes>
```

```xml
                <ClassInContextRef nameOfClass="DayType" classRefType="memberReferences"/>
            </classes>
        </TypeOfFrame>
        <TypeOfFrame version="1.5" changed="2011-01-20T00:00:00" id="gtfs:Gtfs_Site">
            <FrameClassRef nameOfClass="SiteFrame"/>
            <classes>
                <ClassInContextRef nameOfClass="StopPlace" classRefType="memberReferences"/>
            </classes>
        </TypeOfFrame>
        <TypeOfFrame version="1.5" changed="2011-01-20T00:00:00" id="gtfs:Gtfs_Service">
            <FrameClassRef nameOfClass="ServiceFrame"/>
            <classes>
                <ClassInContextRef nameOfClass="Direction" classRefType="memberReferences"/>
                <ClassInContextRef nameOfClass="Line" classRefType="memberReferences"/>
                <ClassInContextRef                              nameOfClass="ScheduledStopPoint"
classRefType="memberReferences"/>
                <ClassInContextRef nameOfClass="TariffZone" classRefType="memberReferences"/>
            </classes>
        </TypeOfFrame>
            <TypeOfFrame version="1.5" changed="2011-01-20T00:00:00" id="gtfs:Gtfs_Timetable">
            <FrameClassRef nameOfClass="TimetableFrame"/>
            <classes>
                <ClassInContextRef                           nameOfClass="AvailabilityCondition"
classRefType="memberReferences"/>
                <ClassInContextRef nameOfClass="TimetableFrame" classRefType="members"/>
                <ClassInContextRef nameOfClass="ServiceJourney" classRefType="members"/>
                <ClassInContextRef nameOfClass="TemplateServiceJourney" classRefType="members"/>
            </classes>
        </TypeOfFrame>
    </typesOfFrame>
</TypeOfFrame>
```

7.3.7 Generic Validity

7.3.7.1 Generic VALIDITY – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

An ENTITY, a VERSION or a VERSION FRAME may be associated with VALIDITY CONDITION, detailing when an entity or version is active or available.

Each VALIDITY CONDITION can consist of:

— a parameter (e.g. a start date);

— a type of application of this parameter (“for”, “from”, “until”, etc.).

A VALIDITY CONDITION parameter may be:

— a time-related parameter, which will be in general an instance of an entity: OPERATING DAY, PERIOD (with a start date and possibly an end date), PROPERTY OF DAY, DAY TYPE, TIME BAND, TYPE OF TIME BAND, etc.;

— a VALIDITY TRIGGER (road works, rainy weather, until further advice, etc.), which will be activated thanks to a mechanism, an external output or a manual entry;

— any other VALIDITY RULE PARAMETER;

— to simplify implementation a condition with standardised temporal attributes is provided – See AVAILABILITY CONDITION LATER BELOW.

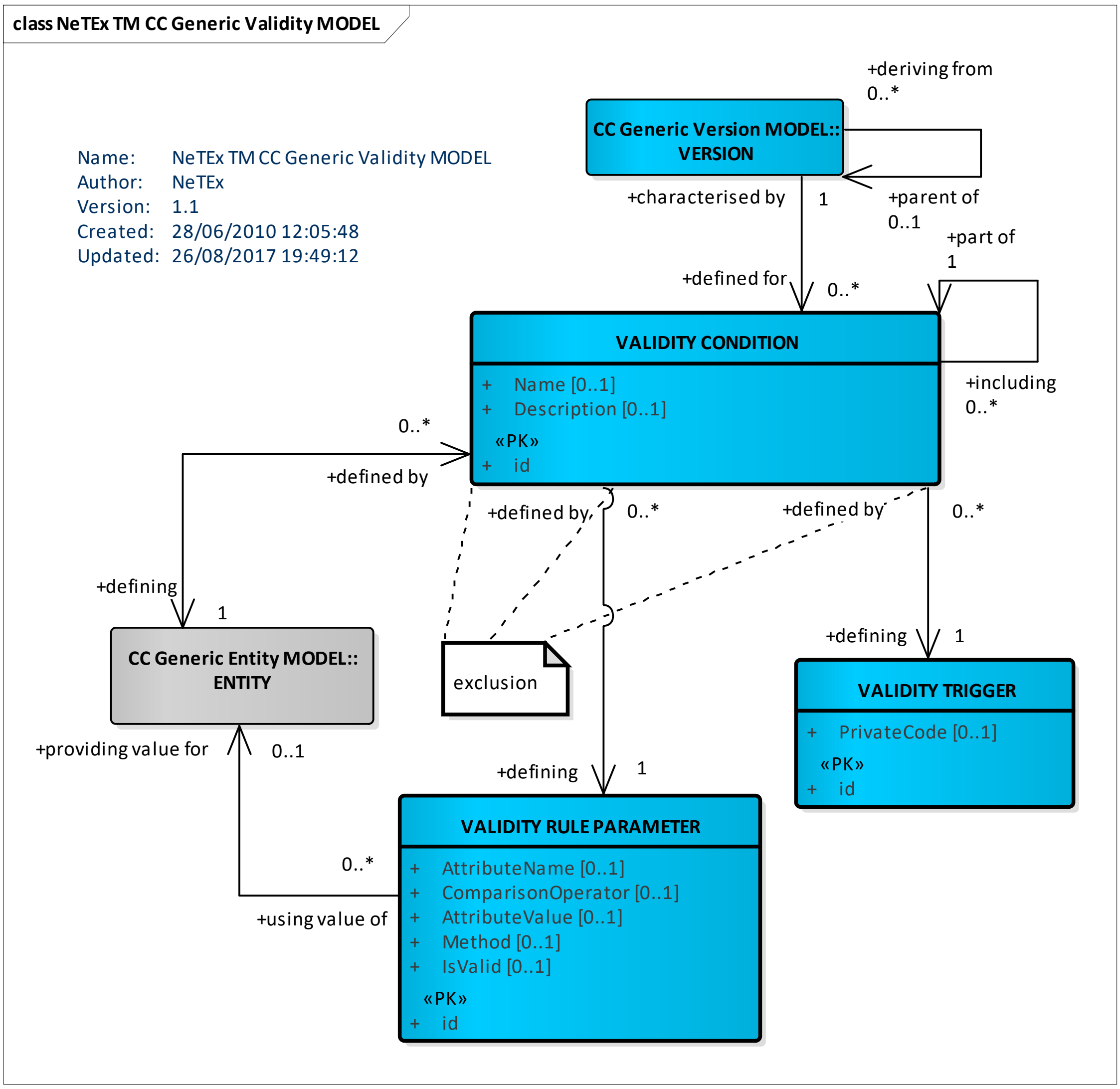


Figure 65 — Generic Validity – Conceptual MODEL (UML)

7.3.7.2 Generic Validity – Physical Model

The following figure shows the physical model for VALIDITY CONDITIONs.

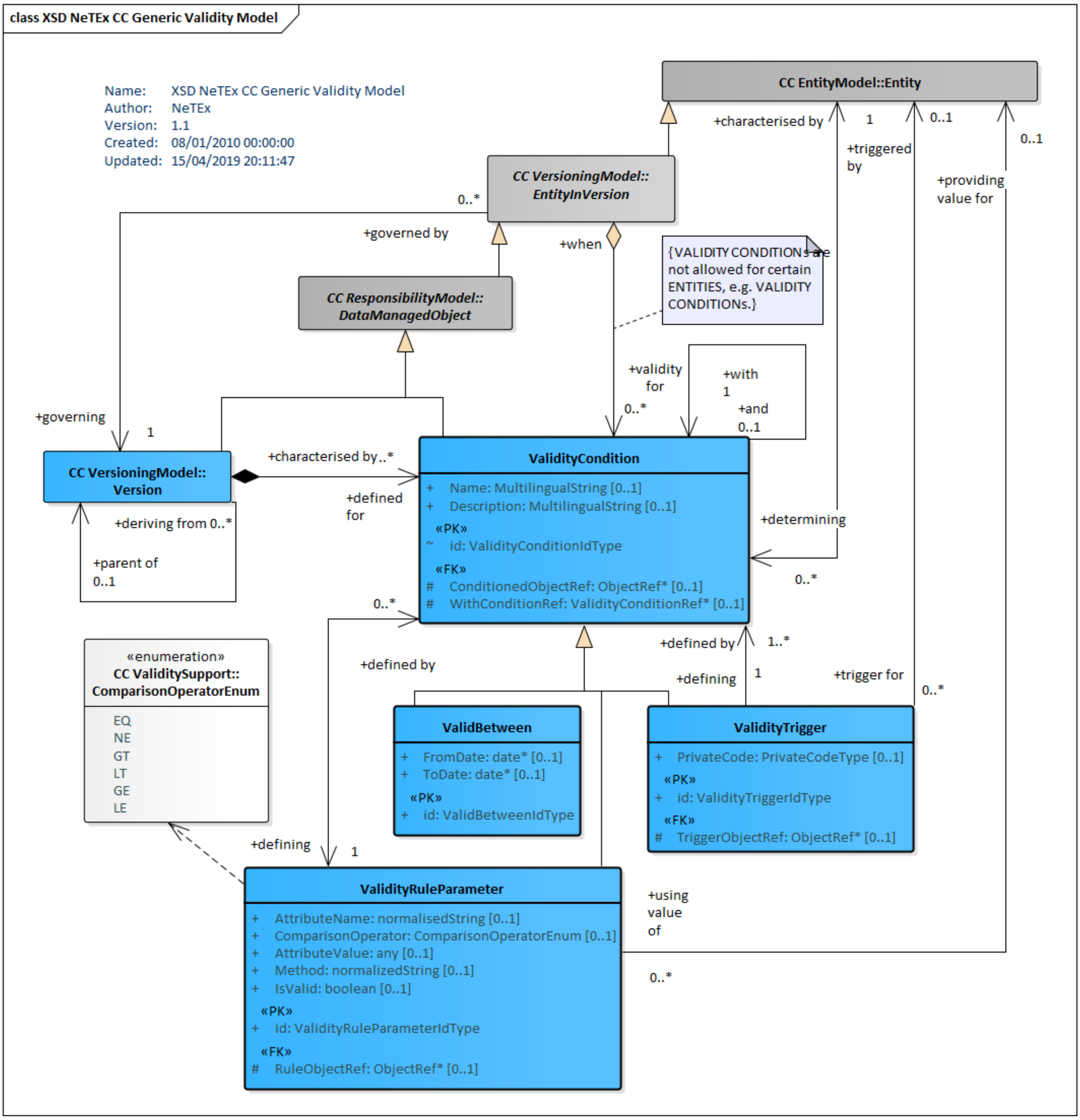


Figure 66 — Generic Validity – Physical Model (UML)

7.3.7.2.1 Use of Validity Conditions

VALIDITY CONDITIONs are not allowed on VALIDITY CONDITIONs and some other elements.

To simplify implementation, VALIDITY CONDITIONs are normally associated only with a subset of NeTEx elements, including those shown in the following diagram.

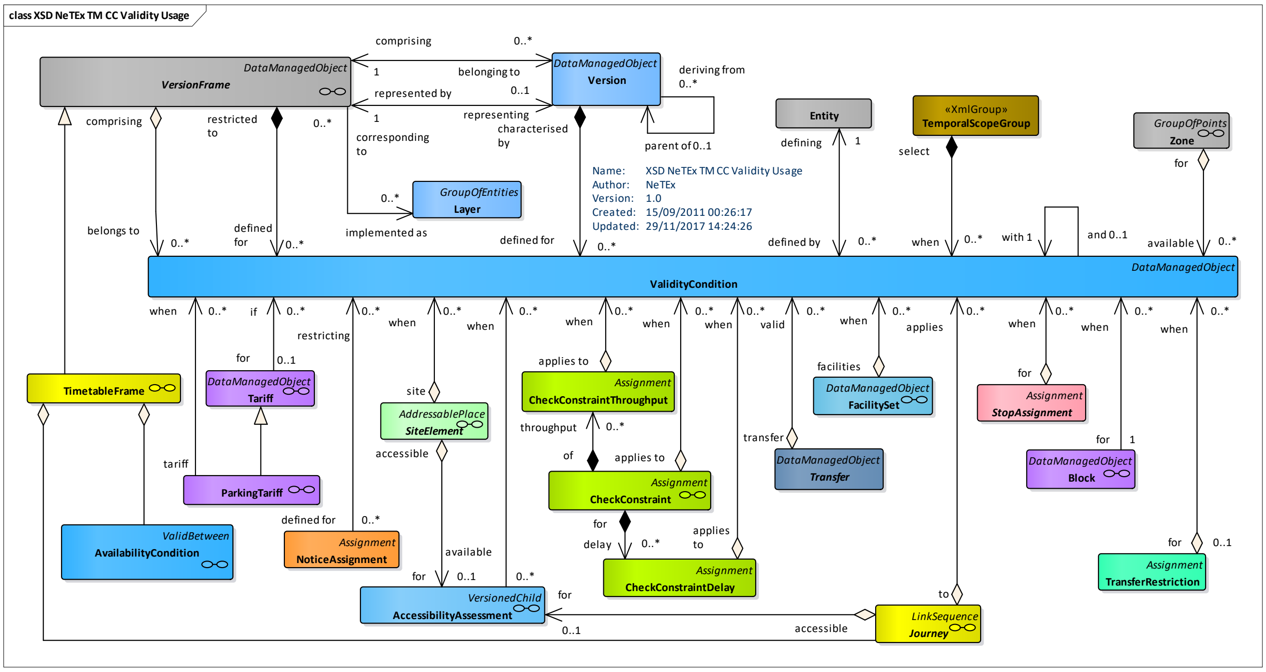


Figure 67 — Use of Validity Conditions (UML)

Table 39—Entities that often have Validity Conditions

	Element	Purpose of VALIDITY CONDITION
Framework	***VersionFrame***	To specify when the contents of a particular FRAME are valid.
Part1	***SiteElement***	To specify the availability of any SITE (STOP PLACE, POINT OF INTEREST) etc. or SITE COMPONENT, (ENTRANCE QUAY etc.).
	CheckConstraint	To specify when a particular CHECK CONSTRAINT applies to a SITE COMPONENT.
	CheckConstraintDelay	To specify when a particular delay applies to a SITE COMPONENT.
	CheckConstraintThroughput	To specify when a particular throughput applies to a SITE COMPONENT.
	Transfer	To specify the availability of any CONNECTION, ACCESS LINK etc.
	StopAssignment	To specify the availability of any platform assignment etc.
	AccessibilityAssessment	To specify the availability of any ACCESSIBILITY LIMITATION or SUITABILITY.
	NoticeAssignment	To specify the applicability of a NOTICE.
Part 2	***Journey***	To specify the availability of a VEHICLE JOURNEY or SERVICE JOURNEY.
	Timetable	To specify the availability of a VEHICLE JOURNEY or SERVICE JOURNEY.

7.3.7.3 ValidityCondition – Attributes and XSD

7.3.7.3.1 ValidityCondition – Model Element

Condition used in order to characterise a given VERSION of a VERSION FRAME. A VALIDITY CONDITION consists of a parameter (e.g. date, triggering event, etc.). and its type of application (e.g. for, from, until, etc.).

Note that ValidityCondition can be combined and ANDed (all the conditions must be fulfilled at the same time) thanks to the ***WithConditionRef*** attribute.

Table 40 — *ValidityCondition* – Element

Classification	Name	Type	Cardinality	Description
::>		*EntityInVersion*		NB Not allowed a VALIDITY
::>	::>	*DataManagedObject*	*::>*	VALIDITY CONDITION Inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ValidityCondition-IdType*	1:1	Identifier of VALIDITY CONDITION.
	Name	*MultilingualString*	0:1	Name of VALIDITY CONDITION.
	Description	*MultilingualString*	0:1	Description of VALIDITY CONDITION.
«FK»	***Conditioned-ObjectRef***	*VersionOfObjectRef*	0:1	Reference to object to which VALIDITY CONDITION applies.
«FK»	***WithConditionRef***	*ValidityConditionRef+*	0:1	CONDITION with which this rule is logically ANDed.

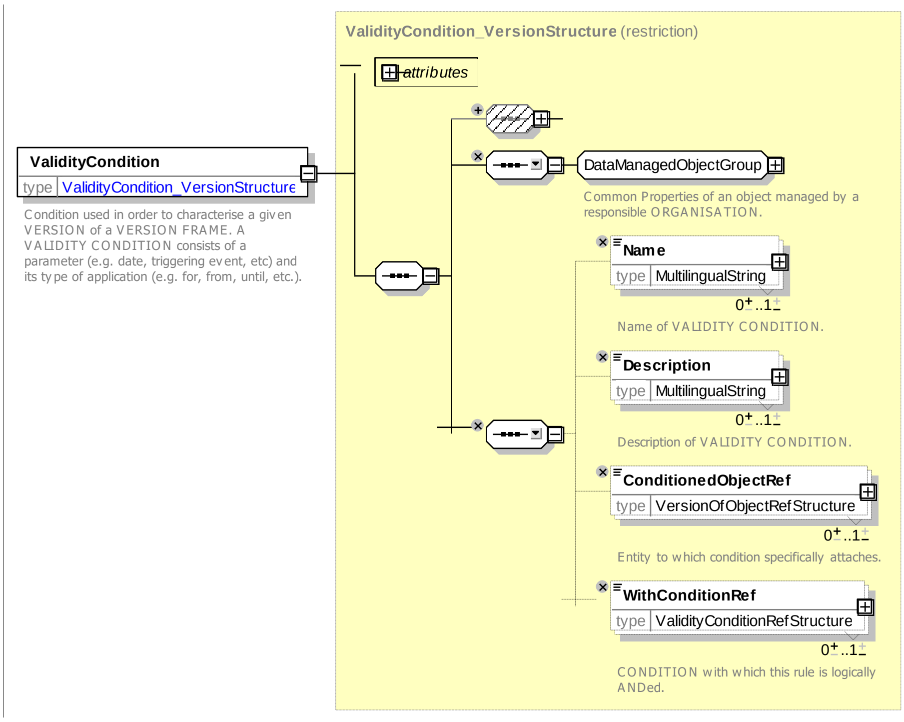


Figure 68 — *ValidityCondition – XSD*

7.3.7.3.2 ValidBetween – Model Element

ValidBetween is a simplified view of a VALIDITY CONDITION, having only a start date and end date. Start or end date may be skipped if unknown (anytime up to, or anytime from), but at least one of them must be provided.

Table 41 — *ValidBetween* – Element

Classification	Name	Type	Cardinality	Description
::>		*EntityInVersion*		NB Not allowed a VALIDITY
::>	::>	*DataManagedObject*	::>	Inherits from DATA MANAGED OBJECT. NB not allowed attributes.
	FromDate	*xsd:dateTime*	0:1	Start date of AVAILABILITY CONDITION.
	ToDate	*xsd:dateTime*	0:1	End of AVAILABILITY CONDITION. Date is INCLUSIVE.

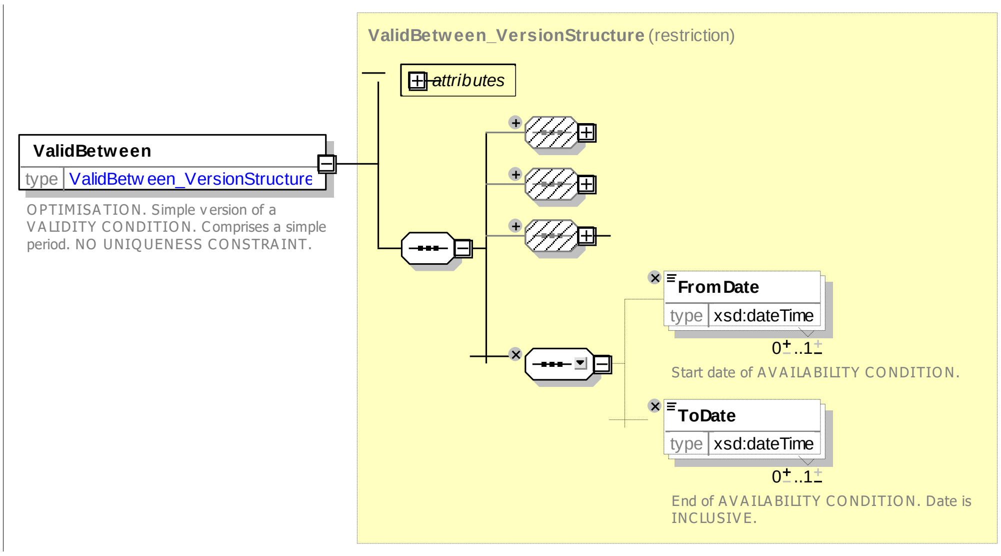


Figure 69 — *ValidBetween – XSD*

7.3.7.3.3 ValidityRuleParameter – Model Element

A user defined VALIDITY CONDITION used by a rule for selecting versions, E.g. river level > 1,5 m and bad weather.

The values of an attribute of the object can be specified to test for values. Can also test if the object is current considered “valid” ie curremnt within its validity dates and any actoive/inactive status.

Multiple Rule Parameters can be logically ANDed together.

Table 42 *ValidityRuleParameter* – Element

Classification	Name	Type	Cardinality	Description
::>		*EntityInVersion*		NB Not allowed a VALIDITY
::>	::>	*DataManagedObject*	::>	VALIDITY RULE PARAMETER Inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ValidityRuleParameter-IdType*	0:1	Identifier of VALIDITY RULE PARAMETER.
«FK»	***RuleObjectRef***	*(VersionOfObjectRef)*	0:1	ENTITY to which this rule applies.
	AttributeName	*xsd:NCTOKEN*	0:1	Name of an Attribute of the referenced ENTITY which provides Trigger value.
«enum»	***Comparison-Operator***	*RelativeOperatorEnum*	0:1	Relative operator for rule. See allowed values.
		choice	0:1	
	a ***AttributeValue***	*xsd:anyType*	0:1	Value of attribute to be compared with that of of ENTITY, using ***ComparisonOperator***.

	b Method	*xsd:anyType*	0:1	Method for computing an attribute. Some systems may support additional means of comparison.
	IsValid	*xsd:boolean*	0:1	Rule should test if the referenced element is currently valid.

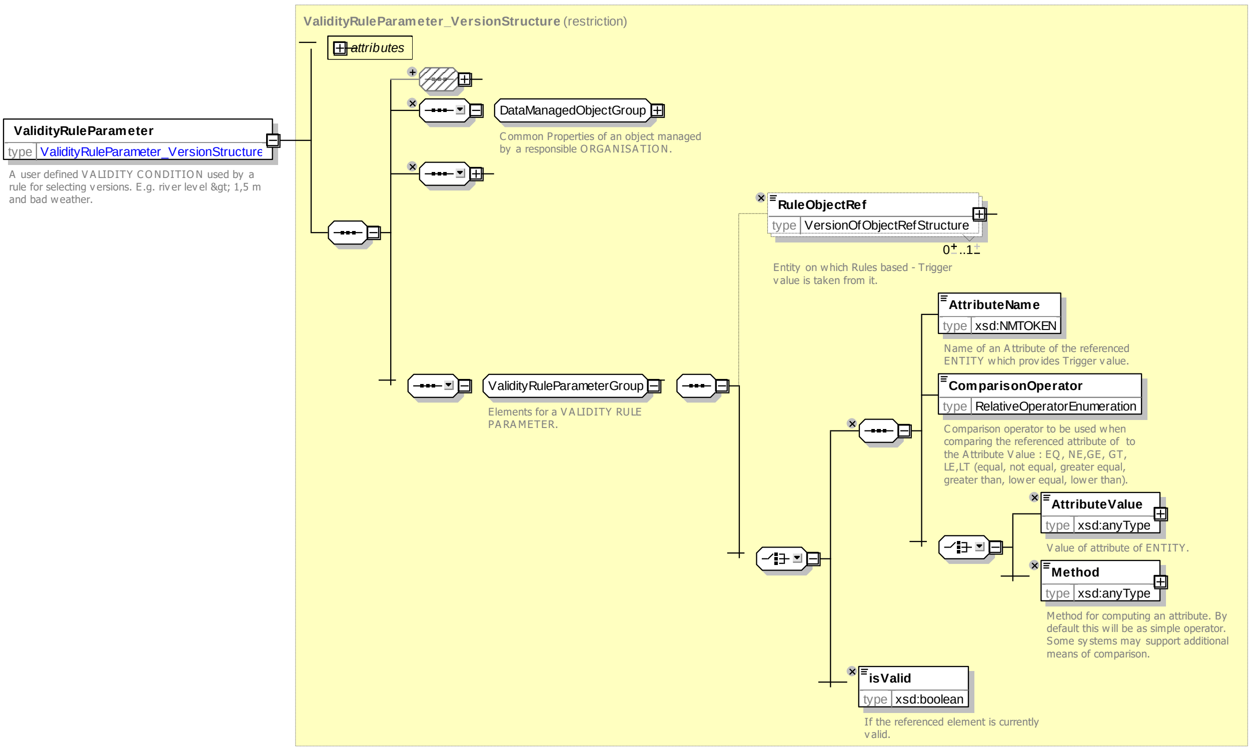


Figure 70 — *ValidityRuleParameter – XSD*

7.3.7.3.3.1 ComparisonOperator – Allowed Values

Allowed values for ***ComparisonOperator*** (*RelativeOperatorEnumeration)*.

Table 43 *ComparisonOperator* – Allowed values

Value	Description
EQ	Equal.
NE	Not equal.
GE	Greater than or equal.
GT	Greater than.
LE	Less than or equal.
LT	Less than.

7.3.7.3.4 ValidityTrigger – Model Element

External event defining a VALIDITY CONDITION, E.g. exceptional flow of a river, bad weather, road closure for works.

Table 44 — *ValidityTrigger* – Element

Classification	Name	Type	Cardinality	Description
::>		*EntityInVersion*		NB Not allowed a VALIDITY

::>	::>	*ValidityCondition*	::>	VALIDITY TRIGGER inherits from VALIDITY CONDITION.
«PK»	***id***	*ValidityTriggerIdType*	1:1	Identifier of VALIDITY TRIGGER.
«FK»	***TriggerObjectRef***	*ObjectRef*	0:1	ENTITY on which VALIDITY TRIGGER is based – the trigger value is taken from it.
	PrivateCode	*PrivateCodeStructure*	0:1	Private code for VALIDITY TRIGGER.

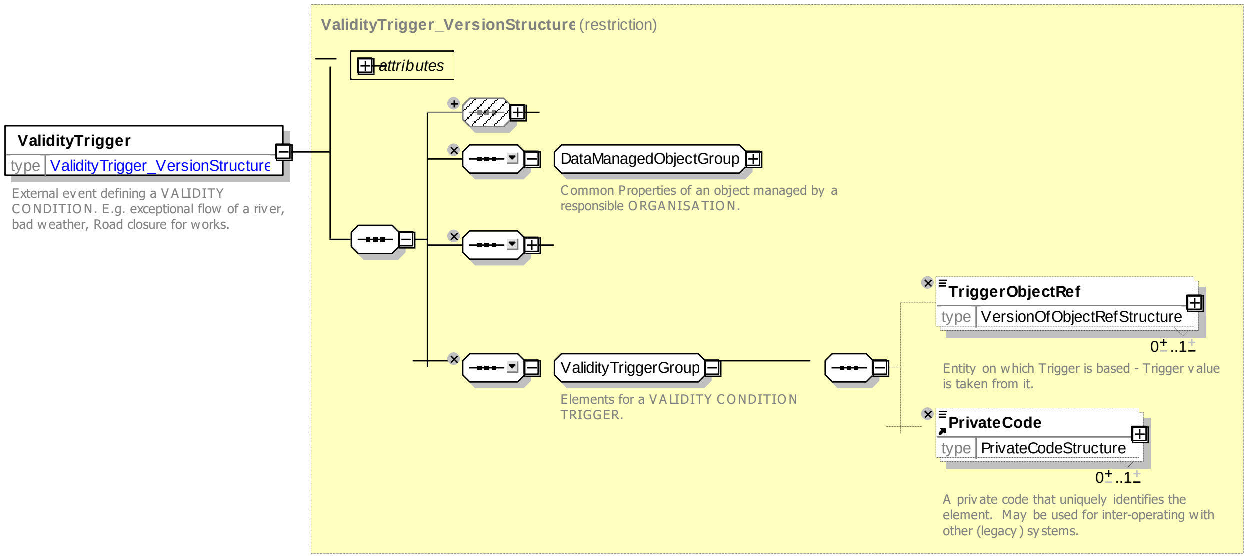


Figure 71 — *ValidityTrigger – XSD*

7.3.7.4 *XML Examples of Validity Conditions*

The following example shows the use of validity Conditions attached to a frame. In this case the Timetable frame is the weather is icy and the day type is in winter

EXAMPLE

```
    <ServiceTimetableFrame version="any" id="hde:SCF01">
            <frameValidityConditions>
                <ValidityRuleParameter version="any" id="hde:VC_ntwkf003_002_icy">
                    <Description>Use when Icy</Description>
                    <ConditionedObjectRef nameOfRefClass="TimetableFrame" ref="abc:Tf01"/>
                    <RuleObjectRef                               nameOfRefClass="Weather"
ref="Myobj:Cur001">EXTERNAL</RuleObjectRef>
                    <AttributeName>roadCondition</AttributeName>
                    <ComparisonOperator>EQ</ComparisonOperator>
                    <AttributeValue>icy</AttributeValue>
                    <WithConditionRef version="any" ref="hde:rp_02"/>
                </ValidityRuleParameter>
                ref="hde:ValidityRuleParameter:VC_ntwkf003_002_winter"/>
                </ValidityRuleParameter>
                <ValidityRuleParameter version="any" id="hde::VC_ntwkf003_002_winter">
                    <Description>Use when Winter</Description>
                    <ConditionedObjectRef nameOfRefClass="TimetableFrame" ref="abc:Tf01"/>
                    <RuleObjectRef nameOfRefClass="DayType" ref="Myobj:DT001" version="1"/>
                     <isValid>true</isValid>
                </ValidityRuleParameter>
            < frameValidityConditions >
            <contentValidityConditions>
                <ValidityTrigger version="any" id="hde:vt_01">
```

```
                <Description>Special event day</Description>
                <ConditionedObjectRef nameOfRefClass="TimetableFrame" ref="abc:Tf01"/>
            </ValidityTrigger>
        </contentValidityConditions>

.........
        <dayTypes>
            <DayType version="any" id="Myobj:DT001">
                <Name>Winter </Name>
                <properties>
                    <PropertyOfDay>
                        <Name>Winter </Name>
                        <Seasons>Winter</Seasons>
                    </PropertyOfDay>
                </properties>
            </DayType>
        </dayTypes>
```

See Availability Condition for further examples.

7.4 Responsibility

7.4.1 Introduction

NeTEx data will be used in passenger information environments that can have a complex organisational structure. For instance, information is planned, revised, forwarded, enriched, combined with other plans and forwarded again to the final user at some time. This process often involves several organisations or departments that each add, change or remove information in a complex workflow. These participating organisations can be strictly PT concerns, or also be external, such as governmental departments or other management agents. Which organisations are involved, what roles they have and what responsibility they execute cannot be determined beforehand for all possible environments in which NeTEx will be used. Even the structure and implementation of the processes for information planning, collecting and forwarding depend on various factors that cannot be determined beforehand. Hence, NeTEx has a generic organisational and responsibility model that can be applied in a variety of different environments and workflows and be used for a variety of purposes. The model in effect defines metadata as to the ownership of data that can be used to help manage the data.

The use of the responsibility model in a specific situation or environment is optional.

The responsibility model makes it possible:

— To define operational responsibility for the real-life entities that are described by the information. For example, processes for a Stop Place information model it can specify which organisation is responsible for planning and maintenance of the physical stop.

— To define data management related responsibilities for the information itself. E.g. functional or technical IT data management regarding a set of produced, collected or forwarded plan information. This can be used to identify who needs to be contacted to correct or amend data.

— To exchange partial information falling under a certain responsibilty set.

If used, the responsibility model can be applied to achieve the following goals:

— Provide as part of the passenger information the contact information of agencies or help-desks to turn to in case of reservations, questions, complaints, etc.

— Provide IT and PT related responsibility information for the purpose of management, assessment, etc. activities concerning quality Management and Quality Control.

— Associate Intellectual Property Rights with individual data elements or groups of elements.

— Allow delegation of data management: a receiving system can check the authorizations in relation to responsibility for provided data and see if the provider is authenticated to manage that data. This concept can be used to protect data in VERSION FRAMEs from being changed by the wrong parties.

7.4.2 Responsibility – Model Dependencies

The Responsibility Model forms part of the core NeTEx framework and is referenced by all other parts. There are three sub models. They extend the basic Entity & Versioning models:

— The core RESPONSIBILITY model describes basic tracking attributes.

— The RESPONSIBILITY ROLE model describes the roles different organisations may take.

— The ORGANISATION model defines the common structures of an organisation. Note that this is further extended in the Reusable Components model (see later) with specific classes for specific types of organisation such as OPERATOR, AUTHORITY, SERVICED ORGANISATION, etc.

The Responsibility Model extends the basic Entity & Versioning models to create the fundamental framework classes from which all the useful NeTEx models are built.

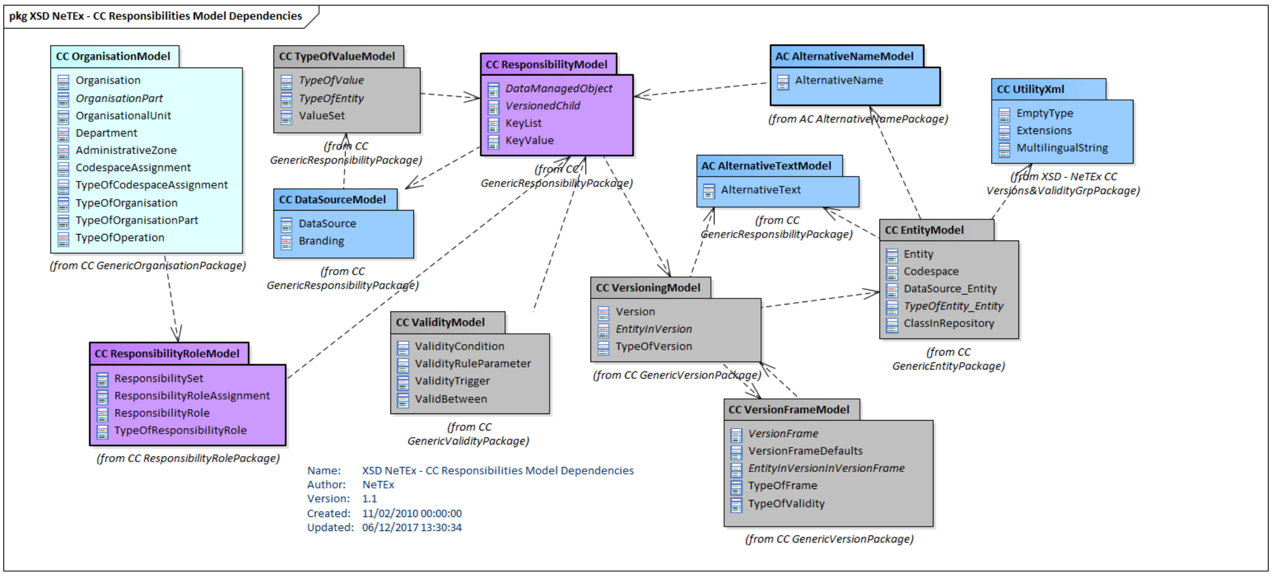


Figure 72 — Responsibility – Model Dependencies (UML)

7.4.3 Generic Responsibility

7.4.3.1 Generic RESPONSIBILITY – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

A certain aspect, or set of aspects, of responsibility in relation to an ENTITY is specified by associating a RESPONSIBILITY SET with the ENTITY. Each RESPONSIBILITY SET can contain one or more RESPONSIBILITY ROLE ASSIGNMENTs that allocate different types of RESPONSIBILITY ROLE to an ORGANISATION or a specific ORGANISATION PART.

RESPONSIBILITY SETs may be used at different levels of aggregation. It is possible to specify a different set for each different ENTITY (or rather ENTITY IN VERSION), or just at the Frame Level. The RESPONSIBILITY SET for an ENTITY may change in successive ENTITY IN VERSIONs.

The RESPONSIBILITY ROLE describes the kind of responsibility that is enacted; the RESPONSIBILITY ROLE ASSIGNMENT assigns the responsibility to the RESPONSIBILITY SET.

The ADMINISTRATIVE ZONE and RESPONSIBILITY ROLE ASSIGNMENT are used to describe the specific situation of the delegation of the regional responsibility of an authority to an organisation. This can be e.g. the delegation using a concession for the operation of a PT service or the delegation of a regional travel information provision service.

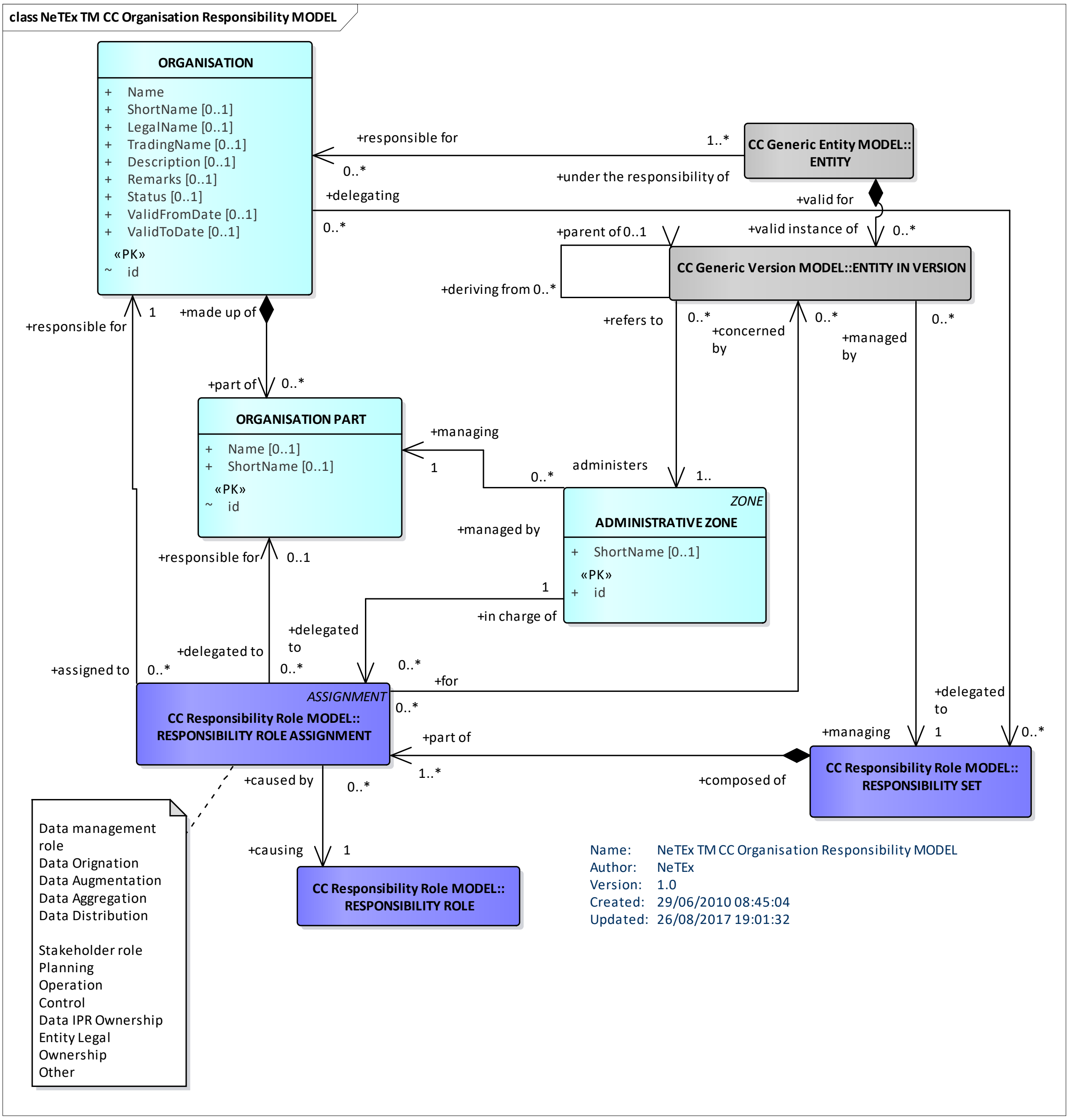


Figure 73 — Responsibility – Conceptual MODEL (UML)

7.4.3.2 Generic Responsibility – Physical Model

7.4.3.2.1 Summary of Responsibility model

The following diagram summarises the Physical model. All NeTEx entities are instances either of DATA MANAGED OBJECT or VERSIONED CHILD.

Only DATA MANAGED OBJECT may have its own RESPONSIBILITY SET. For a composite object, such as a SERVICE PATTERN made up of a number of dependent POINT IN PATTERN instances, all the child elements (which will be a subtype of VERSIONED CHILD) must take the same RESPONSIBILITY SET as their parent.

The detailed attributes of the Responsibility Physical Model are shown in separate further diagrams.

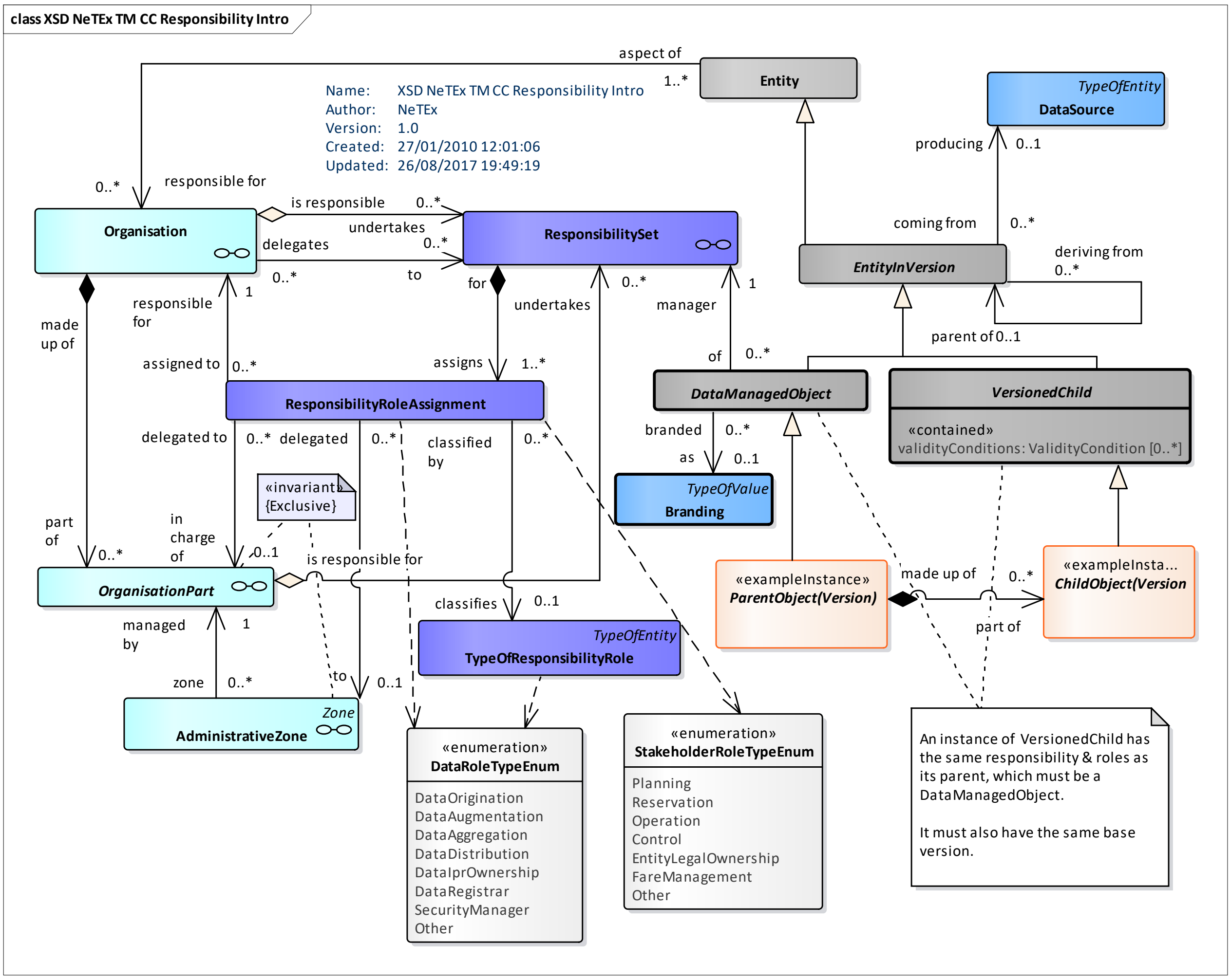


Figure 74 — Responsibility Overview – Physical Model (UML)

7.4.3.2.2 Assigning Responsibilities to an Entity

The next figure shows how a RESPONSIBILITY SET can be associated with an ENTITY IN VERSION by using one of two different specialisations of ENTITY IN VERSION.

DATA MANAGED OBJECT is subtype of an ENTITY IN VERSION for which a specific RESPONSIBILITY SET can be specified.

A VERSIONED CHILD is a component of a DATA MANAGED OBJECT that takes the same responsibility as its parent.

The same RESPONSIBILITY SET may be assigned to many or all of the DATA MANAGED OBJECTs in a frame.

If a specific RESPONSIBILITY SET is not specified for an element, it is assumed to have the same RESPONSIBILITY SET as the frame or other context that contains it.

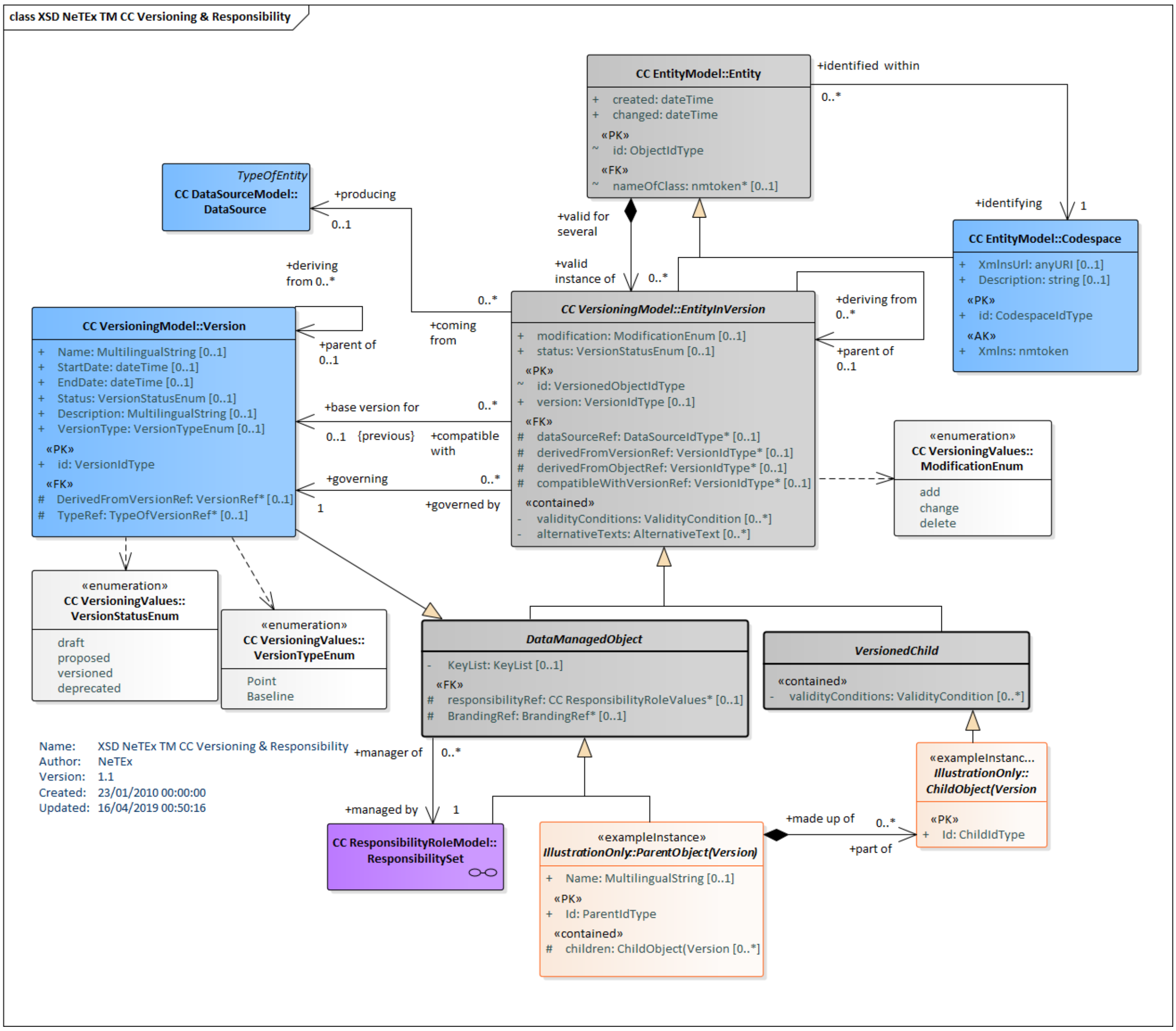


Figure 75 — Responsibility & Versions – Physical Model (UML)

7.4.3.2.3 Assigning Responsibilities to an Entity

The second figure shows how a given RESPONSIBILITY SET associates different ORGANISATIONs or ORGANISATION PARTs with specific roles by means of one or more RESPONSIBILITY ROLE ASSIGNMENTs.

A RESPONSIBILITY SET is itself a DATA MANAGED OBJECT and may be updated and versioned.

A given RESPONSIBILITY SET may be shared by many different entity instances and by many different types of entity.

The choice of division into different RESPONSIBILITY SETs will normally reflect the choice of metadata to support the required workflow. The respective elements can be combined in different ways to represent the different administrative processes (sometimes central, sometimes distributed) found in different countries.

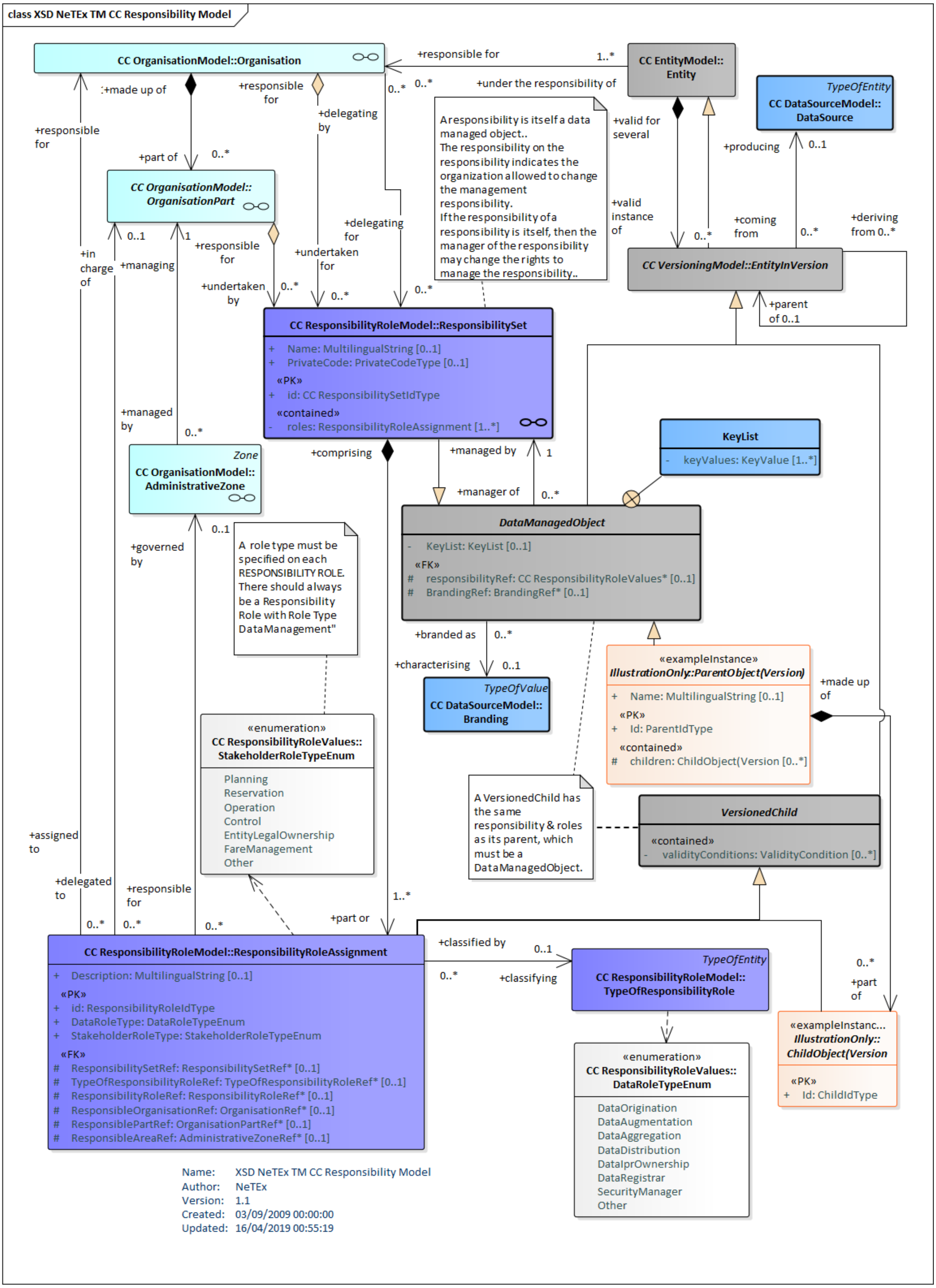


Figure 76 — Responsibility Role Assignments – Physical Model (UML)

7.4.3.2.3.1 Example of RESPONSIBILITY SETs

For example, in the UK, the NPTG (Nation Public Transport Gazetteer) corresponds to a centrally managed set of RESPONSIBILITY SETs managed by the Department of Transport that describe how to coordinate the management of stop data.

- For managing most types of stop data (i.e. for bus stops, airports, ferry stops etc) the country is divided in regions and areas within regions. This can be indicated by a RESPONSIBILITY SET for each area; each set is associated with an ADMINISTRATIVE ZONE that designates the areas boundaries and is used to associate the CODESPACE and prefix to use for stop identifers from that region. Within a designated area, all stop data other than rail station data is collected and mantained by the ORGANIZATION indicated by the RESPONSIBILITY SET (usually an AUTHORITY). In this case the ADMINISTRATIVE ZONEs do not overlap.

- Certain types of stop data, for example rail stations are maintained centrally for the whole country. There is a RESPONSIBILITY SET for each type of data that associates it with the appropriate organization and zone. The ADMINISTRATIVE ZONEs overlap the zones for other types of stop data.

- A single RESPONSIBILITY SET defines the Department of Transport central responsibility for creating all the other RESPONSIBILITY SETs. Another RESPONSIBILITY SET defines the responsibility of a contracting organisation to aggregate and distribute the data.

7.4.3.2.4 Type of Value – Physical Model

It is useful to be able to define arbitrary code values as categories to classify various entities in various ways. For example, TYPE OF POINT, TYPE OF LINK, PURPOSE OF GROUPING, etc. are all examples of such classifications. TYPE OF VALUE provides an abstract class with common properties such as a Name, Description and URL that can be used to define code values and to exchange them. It is specialised to create specific named types of codes. Thus for example, TYPE OF POINT is a subclass of TYPE OF VALUE. If appropriate additional attributes may be added to the specialisation.

NOTE The Transmodel term “Type of” is something of a misnomer – it does not correspond to the computer concept of a type but is rather indications a classification attribute of a given class.

For the purposes of data exchange, sets of instances of the same type of TYPE OF VALUE may be grouped in a named VALUE SET. See example below.

VALUE SET & TYPEs OF VALUEs are DATAMANAGED OBJECTs and so can be versioned so that updates to the code set can be made from time to time.

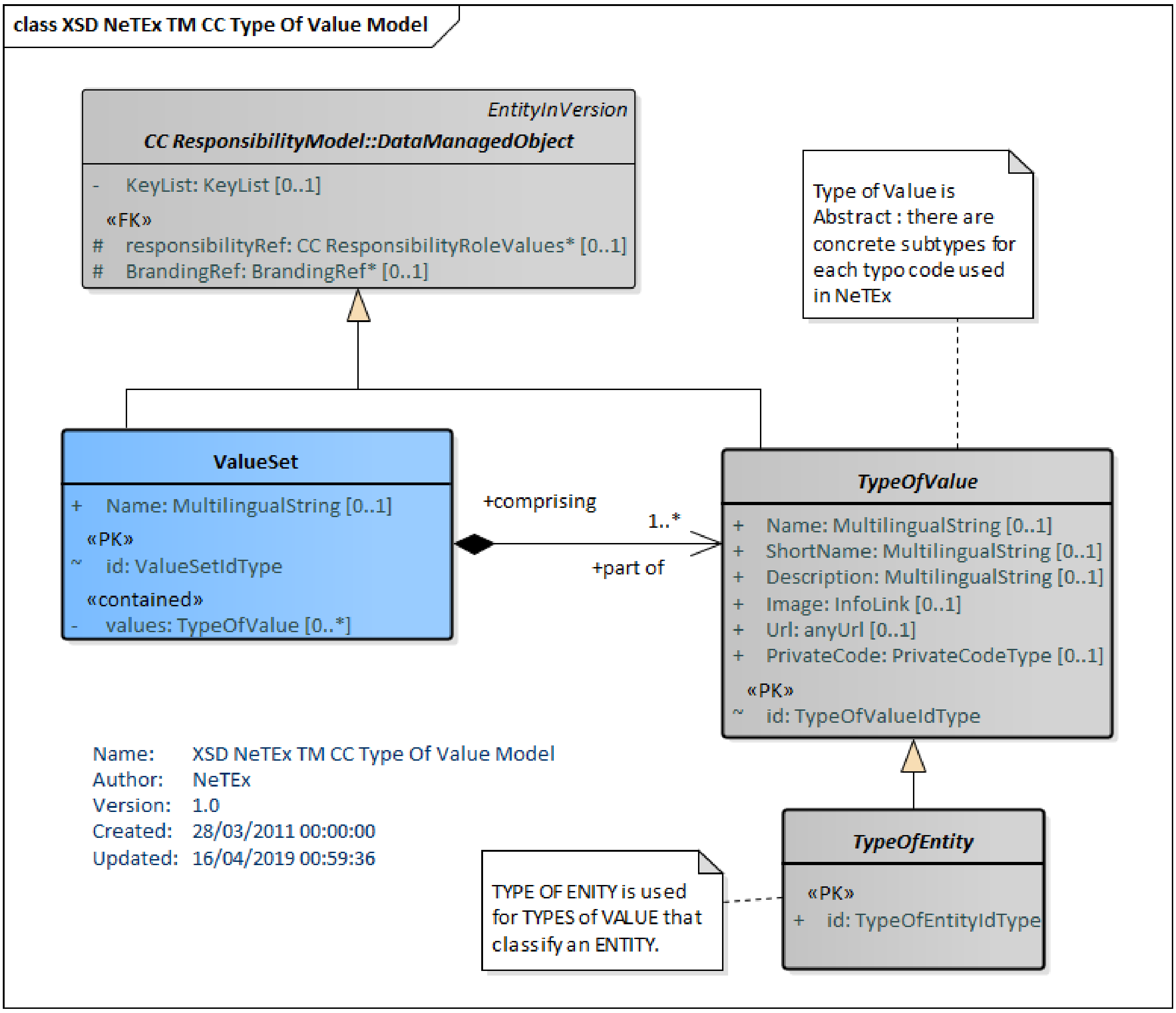


Figure 77 — Type Of Value - Physical Model (UML)

The following figures summarises the descendants of Types of Value found in NeTEx.

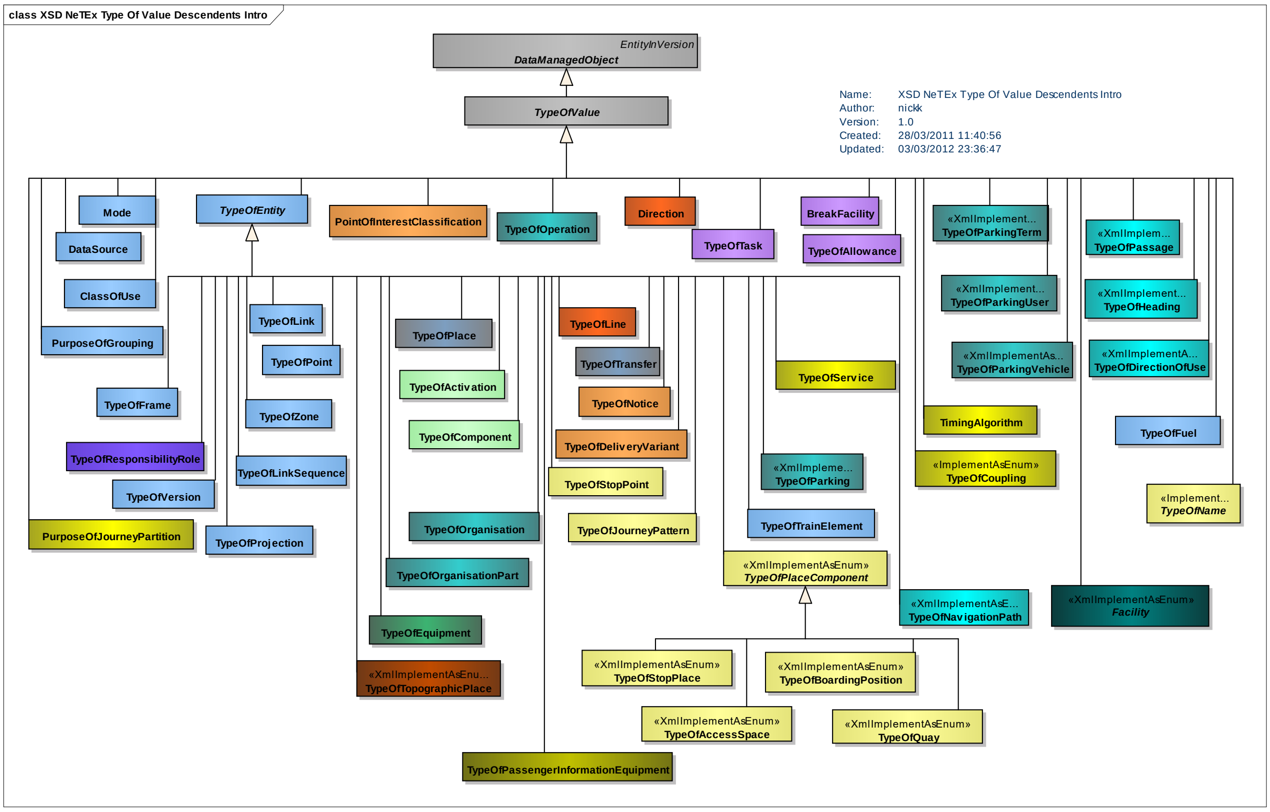


Figure 78 — Type Of Value Descendants - Physical Model (UML)

7.4.3.2.5 Version Object Hierarchy

At this point, having introduce all the fundamental elements, it may be useful to summarise by looking at the overall object hierarchy for ENTITies and VERSIONs in order to see the full set of common inherited attributes found on NeTex physical elements.

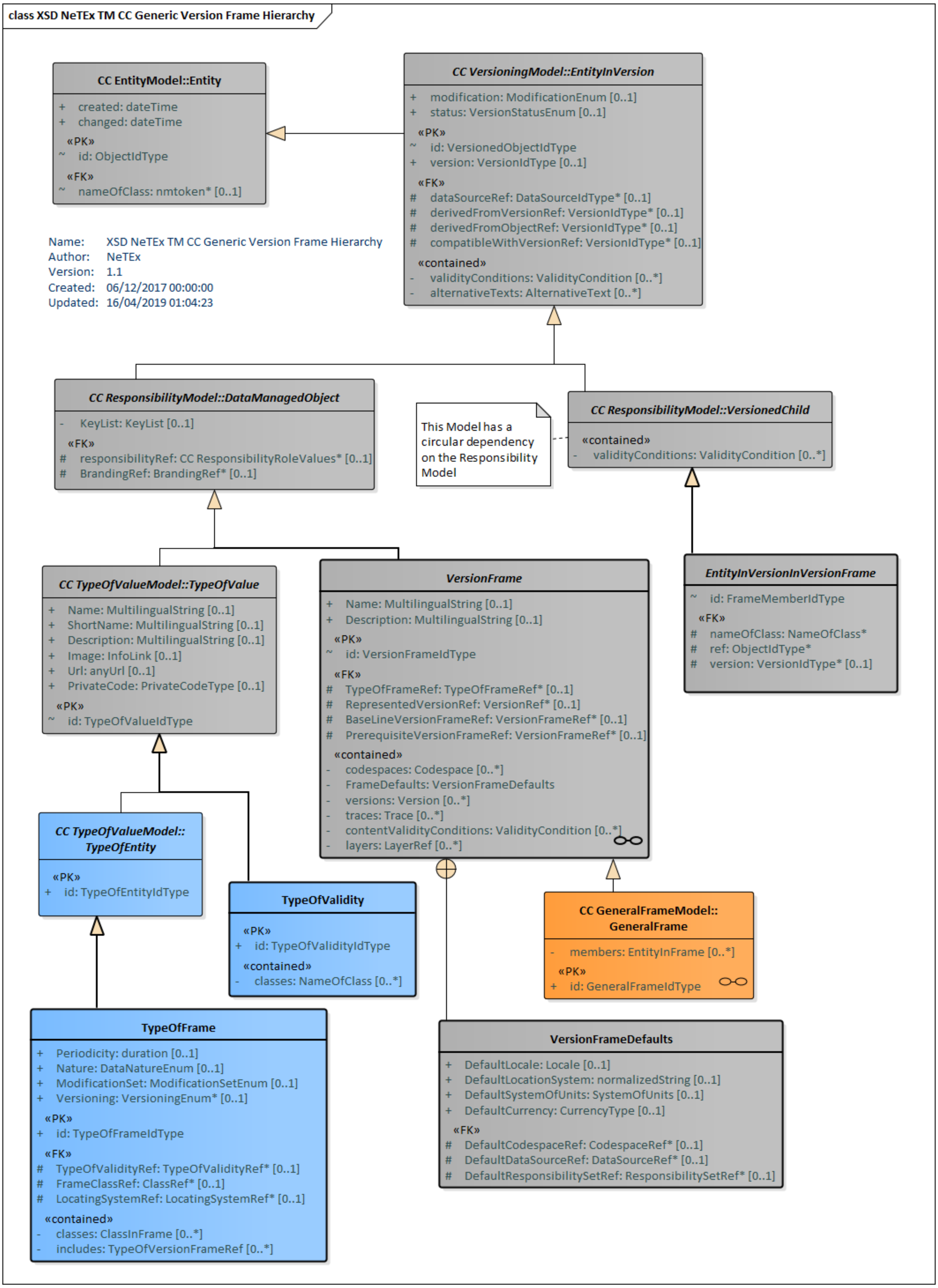


Figure 79 — NeTEx Core Framework — Object Hierarchy (UML)

7.4.3.3 Generic Responsibility – Attributes and XSD

7.4.3.3.1 DataManagedObject – Model Element

An ENTITY in VERSION can be associated with a RESPONSIBILITY SET. The RESPONSIBILTY SET in turn has one or more RESPONSIBILTY ROLE ASSIGNMENTs that can describe responsibilities for the element, for instance who has the responsibility for managing the data.

DataManagedObject also specifies common properties of all first-class NeTEX elements.

— A RESPONSIBILITY SET, specifying data management roles.

— The optional association of an arbitrary list of key/value pairs to provide a simple extensibility mechanims.

— An ***Extensions*** tag that allows the association of arbitrary user defined XML elements with an element.

— A reference to a branding associated with the corresponding ENTITY

Table 45 — *DataManagedObject* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*EntityInVersion*</u>	::>	DATA MANAGED OBJECT inherits from ENTITY IN VERSION.
«atr», «FK»	***responsibilitySetRef***	*ResponsibilitySet-IdType*	1:0	Reference to RESPONSIBILITY SET defining ownership and management responsibilities for object.
«cntd»	***keyList***	<u>*KeyList*</u>	0:1	A list of key value pairs that may be associated with the object. This allows simple arbitrary user extensions
«cntd»	***Extensions***	*ExtensionStructure*	0:1	Wrapper for arbitrary user defined extension elements associated with element.
«FK»	***BrandingRef***	*BrandingRef*	0:1	Reference to a BRANDING.

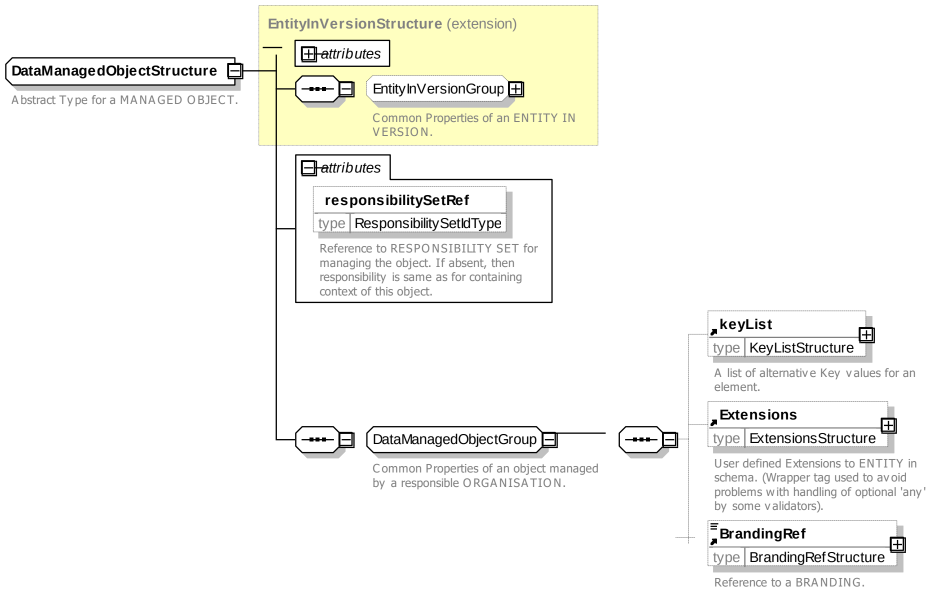


Figure 80 — *DataManagedObject – XSD*

7.4.3.3.2 VersionedChild – Model Element

A child ENTITY whose RESPONSIBILITY SET must be the same as its containing parent object, and which cannot exist independently of its parent in a repository, for example a POINT IN PATTERN. Thus in practice if the parent is deleted, so will the child be.

Table 46 — *DataManagedObject* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*EntityInVersion*	::>	VERSIONED CHILDOBJECT inherits from ENTITY IN VERSION.
«cntd»	***Extensions***	*ExtensionStructure*	0:1	Wrapper for arbitrary user defined extension elements associated with element.

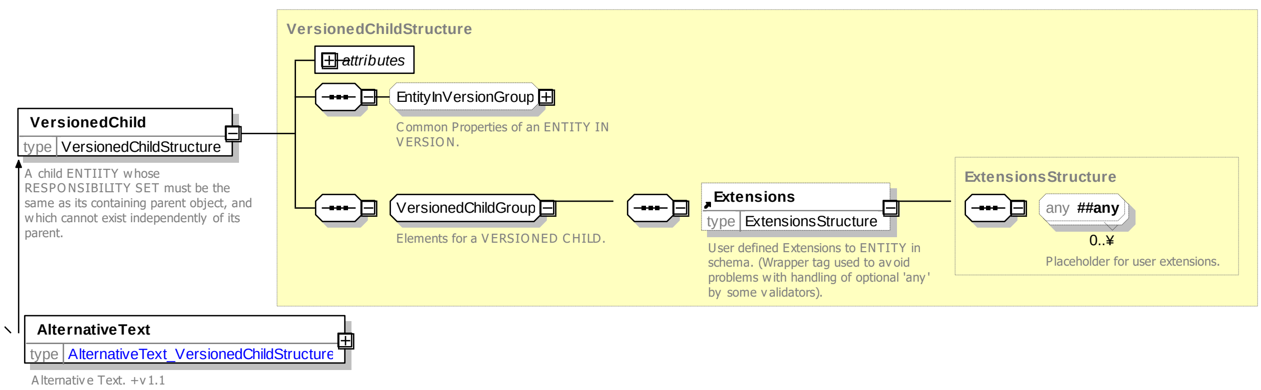


Figure 81 — *VersionedChild – XSD*

7.4.3.3.3 TypeOfValue – Model Element

A coded value with a name and other properties. TYPE OF VALUE is specialized to make specific sets of codes used to classify other NeTEx entities. TYPEs OF VALUE can be used to exchange metadata used to validate or collect data, such as the description and allowed values for codes. Concrete examples are TYPE OF POINT, TYPE OF LINK, TYPE OF SERVICE, etc.

Table 47 — *TypeOfValue* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	TYPE OF VALUE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*TypeOfValueIdType*	1:1	Identifier of TYPE OF VALUE.
	Name	*MultilingualString*	0:1	Name of TYPE OF VALUE.
	ShortName	*MultilingualString*	0:1	Short Name of TYPE OF VALUE.
	Description	*MultilingualString*	0:1	Description of TYPE OF VALUE.
	Image	*xsd:anyURI*	0:1	Default image for TYPE OF VALUE.
	Url	*xsd:anyURI*	0:1	URL associated with of TYPE OF VALUE.
	PrivateCode	*PrivateCode*	0:1	Private Code associated with TYPE OF VALUE.

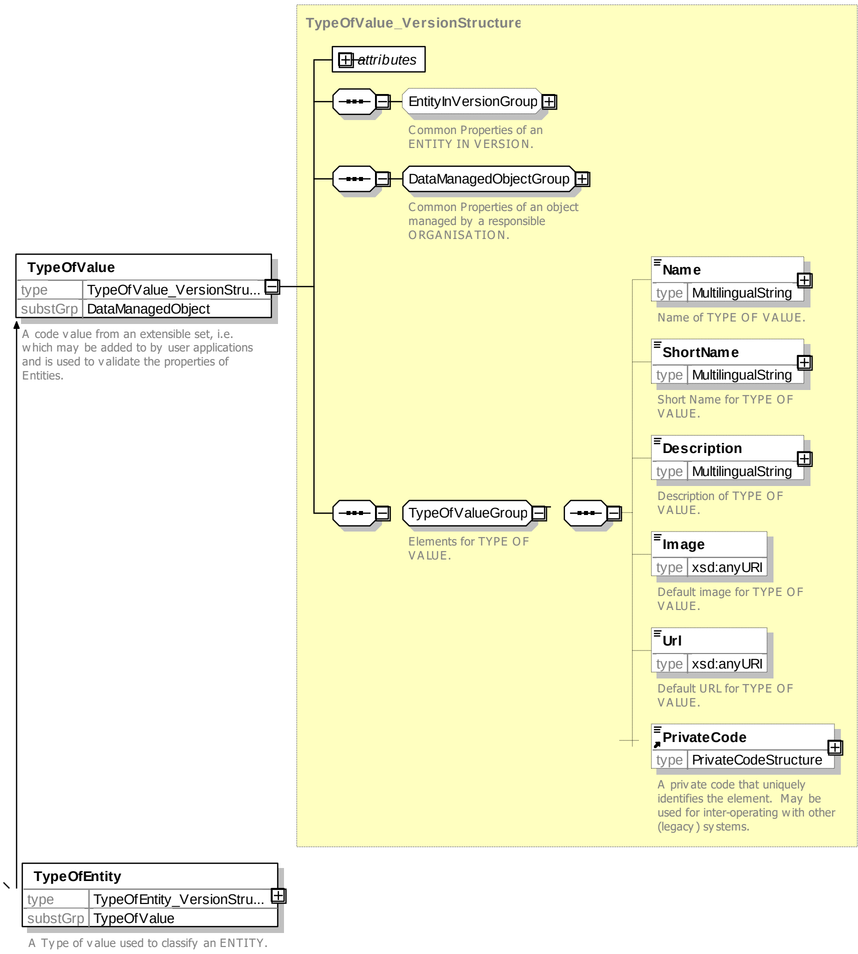


Figure 82 — *TypeOfValue – XSD*

7.4.3.3.4 DataSource – Model Element

The DATA SOURCE identifies the system that has produced the data.

References to a DATA SOURCE are useful in an interoperated computer system to identify the origin of data.

Table 48 *DataSource* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	DATA SOURCE inherits from TYPE OF VALUE.
«PK»	***id***	*DataSourceIdType*	1:1	Identifier of DATA SOURCE.
	Email	*EmailAddressType*	0:1	Contact email address for DATA SOURCE.

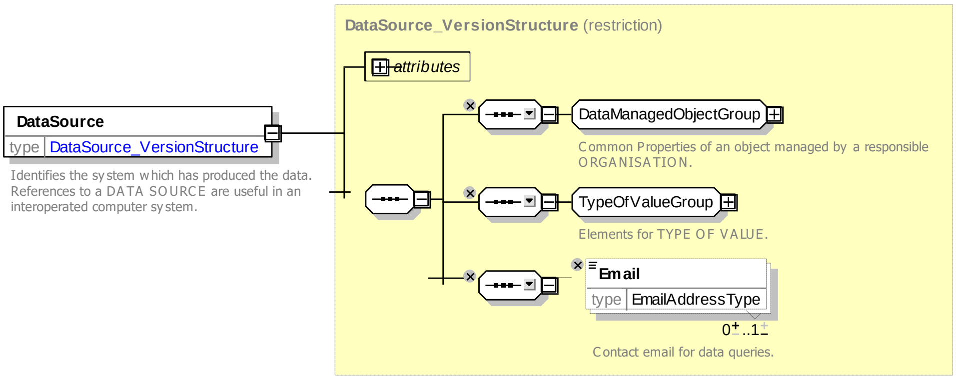


Figure 83 — *DataSource – XSD*

7.4.3.3.5 ValueSet – Model Element

A grouping of instances of a specific TYPE OF VALUE instances for the purposes of exchange (i.e. a list of codes).

Table 49 — *ValueSet* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VALUE SET inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ValueSetIdType*	1:1	Identifier of VALUE SET.
	Name	*MultilingualString*	0:1	Name of VALUE SET.
«cntd»	***values***	*TypeOfValue*	0:*	TYPES OF VALUE in VALUE SET.

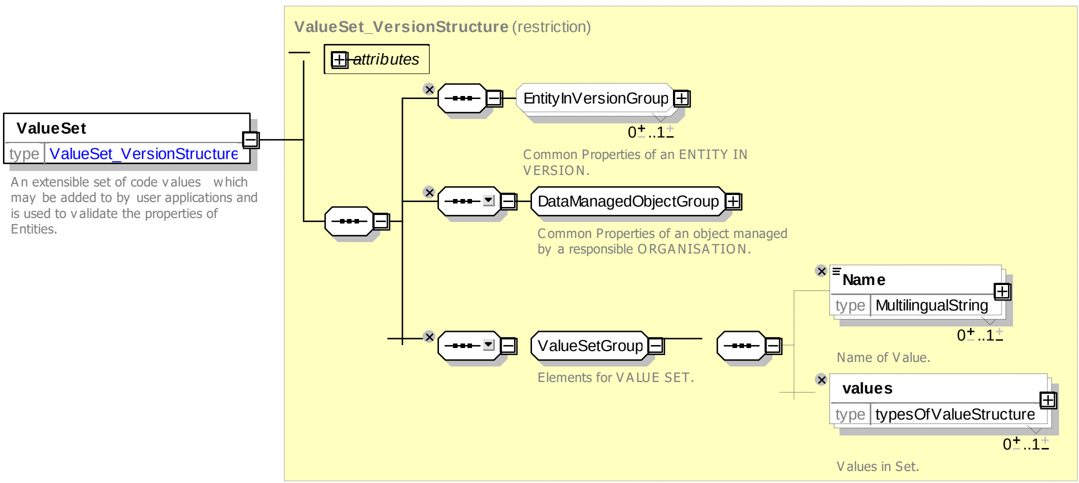


Figure 84 — *ValueSet – XSD*

7.4.3.4 *XML Examples of Responsibility*

7.4.3.4.1 Assigning a ResponsibilitySet to an element – XML Example fragment

The following example shows the use of a RESPONSIBILITY SET to define the data management for a STOP PLACE and its child elements e.g. ACCESSIBILITY ASSESSMENT. See the later section on RESPONSIBILITY ROLEs for an example of defining a RESPONSIBILITY SET.

EXAMPLE

```xml
<StopPlace version="any" responsibilitySetRef="napt:RS_147" id="napt:940GZZLUWHM">
    <Name>West Ham Underground Station </Name>
        <ShortName>West Ham</ShortName>
        <types>
            <TypeOfZoneRef version="any" ref="napt:TypeOfZone:GTMU"/>
        </types>
        <AccessibilityAssessment version="any" id="napt:940GZZLUWHM">
            <MobilityImpairedAccess>true</MobilityImpairedAccess>
            <limitations>
                <AccessibilityLimitation created="2010-05-17T09:30:47Z" modification="new">
                    <WheelchairAccess>true</WheelchairAccess>
                    <StepFreeAccess>true</StepFreeAccess>
                    <EscalatorFreeAccess>true</EscalatorFreeAccess>
                    <LiftFreeAccess>true</LiftFreeAccess>
                    <AudibleSignalsAvailable>false</AudibleSignalsAvailable>
                    <VisualSignsAvailable>true</VisualSignsAvailable>
                </AccessibilityLimitation>
            </limitations>
        </AccessibilityAssessment>
        <TransportMode>metro</TransportMode>
        <StopPlaceType>metroStation</StopPlaceType>
```

7.4.3.4.2 Declaring a ValueSet of user defined Types Of Values – XML Example fragment

The following example shows the use of a VALUE SET to define some arbitrary code values.

EXAMPLE

```xml
<!--- ==== Subset of NaPTAnN Stop Types===== -->
    <ValueSet version="any" id="napt:ValueSet:StopTypes">
        <Name>UK NaptAN Stop Types</Name>
        <values>
            <TypeOfZone version="any" id="napt:TMU">
                <Name>Tram or Metro stop</Name>
            </TypeOfZone>
            <TypeOfZone version="any" id="napt:RSE">
                <Name>Rail Station Entrance</Name>
            </TypeOfZone>
            <TypeOfZone version="any" id="napt:BCT">
                <Name>On Street Bus</Name>
            </TypeOfZone>
        </values>
    </ValueSet>
```

The values defined in the set may be referenced by other entities and will be validated by an XML validator. For example, in the following fragment the ***TypeOfZoneRef*** value '*napt: GTMU'* will be validate against the declared value.

EXAMPLE

```xml
    <StopPlace id="napt:940GZZLUWHM">
        <Name>West Ham Underground Station </Name>
        <ShortName>West Ham</ShortName>
        <Centroid>
            <Location>
                <Longitude>-0.2065219984</Longitude>
                <Latitude>51.4213610557</Latitude>
            </Location>
        </Centroid>
```

```
        <types>
            <TypeOfZoneRef version="any" ref="napt:GTMU"/>
        </types>
        <TransportMode>metro</TransportMode>
    <StopPlaceType>metroStation</StopPlaceType>
```

If the version is omitted on the reference, then a check for referential integrity will **not** be made and the referenced valued does not need to be declared. For example, in the following fragment the ***TypeOfZoneRef*** value '*napt: GTMU'* will **not** be validate against the declared value.

EXAMPLE

```
    <StopPlace id="napt:940GZZLUWHM">
        <Name>West Ham Underground Station </Name>
        <ShortName>West Ham</ShortName>
        <Centroid>
            <Location>
                <Longitude>-0.2065219984</Longitude>
                <Latitude>51.4213610557</Latitude>
            </Location>
        </Centroid>
        <types>
            <TypeOfZoneRef ref="napt:GTMU"/>
        </types>
        <TransportMode>metro</TransportMode>
    <StopPlaceType>metroStation</StopPlaceType>
```

7.4.4 Responsibility Role

7.4.4.1 RESPONSIBILITY ROLE – Conceptual MODEL

The RESPONSIBILITY ROLE model describes the specific properties of a RESPONSIBILITY SET as a set of assignments of specific roles to specific ORGANISATIONs or ORGANISATION PARTs.

Each RESPONSIBILITY ROLE ASSIGNMENT allocates a specific role to a specified ORGANISATION or ORGANISATION PART.

A full information delivery chain for Travel Information could involve multiple actors. This model will allow identifying the different roles actor can have in such a multi-organisation process.

As different aspects of public transport could be handled by different organisation parts and sometimes are subcontracted to third parties, it is often useful to describe who is responsible for a specific role, within the delivered data.

Examples of roles are:

- Data Orignation
- Data Augmentation
- Data Aggregation
- Data Distribution
- Planning
- Operation
- Control Centre (directive PT-management centre)
- Monitor Centre (only receiving and collecting data)
- Data IPR Ownership
- Entity Legal Ownership
- Scheduler,
- StopPointManager,
- RoadManager,
- RoadDisplayManager,
- SubContractor,
- TravelInformationServiceProvider,

— Other

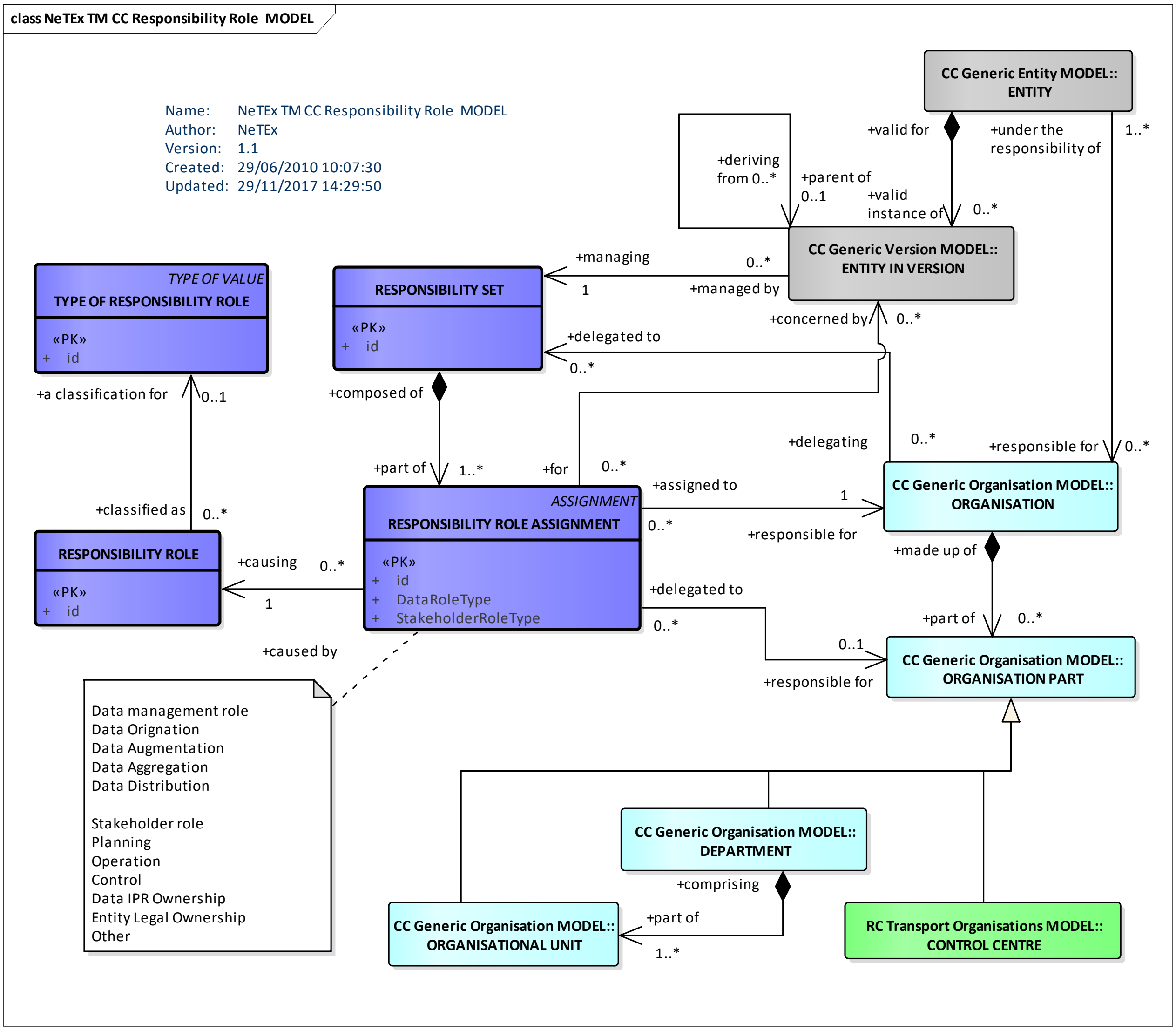


Figure 85 — Responsibility Role – Conceptual MODEL (UML)

7.4.4.2 Responsibility Role – Physical Model

The Physical model provides a number of predefined role types that may be assigned.

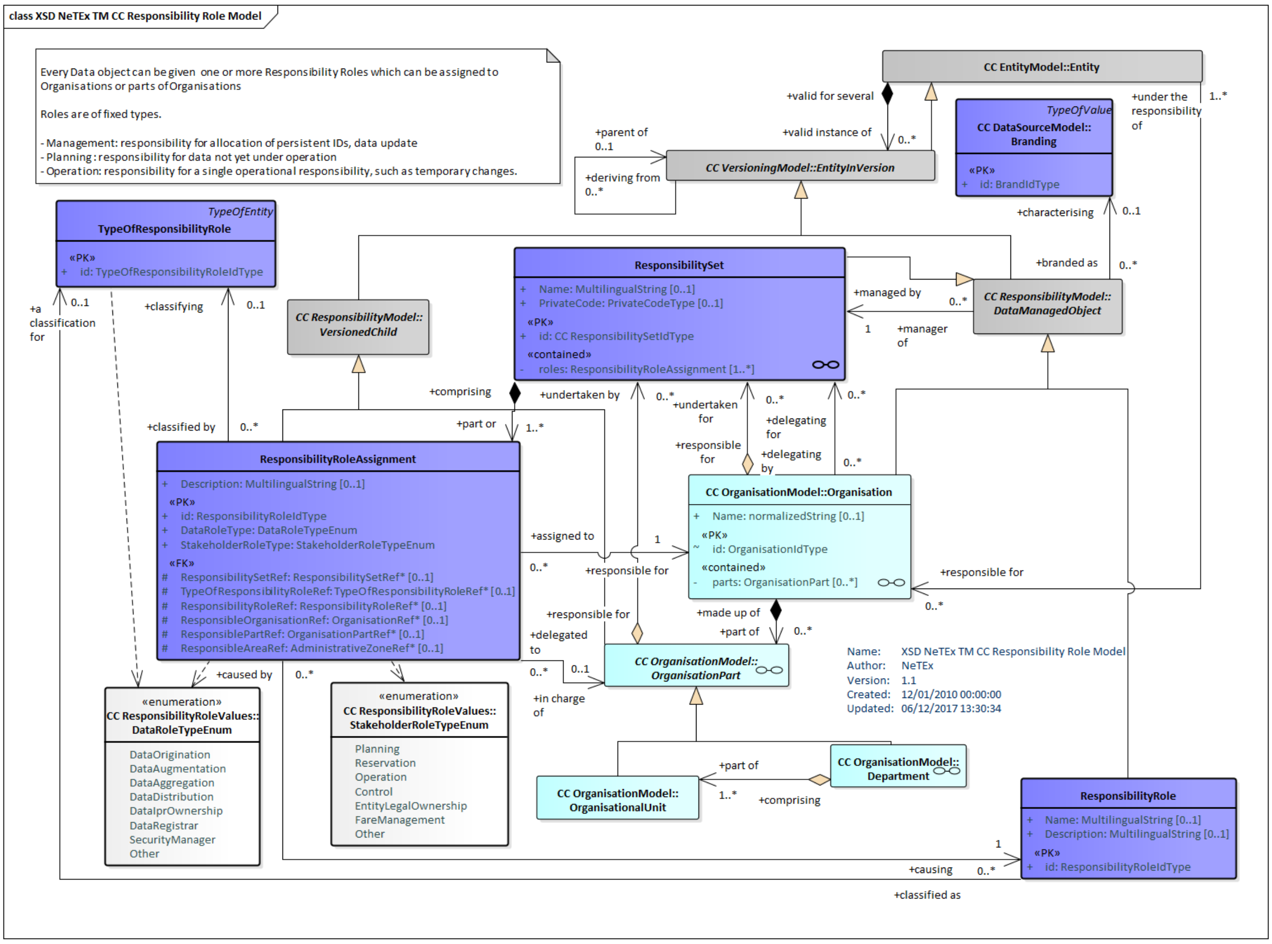


Figure 86 — Responsibility Role – Physical Model (UML)

7.4.4.3 ResponsibilitySet – Attributes and XSD

7.4.4.3.1 ResponsibilitySet – Model Element

In essence a list of responsibilities that applies to one or more ENTITies IN VERSION.

A RESPONSIBILITY SET is composed of one or more RESPONSIBILITY ROLE ASSIGNMENTs.

A RESPONSIBILITY ROLE ASSIGNMENT expresses an assignment of a RESPONSIBILITY ROLE (such as data origination, ownership, etc.) to an ORGANISATION or ORGANISATION PART.

Table 50 — *ResponsibilitySet* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	RESPONSIBILITY SET inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ResponsibilitySetIdType*	1:1	Identifier of RESPONSIBILITY SET.
	Name	*MultilingualString*	0:1	Description of RESPONSIBILITY SET.

	PrivateCode	*PrivateCode*	0:1	A private code that uniquely identifies the element. May be used for inter-operating with other (legacy) systems.
«cntd»	***roles***	*ResponsibilityRoleAssignment*	1:*	RESPONSIBILITY ROLE ASSIGNMENTs making up the RESPONSIBILITY SET.

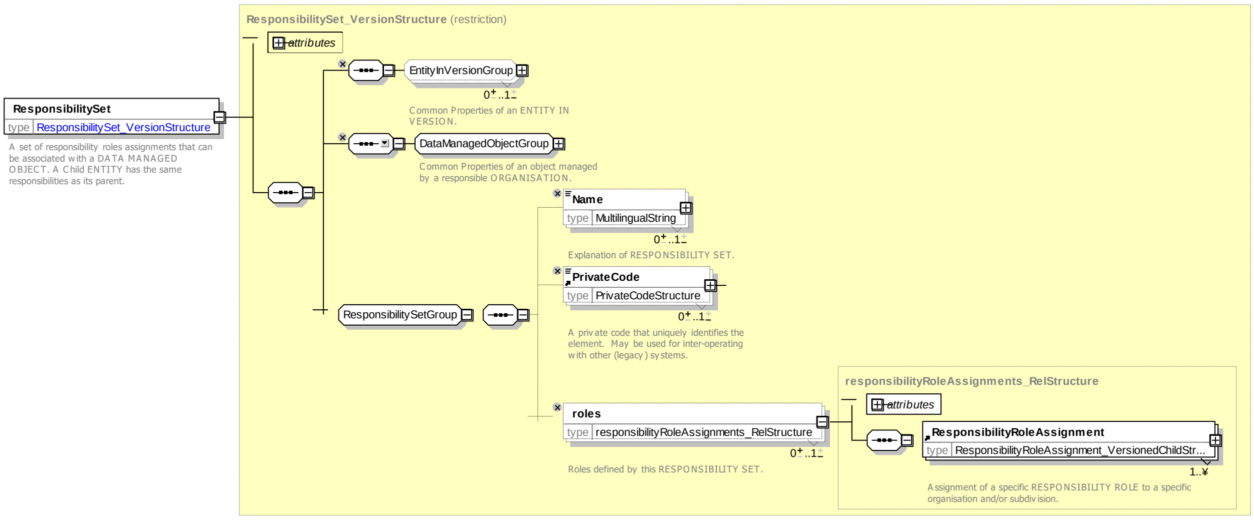


Figure 87 — *ResponsibilitySet – XSD*

7.4.4.3.2 ResponsibilityRoleAssignment – Model Element

The assignment of one or more roles to an ORGANISATION or an ORGANISATION PART as regards the responsibility it will have for specific data (e.g. ownership, planning, etc.) or the management of this data (e.g. distribution, updates, etc.).

Table 51 — *ResponsibilityRoleAssignment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	RESPONSIBILITY ROLE ASSIGNMENT inherits from VERSIONED CHILD.
«PK»	***id***	*ResponsibilityRole-AssignmentIdType*	1:1	Identifier of RESPONSIBILITY ROLE ASSIGNMENT.
«FK»	***ResponsibilitySetRef***	*ResponsibilitySetRef*	0:1	Reference to a RESPONSIBILITY to which this RESPONSIBILITY ROLE ASSIGNMENT belongs.
	Description	*MultilingualString*	0:1	Description of RESPONSIBILITY ROLE ASSIGNMENT.
«enum»	***DataRoleType***	*DataRoleTypeEnum*	0:1	Data Role type of RESPONSIBILITY ROLE. ASSIGNMENT See allowed values below.
«enum»	***StakeholderRoleType***	*StakeholderRoleType-Enum*	0:1	Stakeholder role type of RESPONSIBILITY ROLE.

		CHOICE	0:1	Can be reference or inline
«FK»	*a* ***TypeOf-Responsibility-RoleRef***	*TypeOfResponsibility-RoleRef*	0:1	Reference to a TYPE OF RESPONSIBILITY ROLE.
«ctnd»	*b* ***Responsibility-RoleRef***	*ResponsibilityRoleRef*	0:1	Reference to a RESPONSIBILITY ROLE.
«FK»	***Responsible-OrganisationRef***	*(OrganisationRef)*	0:1	Reference to an ORGANISATION to which this RESPONSIBILITY ROLE is assigned.
«FK»	***ResponsiblePartRef***	*OrganisationPartRef*	0:1	Reference to an ORGANISATION PART to which this RESPONSIBILITY ROLE is assigned.
«FK»	***ResponsibleAreaRef***	*AdministrativeZoneRef*	0:1	Reference to an ADMINISTRATION ZONE to which this RESPONSIBILITY ROLE is assigned.

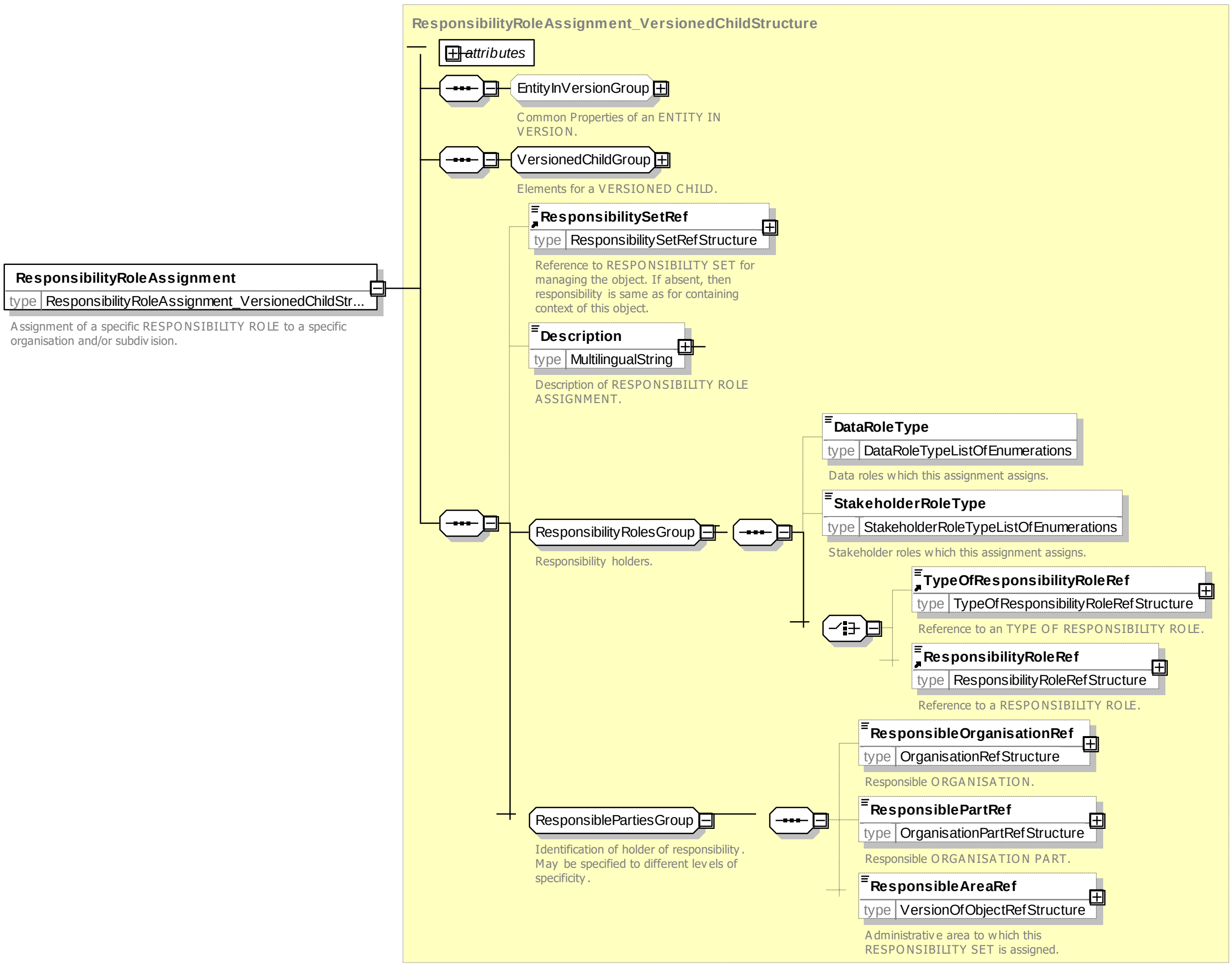


Figure 88 — *ResponsibilitySet – XSD*

7.4.4.3.2.1 DataRoleType – Allowed Values

Allowed values for ***DataRoleType*** (*DataRoleTypeEnumeration*)).

Table 52 — *DataRoleType*– Allowed Values

Value	Description
all	All roles
creates	Role is to create new data instances.
augments	Role is to edit and add data instances from other parties.
validates	Role is to validate data
collects	Role is to collect data instances.
aggregates	Role is to aggregate data instances from other parties.
distributes	Role is to distribute aggregated data instances to other parties.
secures	Role is to secure the data.
owns	Role is to legally own the Intellectual Property Rights.
Other	Other Role.

7.4.4.3.2.2 StakeholderRoleType – Allowed Values

Allowed values for ***StakeholderRole*** *(StakeholderRoleTypeEnumeration)).*

Table 53 — *StakeholderRoleType*– Allowed Values

Value	Description
p*Planning*	Role is to plan transport services.
reservation	Role is to control running of transport services.
operation	Role is to operate transport services.
control	Role is to control running of transport services.
fareManagement	Role is to manageFares.
securityManagement	Role is to manage Security.
dataRegistrar	Role is to tobe data Regsitrar.
entityLegalOwnership	Role is to legally own the real-world physical entities corresponding to the software objects.
Other	Other roles.

7.4.4.3.3 ResponsibilityRole – Model Element

A particular role an ORGANISATION or an ORGANISATION PART is playing as regards certain data, for example data origination, data augmentation, data aggregation, data distribution, planning, operation, control, ownership etc.).

Table 54 — *ResponsibilityRole – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	RESPONSIBILITY ROLE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ResponsibilityRoleIdType*	1:1	Identifier of RESPONSIBILITY ROLE.
	Name	*MultilingualString*	0:1	Description of RESPONSIBILITY ROLE.

	Description	*MultilingualString*	0:1	Description of RESPONSIBILITY ROLE.
«FK»	***TypeOf-Responsibility-RoleRef***	*TypeOfResponsibility-RoleRef*	0:1	Reference to a TYPE OF RESPONSIBILITY ROLE.

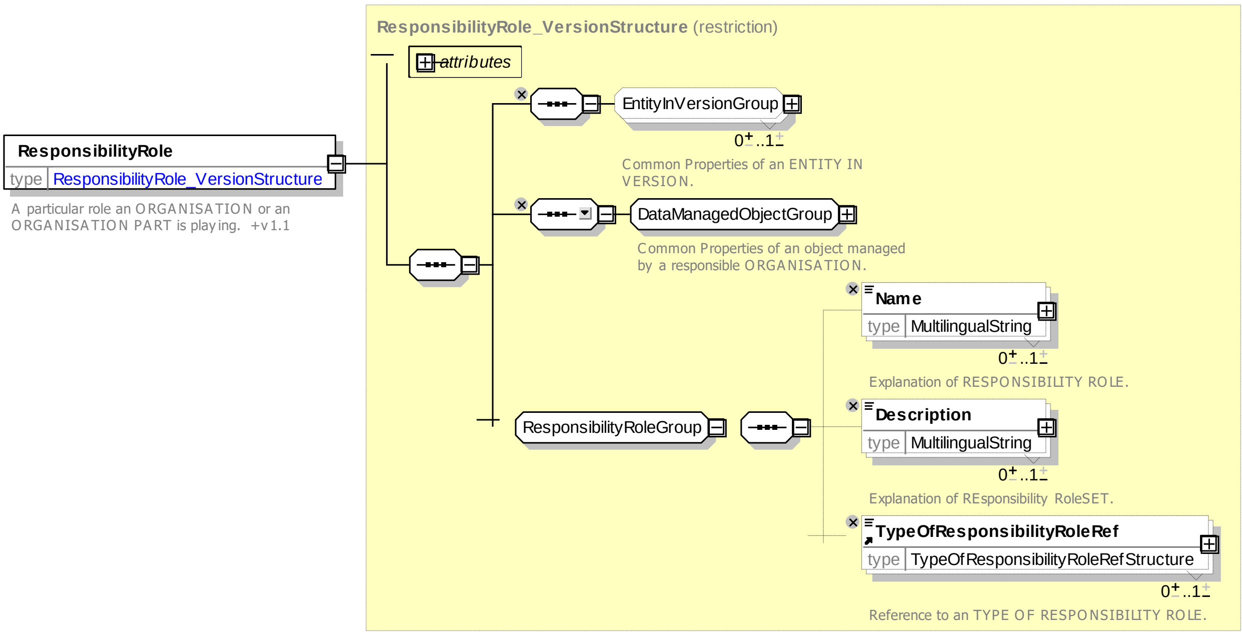


Figure 89 — *ResponsibilityRole – XSD*

7.4.4.3.4 TypeOfResponsibilityRole – Model Element

A classification of RESPONSIBILITY ROLEs, e.g. Data Ownership. Allows open values.

TYPE OF RESPONSIBILITY ROLE is also alternatively implemented by the ***DataRoleType*** and ***StakeholderRoleType*** enumerations which specify a fixed set of standard values.

Table 55 — *TypeOfResponsibilityRole – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*TypeOfEntity*</u>	::>	TYPE OF RESPONSIBILITY ROLE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfResponsibilityRoleIdType*	1:1	Identifier of TYPE OF RESPONSIBILITY ROLE.

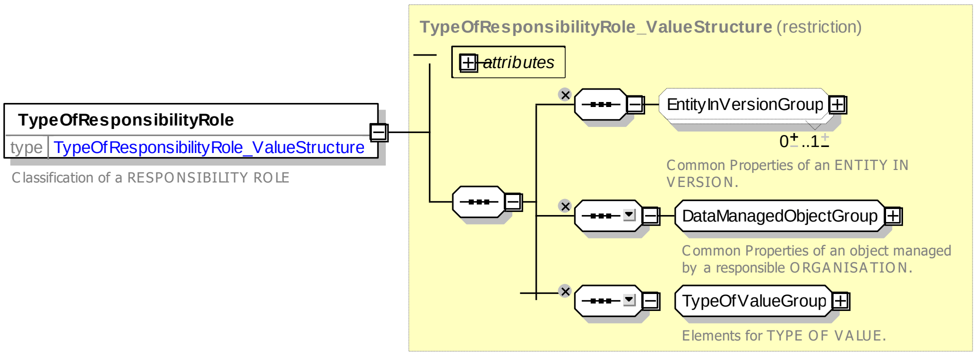


Figure 90 — *TypeOfResponsibilityRole – XSD*

7.4.4.4 *XML Examples of ResponsibilitySet*

7.4.4.4.1 Declaring a ResponsibilitySet – XML Example Fragments

The following example shows the hierarchical use of RESPONSIBILITY SETs. Three different RESPONSIBILITY SETs are declared*: 'napt:RS_82'* for London bus stop data, *'tbd:XtraAccessibilityData'* for accessibility data, and *'mydata:DB770' for* the producer of the delivered frames that contains the assembled data. The outer ***CompositeFrame*** sets *'mydata:DB770'* as the default to use if no specific source is provided. Each ***StopPlace*** is declared as coming from *'nptg:NaPTAN'.* The accessibility data for each stop however has been added from a different source: *'tbd:XtraAccessibilityData'.*

EXAMPLE

```
<CompositeFrame version="any" id="tbd::cf01" dataSourceRef=" mydata:DB770”>
    <ResourceFrame version="any" id="tbd::RS01" >
        <responsibilitySets>
            <ResponsibilitySet version="001" responsibilitySetRef="napt:RS_nptg" id="napt:RS_82">
            <Name>London collects London data</Name>
                <roles>
                    <ResponsibilityRoleAssignment version="001" id="napt:RS_82_01">
                        <Description>490 London - Managed centrally</Description>
                        <DataRoleType>collects</DataRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TL001"/>
                        <ResponsibleAreaRef version="any" ref="napt:82"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
            <ResponsibilitySet version="001" responsibilitySetRef="napt:RS_nptg" id="napt:RS_110">
                <Name>DfT collects Rail Data Nationally</Name>
                <roles>
                    <ResponsibilityRoleAssignment version="001" id="napt: RS_110_01">
                        <Description>910 UK Rail - Managed centrally</Description>
                        <DataRoleType>collects</DataRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TD002"/>
                        <ResponsibleAreaRef version="any" ref="nptg:110"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
            <ResponsibilitySet version="001" responsibilitySetRef="napt:RS_nptg" id="napt:RS_nptg">
                <Name>DfT collects and distrinbute station NPTG codes for areas</Name>
                <roles>
                    <ResponsibilityRoleAssignment version="001" id="napt:RS_nptg">
                        <Description>NPTG Coordination of area codes UK London</Description>
                        <DataRoleType>collects validates distributes</DataRoleType>
                        <StakeholderRoleType>Operation</StakeholderRoleType>
                        <ResponsibleOrganisationRef version="any" ref="tbd:Org_TD002"/>
                        <ResponsibleAreaRef version="any" ref="nptg:UK"/>
                    </ResponsibilityRoleAssignment>
                </roles>
            </ResponsibilitySet>
        </responsibilitySets>

::>….
```

```xml
<ServiceFrame version="any" id="tbd:SVC01">
<scheduledStopPoints>
    <ScheduledStopPoint version="any" id="napt:9100WIMBLDN"
                responsibilitySetRef="nptg:082" >
        <Name>Wimbledon Rail Station</Name>
        <VehicleModes>rail</VehicleModes>
    </ScheduledStopPoint>
    <ScheduledStopPoint version="any" id="napt:9400ZZLUWIM">
        <Name>Wimbledon Underground</Name>
        <VehicleModes>metro</VehicleModes>
    </ScheduledStopPoint>
    <ScheduledStopPoint version="any" id="napt:490000272P">
        <Name>Wimbledon Stop P</Name>
        <Description>Bus outside station</Description>
        <VehicleModes>bus</VehicleModes>
    </ScheduledStopPoint>
    <ScheduledStopPoint version="any" id="napt:490000272C">
        <Name>Wimbledon Stop C</Name>
        <Description>Bus </Description>
        <VehicleModes>bus</VehicleModes>
    </ScheduledStopPoint>
</scheduledStopPoints>
```

7.4.5 AlternativeText

It is sometimes necessary to provide several variants of a single text, in particular if the information is required in several national languages. The ***AlternativeText*** element is a generic way of providing such variants for any text attribute of a ***DataManagedObject***. It can bee seen as a complement to the ***AlternativeName*** mechanism (described later), and can be used to provide an alias for any description or text attribute.

7.4.5.1 AlternativeText – Physical Model

Since the provision of ***AlternativeText*** is primarily of concern for implementation tools, only a physical model is provided here.
The ***AlternativeText*** is part of a ***DataManagedObject*** and references the name of the attribute in the NeTEx Metamodel for which it is providing an alternative. It contains the alternative text as an attribute of type *MultilingualString* which indicates the language it is in. In addition the text may have a 'Use for' language attribute to indicate a second language for which it may be used as an acceptable presentation if there is no native language alternative; normally this will be the same as the language of the string, but it might be different.

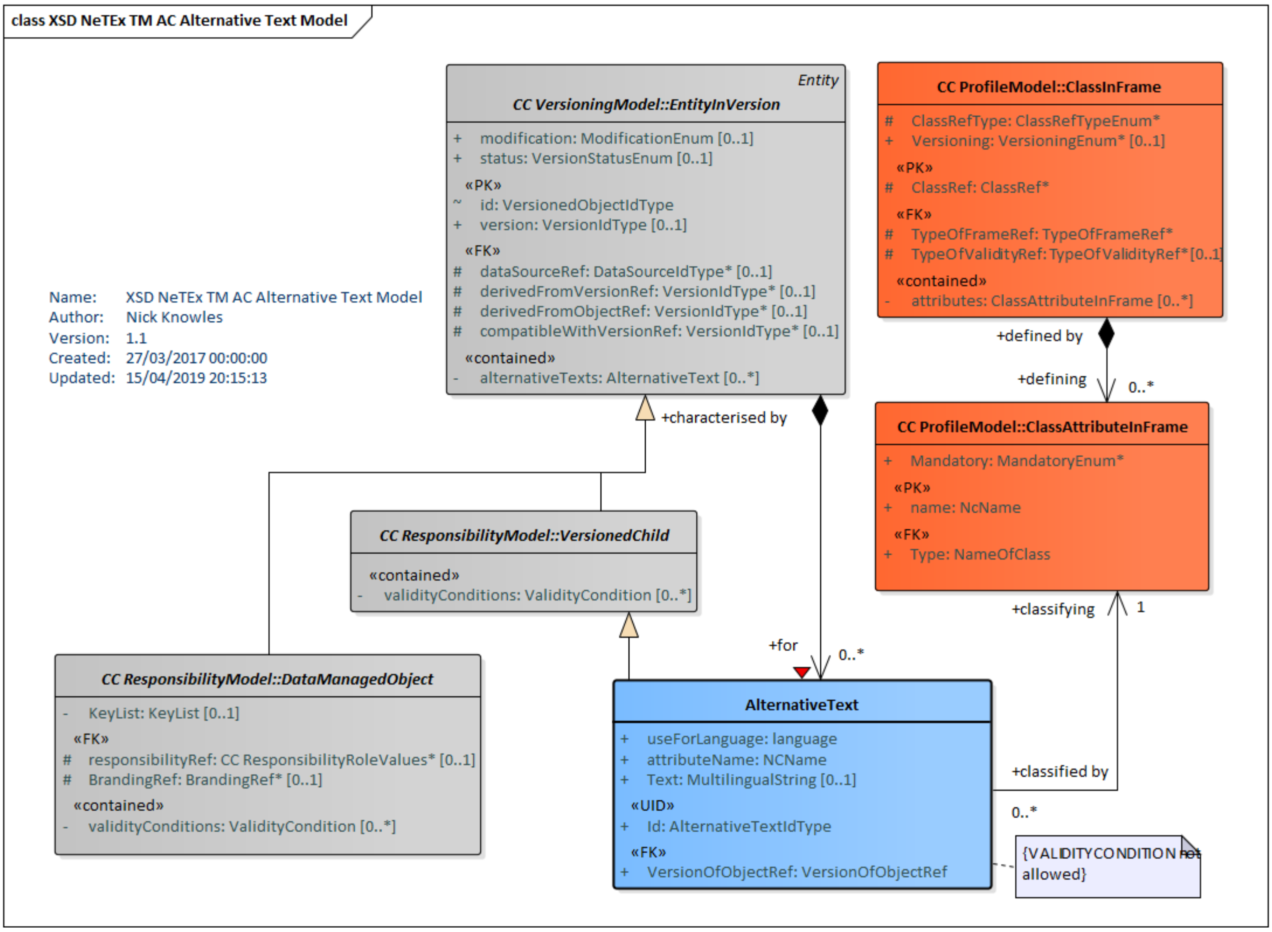


Figure 91 — Alternative Text – Physical Model (UML)

7.4.5.2 AlternativeText – Attributes and XSD

Alternative text for an ENTITY, e.g. in a different language.

Table 56 — AlternativeText – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	**AlternativeText** inherits from VERSIONED CHILD.
«atr»	***attributeName***	xsd:NCName	0:1	Name of text attribute for which this is the alternative text. Must be an existing attribute name.
«atr»	***useForLanguage***	xsd:language	0:1	Name of language for which this is to be used.
«atr»	***order***	xsd:integer	0:1	Order of ALTERNATIVE TEXT.
«FK»	***DataManaged-ObjectRef***	*DataManagedObjectRef*	0:1	Reference to the DATA MANAGED OBJECT containing the attribute.
	Text	*MultilingualString*	1:1	Variant of the text in the specified language.

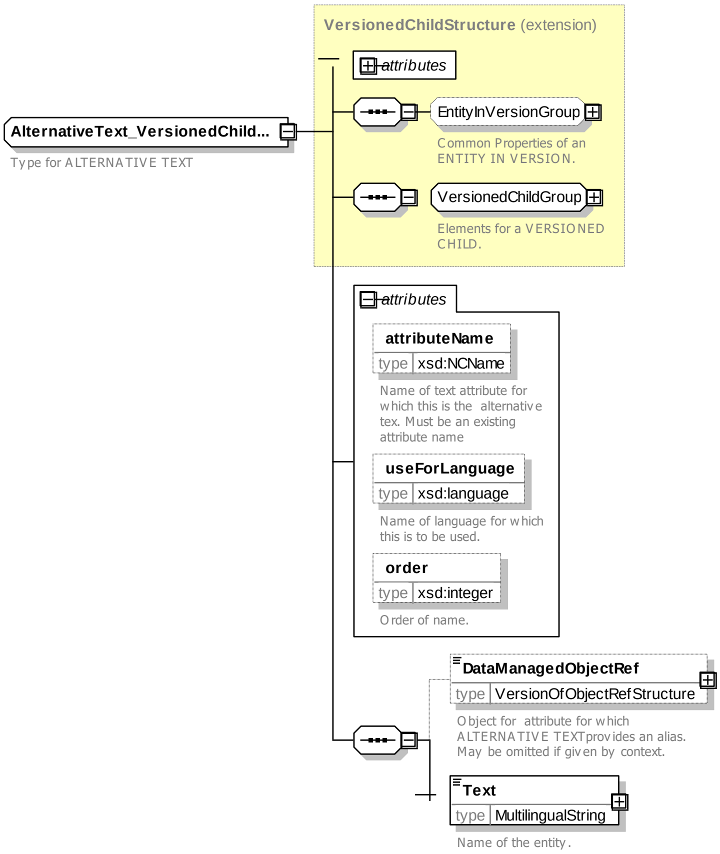


Figure 92 — *AlternativeText – XSD*

7.4.5.2.1 AlternativeText – XML Fragment Example

The following XML fragment shows uses of ***AlternativeText*** within a ***DayType*** to provide the **name** and **shortname** information in several languages.

EXAMPLE

```
<DayType version="any" id="xj:DT_01">
    <alternativeTexts>
        <AlternativeText version="any" id="xj:DT_01" attributeName="Name"
useForLanguage="fr">
            <Text lang="fr">Tous les jours sauf pour les jours fériés</Text>
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="Name"
useForLanguage="de">
            <Text lang="de">Jeden Tag außer an Feiertagen</Text>
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="Name"
useForLanguage="it">
            <Text lang="it">Tutti i giorni eccetto i giorni festivi</Text>
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="Name"
useForLanguage="se">
            <Text lang="se">Varje dag utom helgdagar</Text>
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="ShortName"
useForLanguage="fr">
            <Text lang="fr">Tous les jours</Text>
```

```
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="ShortName"
useForLanguage="de">
            <Text lang="de">Jeden Tag</Text>
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="ShortName"
useForLanguage="it">
            <Text lang="it">Ogni giorno</Text>
        </AlternativeText>
        <AlternativeText version="any" id="xj:DT_01" attributeName="ShortName"
useForLanguage="se">
            <Text lang="se">Varje dag</Text>
        </AlternativeText>
    </alternativeTexts>

    <!--- Values -->
    <Name lang="en">Everyday except for public holidays</Name>
    <ShortName lang="en">Every day</ShortName>
    <EarliestTime>00:00:00Z</EarliestTime>
    <DayLength>PT24H</DayLength>
    <properties>
        <PropertyOfDay>
            <DaysOfWeek>Everyday</DaysOfWeek>
            <HolidayTypes>NotHoliday</HolidayTypes>
        </PropertyOfDay>
    </properties>
</DayType>
```

7.4.6 Alternative Name

7.4.6.1 Alternative Name – Conceptual MODEL

The ALTERNATIVE NAME Model defines reusable name alias elements for significant entities that have names, for example in another language or if a station or product has several names. It allows a structure qualified to be used, for example, to distinguish between two places with the same name in different countries. It complements the ALTERNATIVE TEXT entity which is used to provide translations for individual text attributes of elements.

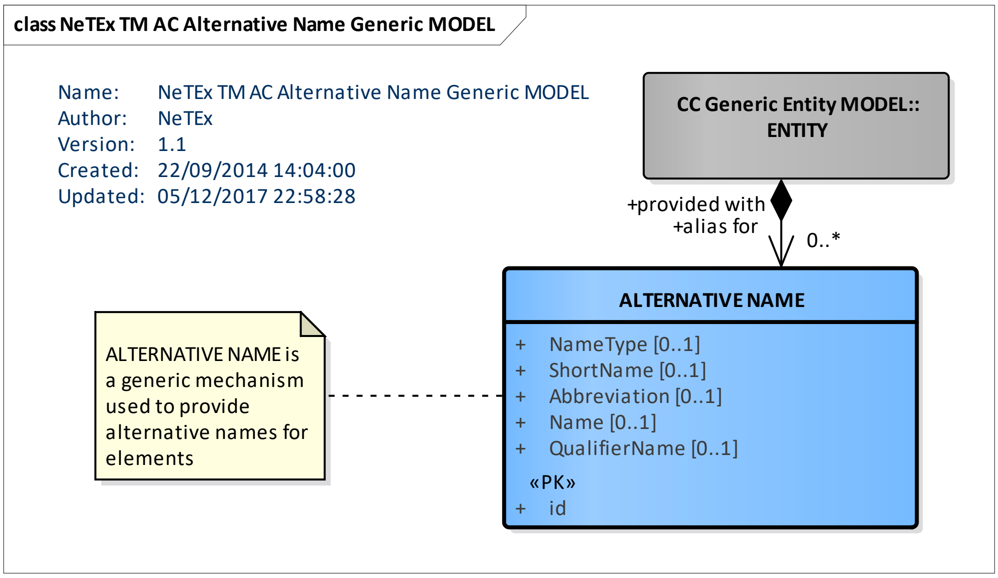


Figure 93 — Alternative Name – Conceptual MODEL (UML)

7.4.6.2 Alternative Name – Physical Model

The following figure shows the Physical model for ALTERNATIVE NAMEs. The content of a ALTERNATIVE NAME is a MULTILINGUALSTRING, thus enabling the text of the ALTERNATIVE NAME to be made available in several languages

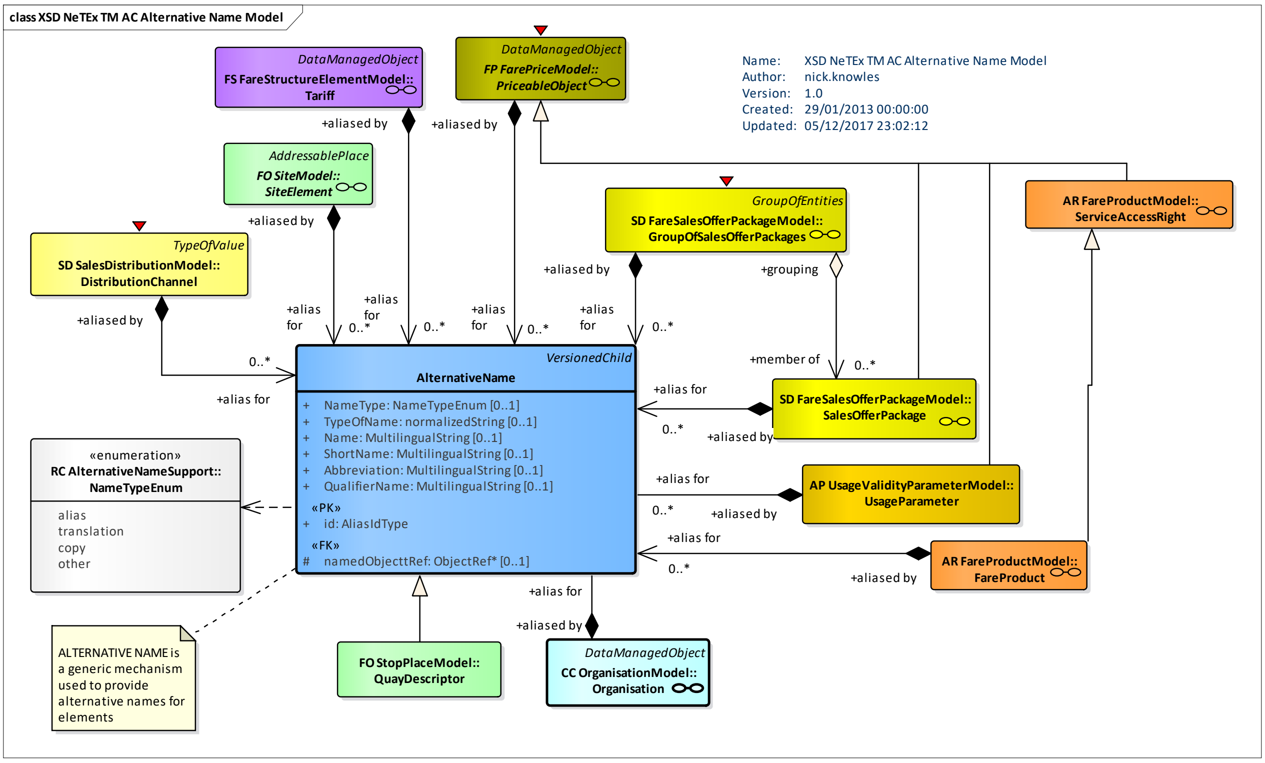


Figure 94 — Alternative Name – Physical Model (UML)

7.4.6.3 Alternative Name – Attributes and XSD

7.4.6.3.1 AlternativeName – Model Element

Alternative name for the entity.

Table 57 – *AlternativeName* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	ALTERNATIVE NAME inherits from VERSIONED CHILD
«PK»	***id***	*AliasIdType*	1:1	Identifier of ALTERNATIVE NAME.
«FK»	***NamedObjectRef***	*(VersionOfObjectRef)*	0:1	Parent PLACE for which this ALTERNATIVE NAME applies.
	Lang	*Language*	0:1	Language of name
«enum»	***NameType***	*NameTypeEnum*	0:1	Type of ALTERNATIVE NAME. See allowed values below
	TypeOfName	*NormalizedString*	0:1	Text name of attribute of which this is of of ALTERNATIVE NAME.

	Name	*MultilingualString*	1:1	Text for ALTERNATIVE NAME.
	ShortName	*MultilingualString*	0:1	Short version of ALTERNATIVE NAME.
	Abbreviation	*MultilingualString*	0:1	Abbreviation associated with ALTERNATIVE NAME.
	QualifierName	*MultilingualString*	0:1	Name of Qualifier to be used with ALTERNATIVE NAME

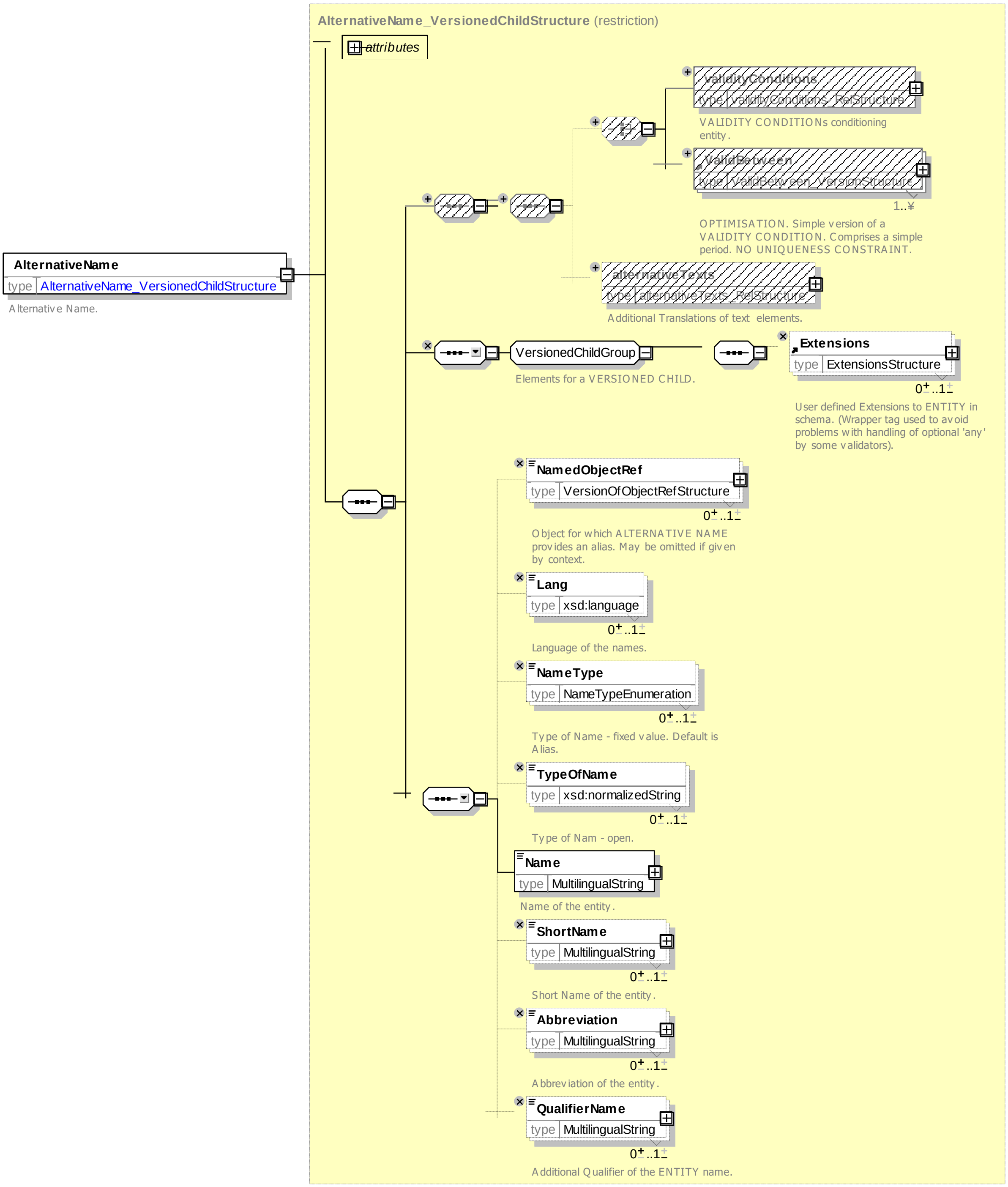


Figure 95 – *AlternativeName* – XSD

7.4.6.3.1.1 NameType – Allowed Values

Allowed values for ***NameType*** (*NameTypeEnumeration*)

Table 58 — *NameType* – Allowed Values

Value	Description
alias	Alternative name is an aliass.
translation	Translation in another languageurce.
copy	Unspecified
label	Name is a shorter version.
other	Other.

7.4.7 Generic Organisation

The Generic ORGANISATION Model defines abstract ORGANISATION elements that can be used wherever there is a need to describe an organisation. It is extended in the Reusable Components section with AUTHORITY, OPERATOR and other concrete organisation definitions specifically relevant for the transport domain.

7.4.7.1 Generic ORGANISATION – Conceptual MODEL

The entity ORGANISATION represents an organisation that is involved in the planning, collecting or provision of PT information. For example, a company providing a public transport information service, an authority, an operator, or a company providing an information collection service.

Many organisations break down their operations in different organisation parts. This may be important not only from the operational point of view, but also for data administration, as such units may have different responsibilities. Some common data will be shared between them whereas some other will be managed by a specific part. The RESPONSIBILITY ROLE ASSIGNMENT can be used to describe these responsibilities.

An ORGANISATION can consist of several DEPARTMENTs or ORGANISATIONAL UNITs. Those departments or units are modelled in the ORGANISATION PART.

A DEPARTMENT can consist of one or more ORGANISATIONAL UNITs, which are in charge of operational functions. In a PTO context, a DEPARTMENT could comprise all ORGANISATIONAL UNITs responsible for the lines served by the same transport mode, or using the same type of operation (e.g. regular service, night service).

In some cases, the organisational aspect of responsibilities for planning and operation need not necessarily to be present in a company data model. Therefore, the relationships to (and the existence of) these organisational entities are optional.

The ADMINISTRATIVE ZONE represents a set of PT objects related to a district, a region, a city, a municipality, a traffic system, a set of lines or other subdivision for a specific purpose.

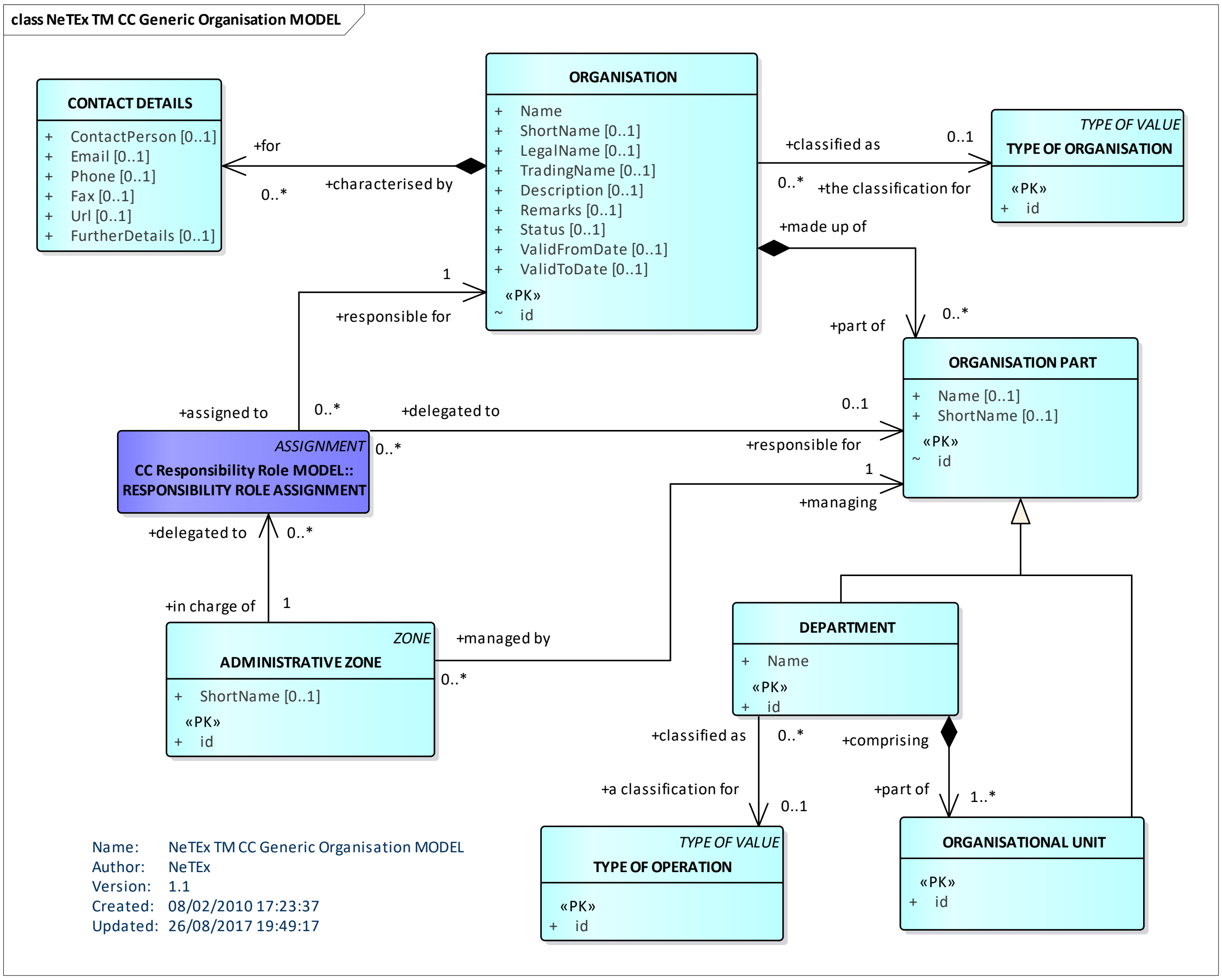


Figure 96 — Generic Organisation — Conceptual MODEL (UML)

7.4.7.2 Generic Organisation – Physical Model

The Physical model adds detailed attributes for the various ORGANISATION related elements. In addition an ADMINISTRATIVE ZONE may be associated with a CODESPACE and class so as to indicate the range of identifiers reserved for a region and organisation or similar.

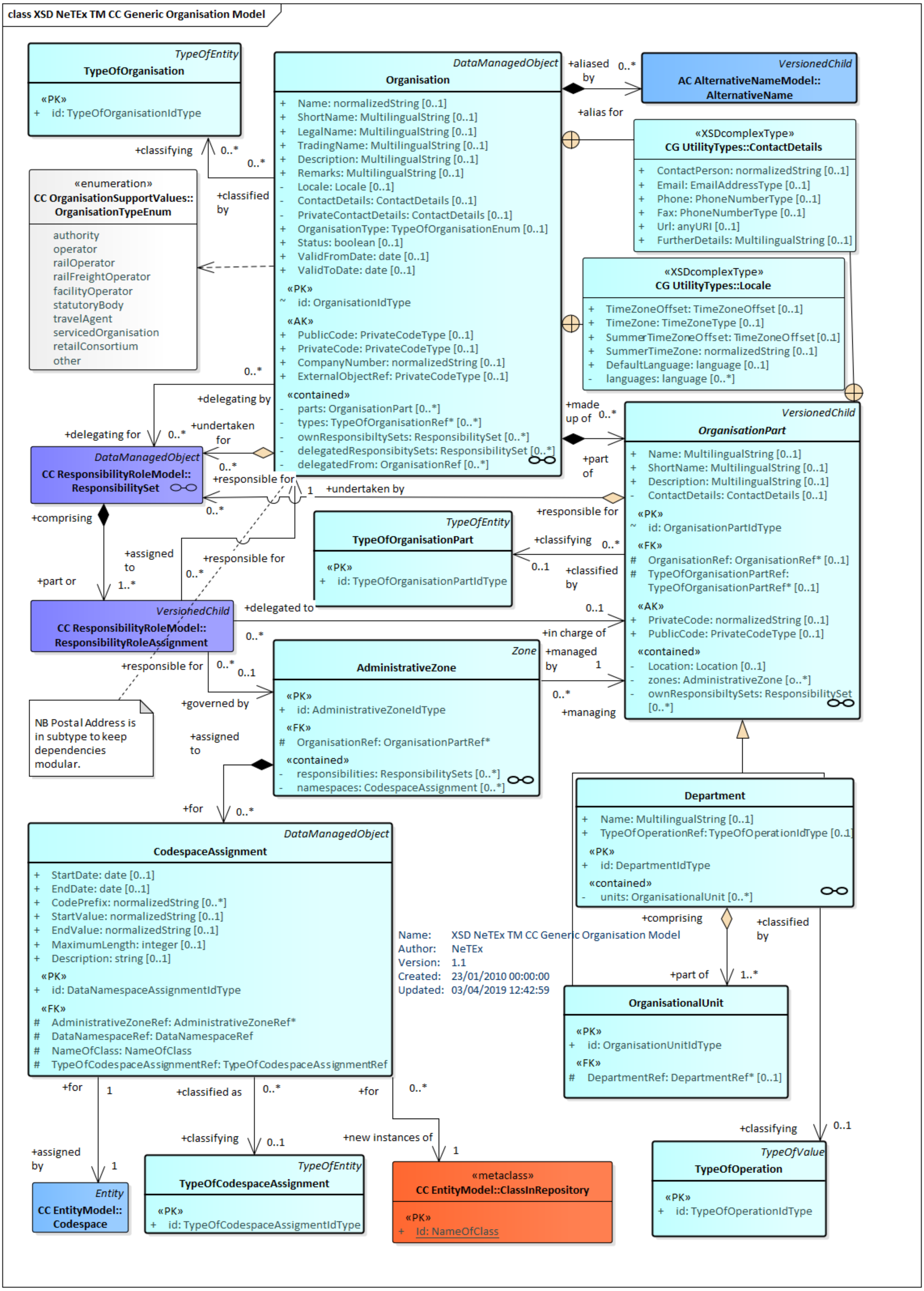


Figure 97 — Generic Organisation — Physical Model (UML)

The following figure summarises the descendants of ORGANISATION found in NeTEx.

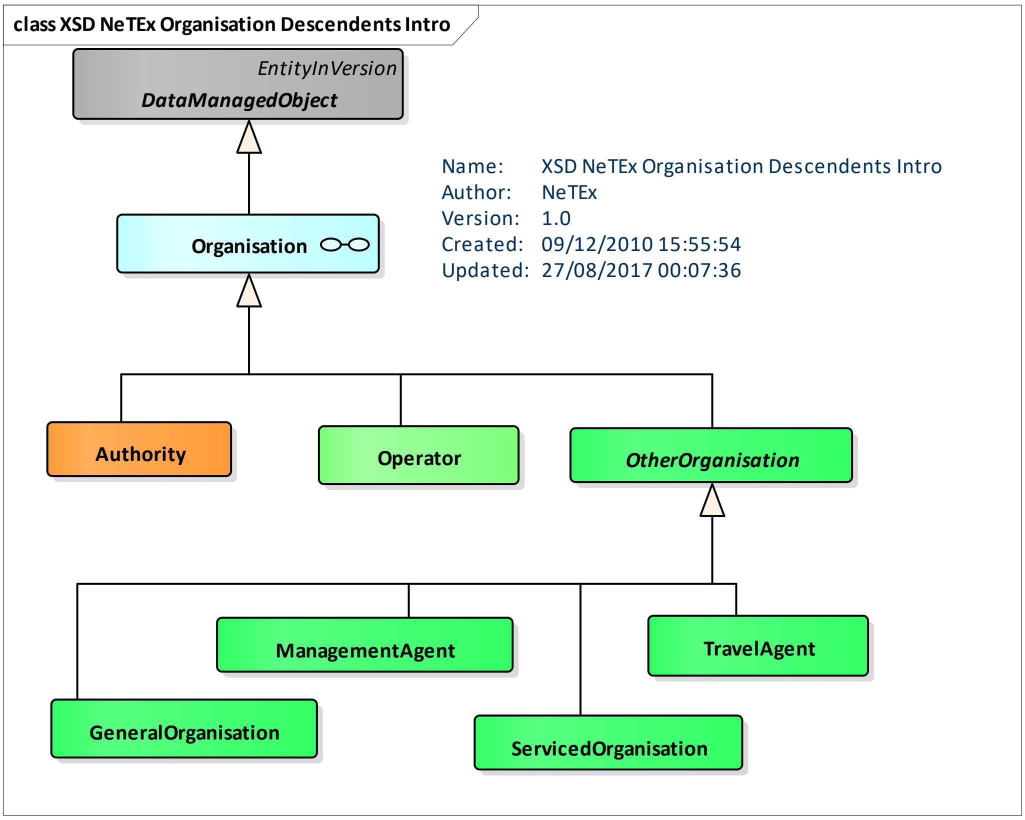


Figure 98 — Organisation Descendants - Physical Model (UML)

7.4.7.3 Organisation – Example of Administrative Zones

The following figure shows how the UK is divided into a hierarchy of administrative zones to manage stop data. There are eleven main regions, divided into areas by authoritry, each of whom has responsibility for stop data of certain sorts within their region. Real-time data may be managed at an even more local level.

Figure 99 — Example of administrative zones

7.4.7.4 Organisation – Attributes and XSD

7.4.7.4.1 Organisation – Model Element

Any legally incorporated body associated with any aspect of the transport system.

Table 59 — *Organisation – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	ORGANISATION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*OrganisationIdType*	1:1	Identifier of ORGANISATION.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Alternative public identifier of ORGANISATION.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative identifier of ORGANISATION.
«AK»	***CompanyNumber***	*xsd:normalizedString*	0:1	Registration number of ORGANISATION a legally incorporated body.
«AK»	***ExternalOperatorRef***	*ExternalObjectRef*	0:1	An alternative code that uniquely identifies the ORGANISATION, specifically for use in AVMS systems. For VDV compatibility.
	Name	*xsd:normalizedString*	0:1	Name of ORGANISATION.
	ShortName	*MultilingualString*	0:1	Short name of ORGANISATION.
	LegalName	*MultilingualString*	0:1	Legal name of ORGANISATION.
	TradingName	*MultilingualString*	0:1	Trading name of ORGANISATION.
«cntd»	***alternativeNames***	*alternativeName*	0:*	Alternative names for ORGANISATION. +v1.1
	Description	*MultilingualString*	0:1	Further description of ORGANISATION.
	Remarks	*MultilingualString*	0:1	Further remarks about ORGANISATION,
«cntd»	***Locale***	*Locale*	0:1	Locale of ORGANISATION specifying time zone, language, etc.
«cntd»	***ContactDetails***	*ContactDetails*	0:1	Contact details for ORGANISATION for public use.
«cntd»	***PrivateContact-Details***	*ContactDetails*	0:1	Contact details for ORGANISATION for privileged use, for example for immediate access to control centres.
«enum»	***OrganisationType***	*TypeOfOrganisationEnum*	0:1	Type of ORGANISATION. See allowed values below.
«cntd»	***typesOf-Organisation***	*TypeOfOrganisationRef*	0:*	References to a TYPE of ORGANISATION that classifies it.

	Status	*xsd:boolean*	0:1	Whether the organisation is active. The default is '*true*'.
«cntd»	***ValidityPeriod***	*SimpleValidityCondition*	0:1	VALIDITY Condition for organisation (deprecated: ValidityCondition is now inhérited from EntityInVersion)
«cntd»	***parts***	*(OrganisationPart)* *OrganisationUnit* | *Department* | *OrganisationUnitRef* | *DepartmenRef*|	0:*	Any component parts of an ORGANISATION, that is OPERATIONAL UNITs or DEPARTMENTs.
«cntd»	***own-Responsibility-Sets***	*ResponsibilitySet* | *ResponsibilitySetRef*	0:*	Own RESPONSIBILITY SETs.
«cntd»	***delegated-Responsibility-Sets***	*ResponsibilitySet* | *ResponsibilitySetRef*	0:*	Delegated RESPONSIBILITY SETs.
«FK»	***delegatedFrom***	*(OrganisationRef)*	0:*	References to Other Organisations that delegate to this ORGANISATION. (TAP TSI B1.)

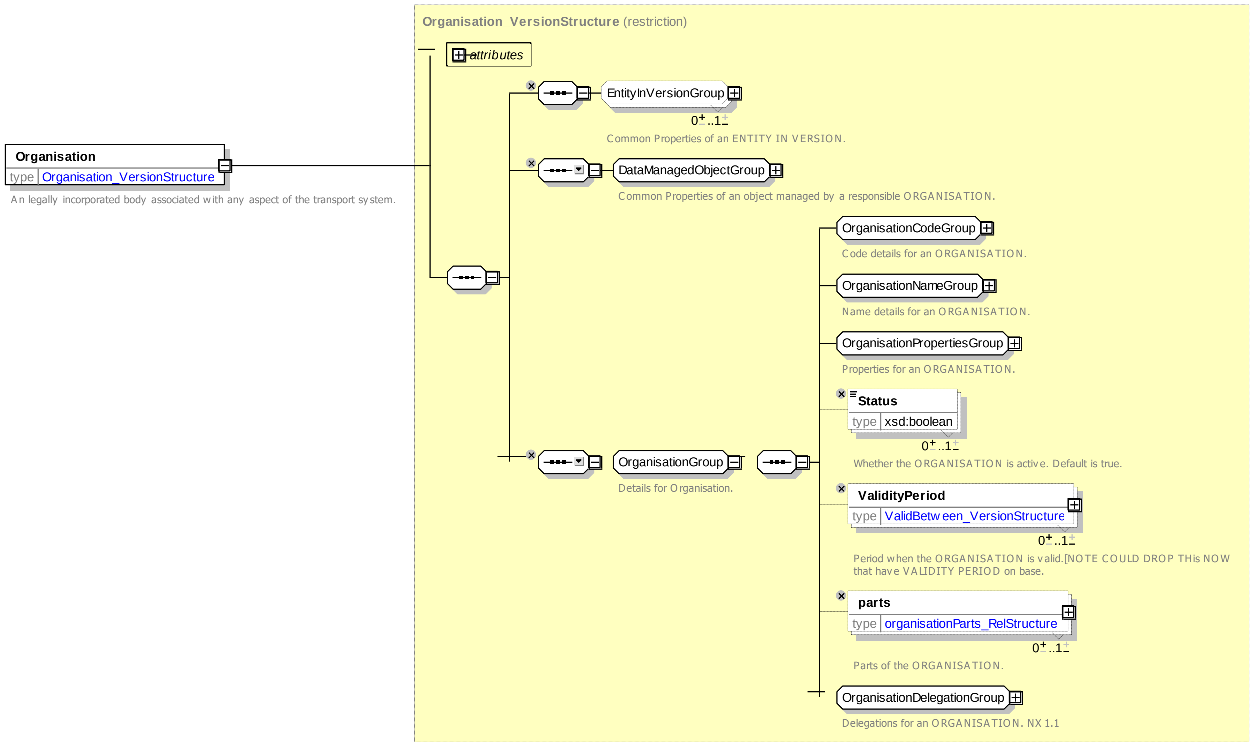


Figure 100 — *Organisation – XSD*

7.4.7.4.1.1 OrganisationCodeGroup – XML Group

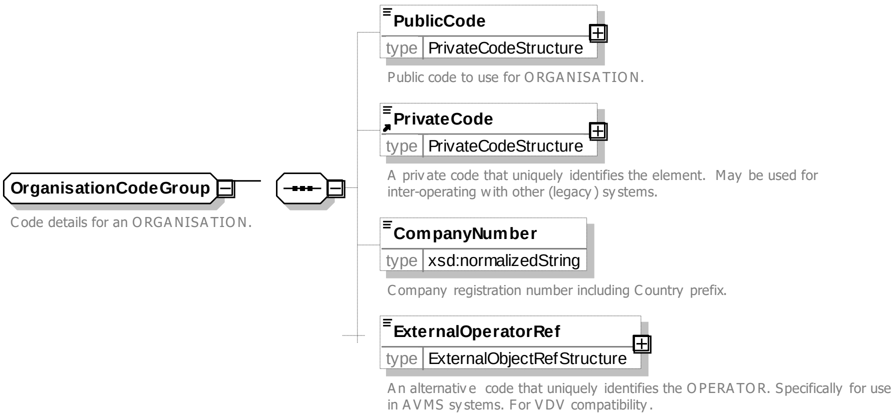


Figure 101 — *OrganisationCodeGroup – XSD*

7.4.7.4.1.2 OrganisationNameGroup – XML Group

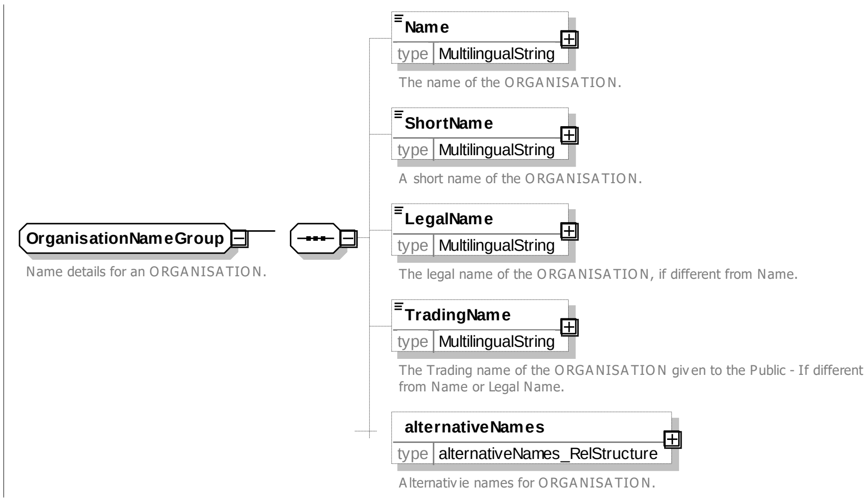


Figure 102 — *OrganisationNameGroup – XSD*

7.4.7.4.1.3 OrganisationPropertiesGroup – XML Group

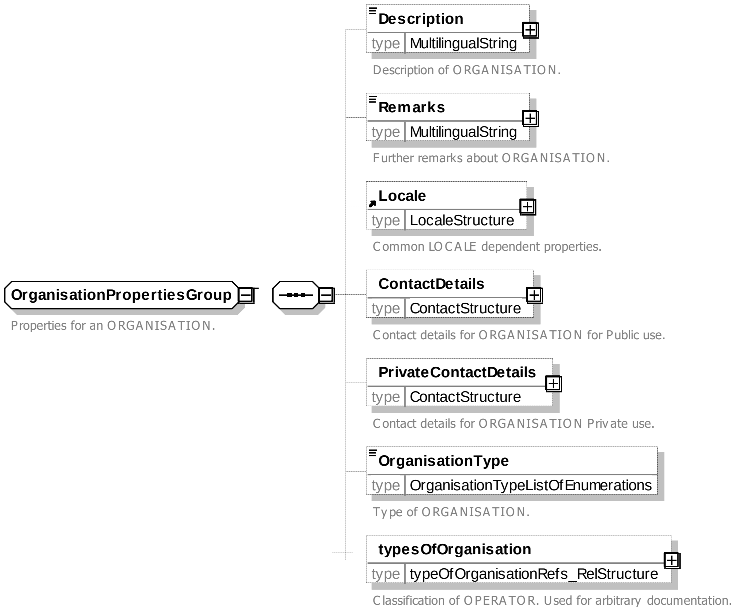


Figure 103 — *OrganisationPropertiesGroup – XSD*

7.4.7.4.1.4 OrganisationDelegationGroup – XML Group

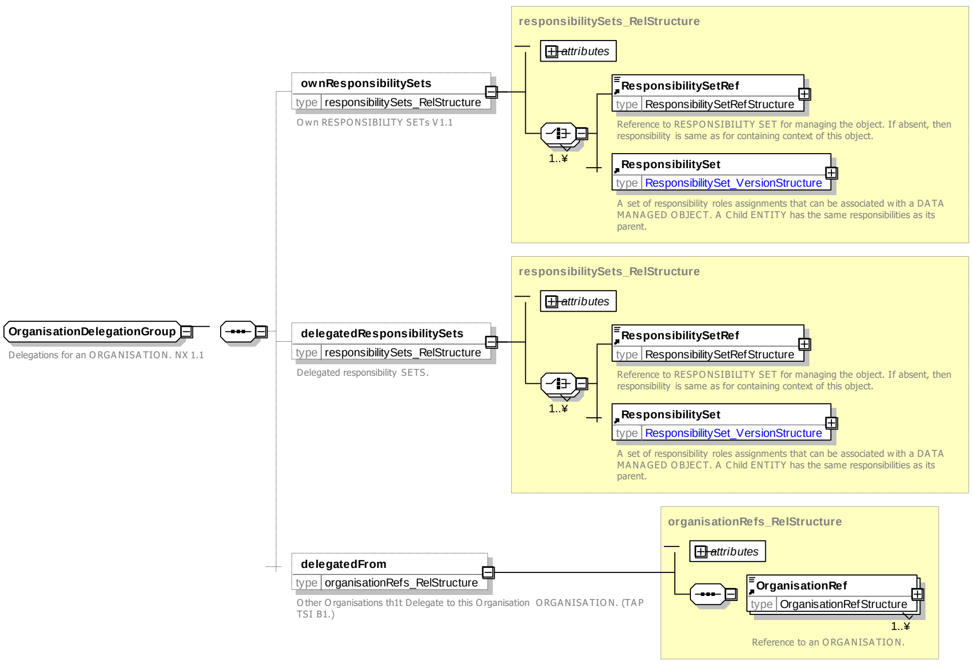


Figure 104 — *OrganisationDelegationGroup – XSD*

7.4.7.4.1.5 OrganisationType – Allowed Values

Allowed values for a TYPE OF ORGANISATION (*OrganisationTypeEnumeration)*.

Table 60 — *OrganisationType* – Allowed Values

Value	Description
authority	Transport Authority or Agency.

operator	Public Transport OPERATOR.
railOperator	Rail OPERATOR.
railFreightOperator	Rail freight OPERATOR.
facilityOperator	Operator of a facility such as a station.
statutoryBody	Statutory body or government department.
travelAgent	Travel Agent.
servicedOrganisation	Business or organisation served by public transport.
retailConsortium	Rretail consortium.
other	Other type of ORGANISATION.

7.4.7.4.2 OrganisationPart – Model Element

A part of an ORGANISATION to which specific responsibilities upon the data and data management may be assigned.

Table 61 — *OrganisationPart – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	ORGANISATION PART inherits from DATA MANAGED OBJECT.
«PK»	***id***	*OrganisationPartIdType*	1:1	Identifier of an ORGANISATION PART.
	Name	*MultilingualString*	0:1	NAME of the ORGANISATION PART.
	ShortName	*MultilingualString*	0:1	SHORT NAME of the ORGANISATION PART.
	Description	*MultilingualString*	0:1	Description of the ORGANISATION PART.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Public code of the ORGANISATION PART.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Private code of the ORGANISATION PART. May be used for inter-operating with other (legacy) systems.
«cntd»	***ContactDetails***	<u>*ContactDetails*</u>	0:1	Contact details for ORGANISATION PART for public use.
«cntd»	***Location***	<u>*Location*</u>	0:1	Location of the ORGANISATION PART.
«FK»	***Organisation-Ref***	*(OrganisationRef)*	0:1	Reference to an ORGANISATION to whom the ORGANISATION PART belongs.
«FK»	***TypeOf-Organisation-PartRef***	*TypeOfOrganisationPart Ref*	0:1	Reference to a TYPE of ORGANISATION PART that classifies this part.
«cntd»	***administrative-Zones***	<u>*AdministrativeZone*</u> | <u>*AdministrativeZoneRef*</u>	0:*	ADMINISTRATIVE ZONEs administered by ORGANISATION PART.

«cntd»	***own-Responsibility Sets***	*<u>ResponsibilitySet</u>* | *ResponsibilitySetRef*	0:*	Own RESPONSIBILITY SETs.
«cntd»	***delegatedResponsibilitySets***	*<u>ResponsibilitySet</u>* | *ResponsibilitySetRef*	0:*	Delegated RESPONSIBILITY SETs.
«FK»	***delegatedFrom***	*organisationRef*	0:1	Other Organisations that delegates to this Organisation Part ORGANISATION. (TAP TSI B1.)

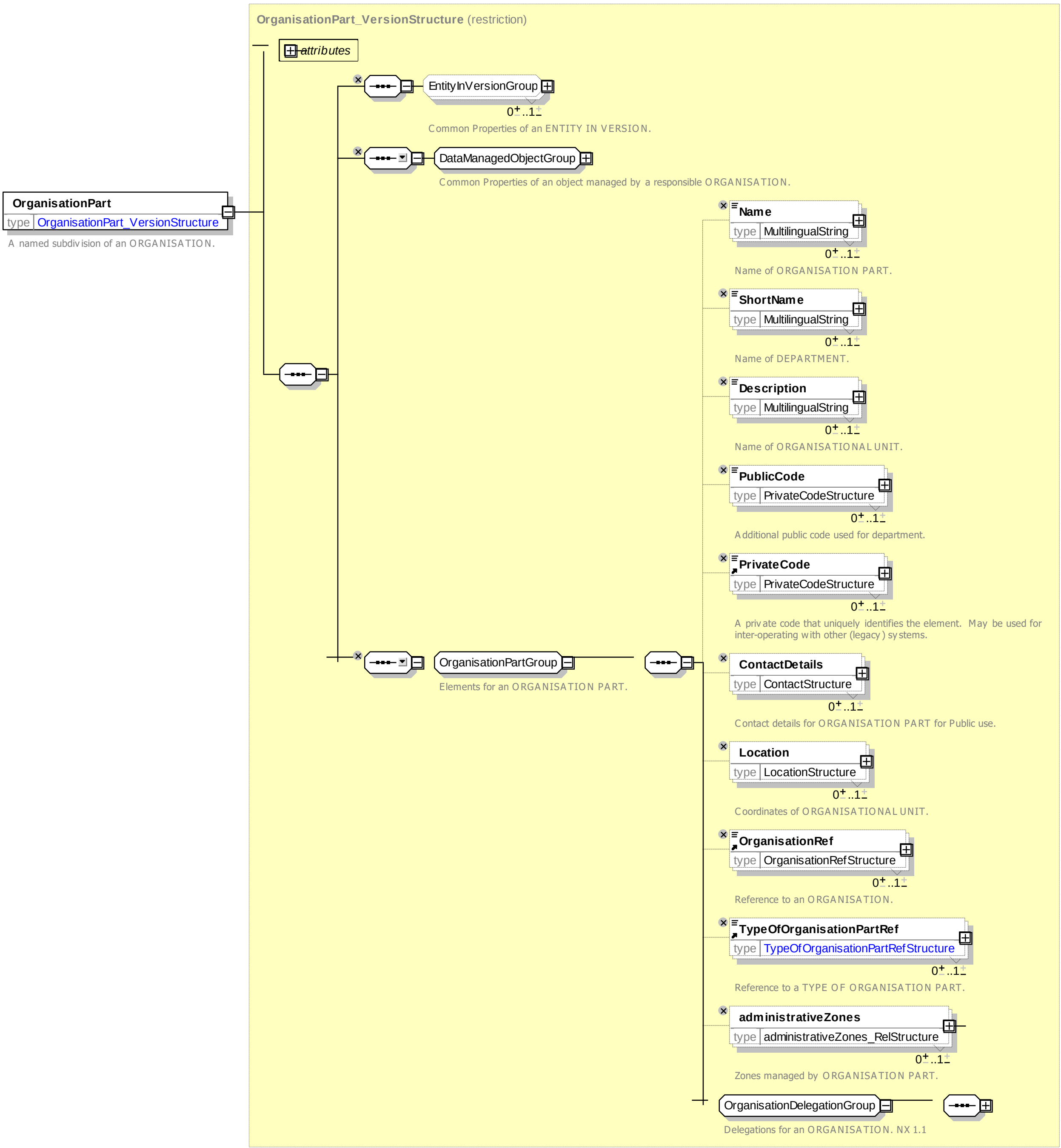


Figure 105 — *OrganisationPart – XSD*

7.4.7.4.3 OrganisationalUnit – Model Element

An ORGANISATION PART to which a set of responsibilities in a public transport company for planning and control is assigned.

Table 62 — *OrganisationalUnit – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*OrganisationPart*	::>	ORGANISATIONAL UNIT inherits from ORGANISATION PART.
«PK»	***id***	*OrganisationUnitIdType*	1:1	Identifier of an ORGANISATIONAL UNIT.
«FK»	***DepartmentRef***	*DepartmentRef*	0:1	Reference to an OPERATING DEPARTMENT to which this ORGANISATIONAL UNIT belongs.

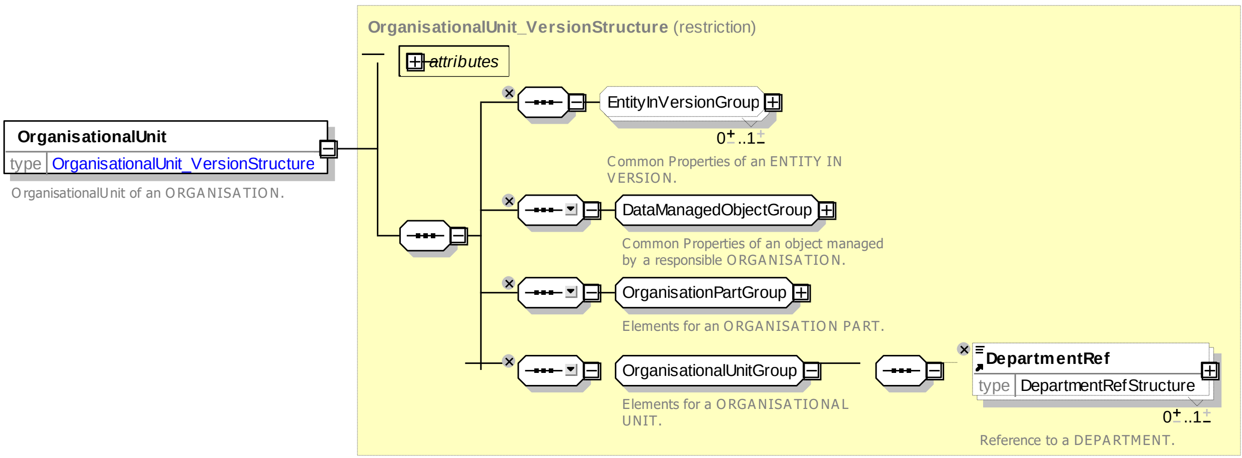


Figure 106 — *OrganisationalUnit – XSD*

7.4.7.4.4 Department – Model Element

An ORGANISATION PART specific to a purpose and/or organisational structure.

Table 63 — *Department – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*OrganisationPart*	::>	DEPARTMENT inherits from ORGANISATION PART.
«PK»	***id***	*DepartmentIdType*	1:1	Identifier of OPERATING DEPARTMENT.
«FK»	***TypeOfOperationRef***	*TypeOfOperationRef*	1:1	TYPE OF OPERATION of DEPARTMENT.
«cntd»	***units***	*OrganisationalUnitRef*	0:*	List of ORGANISATIONAL UNITS belonging to DEPARTMENT.

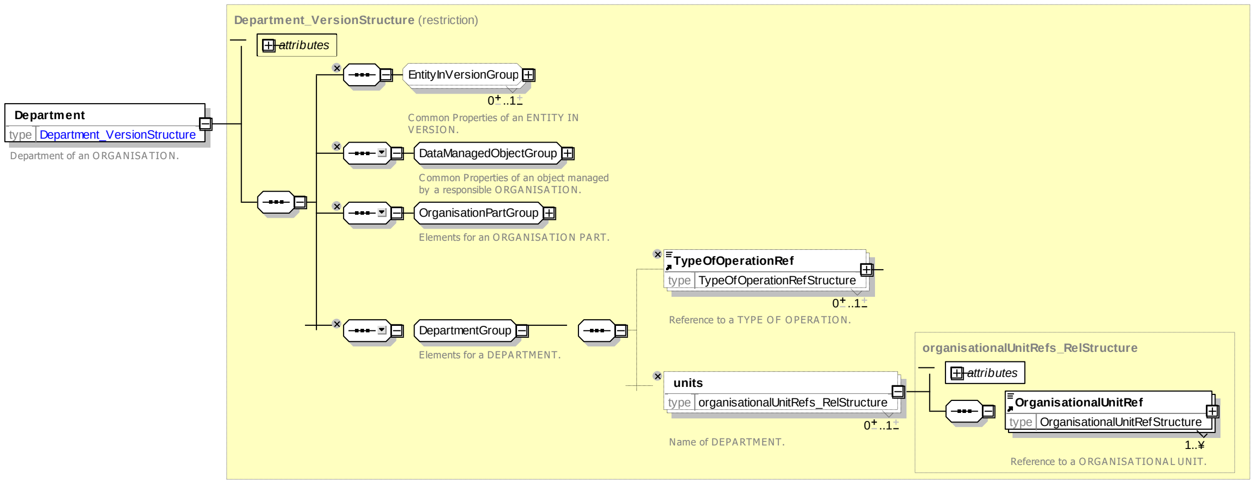


Figure 107 — *Department – XSD*

7.4.7.4.5 AdministrativeZone – Model Element

A certain subdivision such as a district, a region, a city, a municipality, or other area for which an ORGANISATION has a RESPONSIBILITY ROLE.

Table 64 — *AdministrativeZone – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Zone*	::>	ADMINISTRATIVE ZONE inherits from ZONE.
«PK»	***id***	*AdministrativeZoneIdType*	1:1	Identifier of an ADMINISTRATIVE ZONE.
«AK»	***PublicCode***	*PrivateCodeStructure*	1:1	External code for an ADMINISTRATIVE ZONE.
«FK»	***OrganisationRef***	*(OrganisationRef)*	1:1	Reference to an ORGANISATION who manages the ADMINISTRATIVE ZONE.
«cntd»	***responsibilities***	*Management-Responsibilities*	0:*	RESPONSIBILITIES of the ORGANISATION for the ADMINISTRATIVE ZONE.
«cntd»	***codespace-Assignments***	*CodespaceAssignment*	0:*	CODESPACES associated with the ADMINISTRATIVE ZONE.
«cntd»	***subzones***	*AdministrativeZone*	0:*	Subzones of ADMINISTRATIVE ZONE. +v1.1

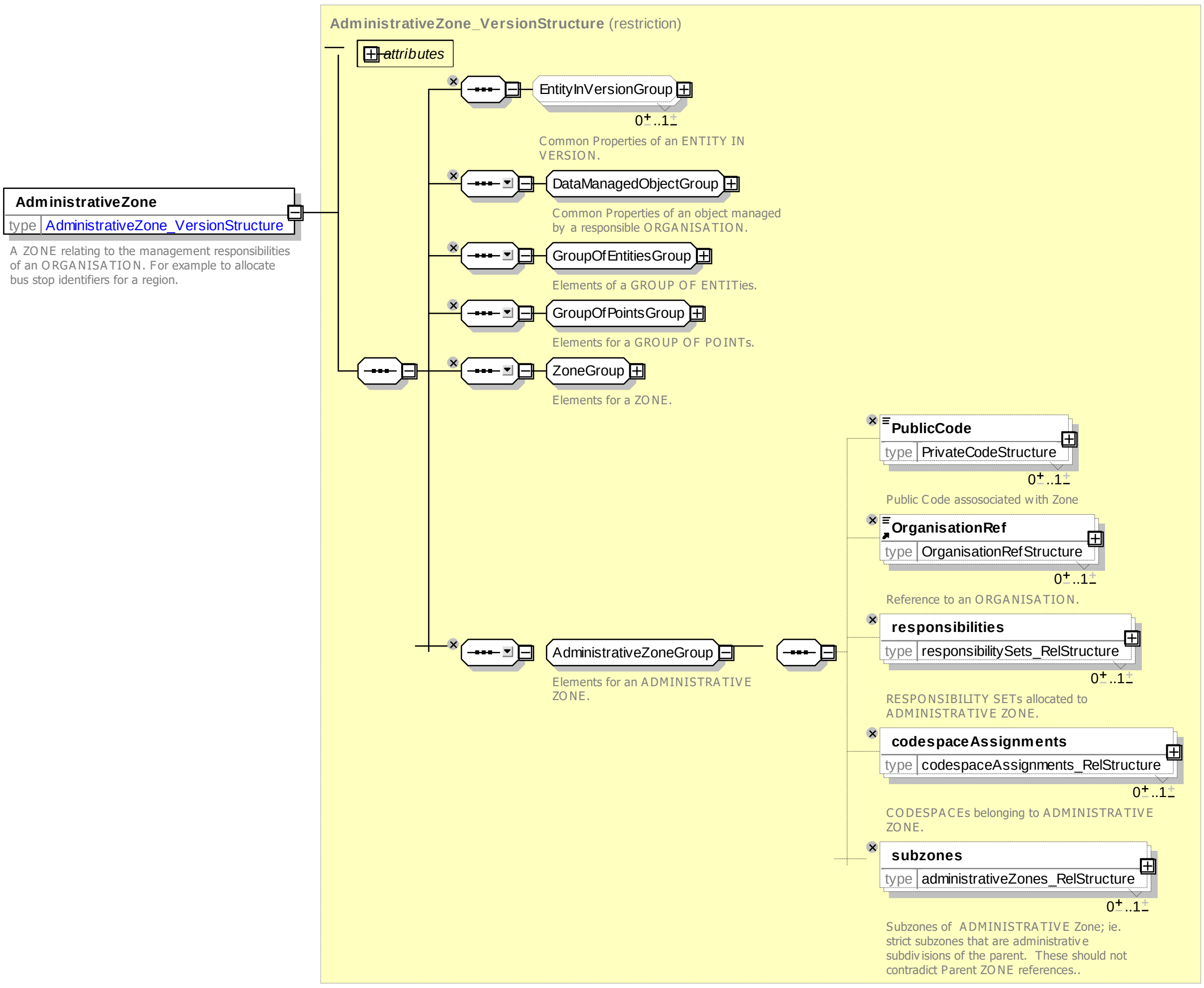


Figure 108 — *AdministrativeZone – XSD*

7.4.7.4.6 CodeSpaceAssignment – Model Element

CODESPACE ASSIGNMENT indicates which ADMINISTRATIVE ZONE is responsible for issuing identifiers that are unique within a given CODESPACE.

This can be used to support the use of distributed processes for allocating unique identifier codes under various different organisational models.

Table 65 — *NamespaceAssignment – Element*

Classification	Name		Type	Cardinality	Description
::>	::>		*VersionedChild*	::>	CODESPACE ASSIGNMENT inherits from VERSIONED CHILD.
«PK»	***id***		*CodespaceAssignment-IdType*	1:1	Identifier of CODESPACE ASSIGNMENT.
«cntd»	a	***Codespace-Refe***	*CodespaceRef*	1:1	Reference to a CODESPACE for which the allocation of codes is assigned.
«cntd»	b	***Codespace***	*Codespace*	1:1	CODESPACE for which the allocation of codes is assigned.

«FK»	***Administrative-ZoneRef***	*AdministrativeZoneRef*	1:1	Reference to the ADMINISTRATIVE ZONE to whom this CODESPACE ASSIGNMENT assigns the allocation of codes.
	StartDate	*xsd:date*	0:1	Start date for CODESPACE ASSIGNMENT.
	EndDate	*xsd:date*	0:1	End date for CODESPACE ASSIGNMENT.
«FK»	***NameOfClass***	*NameOfClass*	0:1	Name of type of element whose names are assigned.
	CodePrefix	*xsd:normalizedString*	0:*	Allowed values for code prefixes managed by this Zone. Only this ZONE may allocate codes beginning with these prefixes
	StartValue	*xsd:normalizedString*	0:1	Start range for codes allocated by Organisation under this CODESPACE ASSIGNMENT. +v1.1
	EndValue	*xsd:normalizedString*	0:1	End range for codes allocated by Organisation under this CODESPACE ASSIGNMENT. +v1.1
	MaximumLength	*xsd:normalizedString*	0:1	Maximum Length for code. +v1.1
	Description	*xsd:string*	0:1	Description of CODESPACE ASSIGNMENT.
«FK»	***TypeOfCodespace AssignmentRef***	*TypeOfCodespace-AssignmentRef*	0:1	Classification of TYPE OF CODESPACE ASSIGNMENT. +v1.1

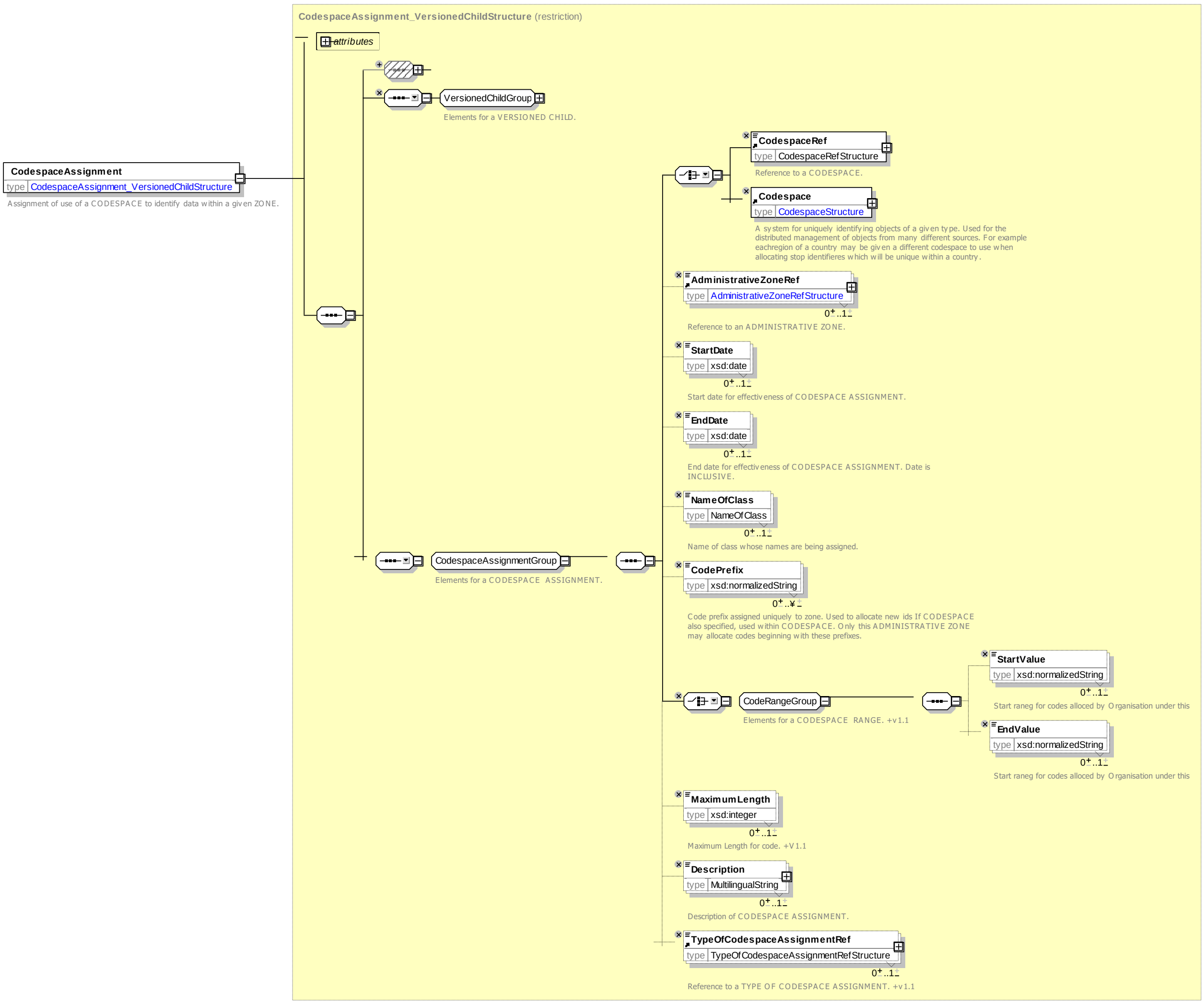


Figure 109 — *CodespaceAssignment – XSD*

7.4.7.4.7 TypeOfOrganisation – Model Element

A classification for the ORGANISATIONs according to their activity, e.g. a public transport company, an IT company, etc.).

Table 66 — *TypeOfOrganisation – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF ORGANISATION PART inherits from TYPE OF ENTITY.
«FK»	***id***	*TypeOfOrganisationIdType*	1:1	Identifier of TYPE OF ORGANISATION.

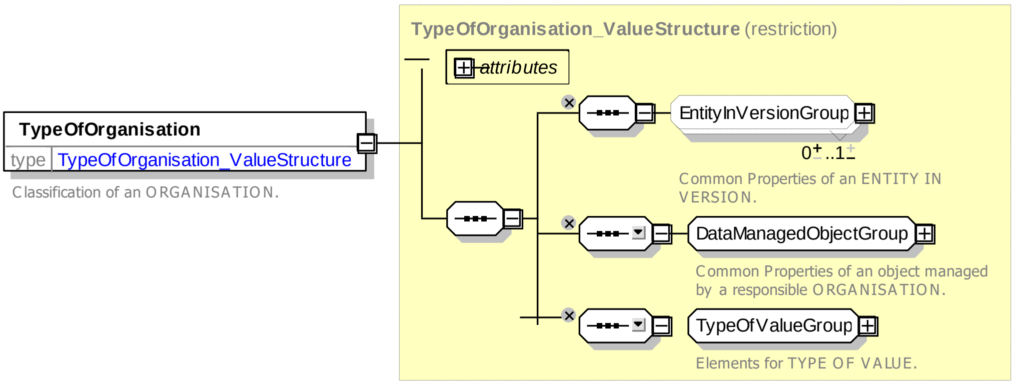


Figure 110 — *TypeOfOrganisation – XSD*

7.4.7.4.8 TypeOfOrganisationPart – Model Element

A classification for the ORGANISATION PARTs according to their activity, e.g. a public transport company operating department, the marketing department, etc.).

Table 67 — *TypeOfOrganisationPart – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF ORGANISATION PART inherits from TYPE OF ENTITY.
«FK»	***id***	*TypeOfOrganisationPartIdType*	1:1	Identifier of TYPE OF ORGANISATION PART.

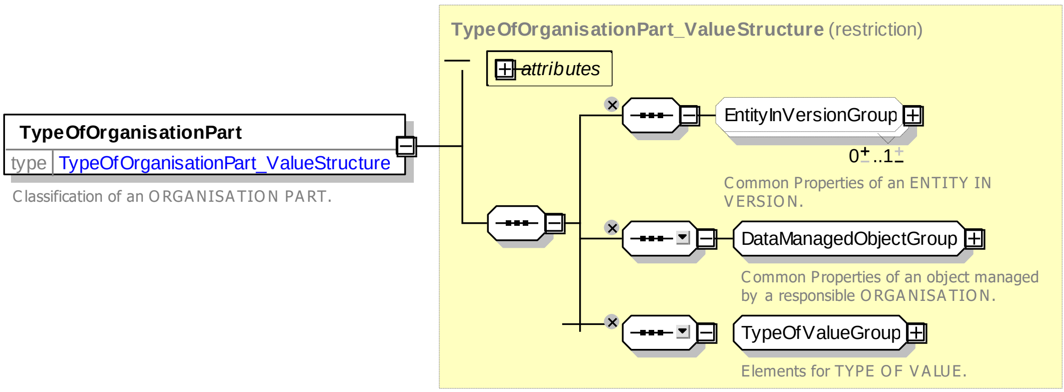


Figure 111 — *TypeOfOrganisationPart – XSD*

7.4.7.4.9 TypeOfOperation – Model Element

A classification of operations to express the different functional roles of a DEPARTMENT.

Table 68 — *TypeOfOperation – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE OF OPERATION inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfOperationIdType*	1:1	Identifier of TYPE OF OPERATION.

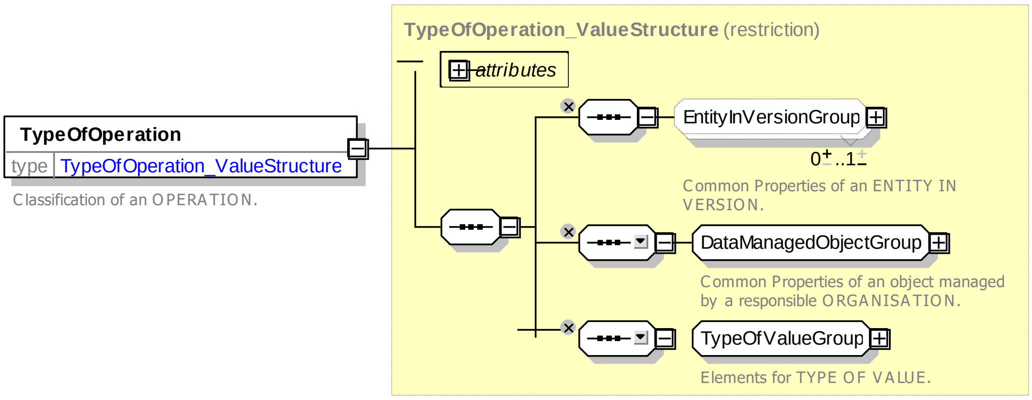


Figure 112 — *TypeOfOperation – XSD*

7.4.7.4.10 TypeOfCodespaceAssignment – Model Element

A classification of CODESPACE ASSIGNMENT.

Table 69 — *TypeOfCodespaceAssignment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE OF CODE SPACE ASSIGNMENT inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfCodespace-AssignmentIdType*	1:1	Identifier of TYPE OF CODESPACE ASSIGNMENT.

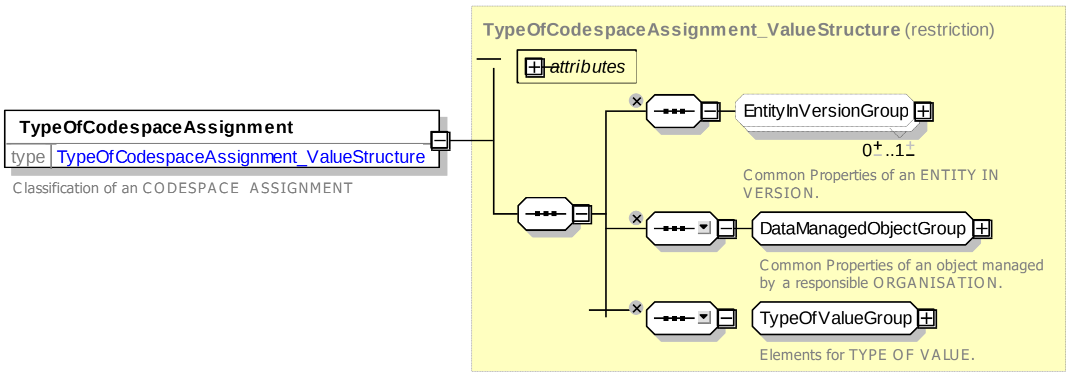


Figure 113 — *TypeOfCodespaceAssignment – XSD*

7.4.7.4.11 OrganisationView – Model View Element

A view of a legally incorporated body associated with any aspect of the transport system. In certain places it is useful to use this instead of a simple reference so that some of the properties of the ORGANISATION derived through the relationship, such as the name and type, can be shown as well.

Table 70 — *OrganisationView* – View Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DerivedView*</u>	::>	ORGANISATION VIEW inherits from DERIVED VIEW.
«FK»	***OrganisationRef***	(*OrganisationRef*)	1:1	Reference to an ORGANISATION.
	Name	*xsd:normalizedString*	1:1	Name of ORGANISATION.
	ShortName	*MultilingualString*	0:1	Short name of ORGANISATION.
	LegalName	*MultilingualString*	0:1	Legal name of ORGANISATION.
	TradingName	*MultilingualString*	0:1	Trading name of ORGANISATION.
«cntd»	***alternativeNames***	<u>*AlternativeName*</u>	0:*	Alternative names for he ORGANISATION.
«cntd»	***ContactDetails***	<u>*ContactDetails*</u>	1:1	ContactDetails for ORGANISATION.

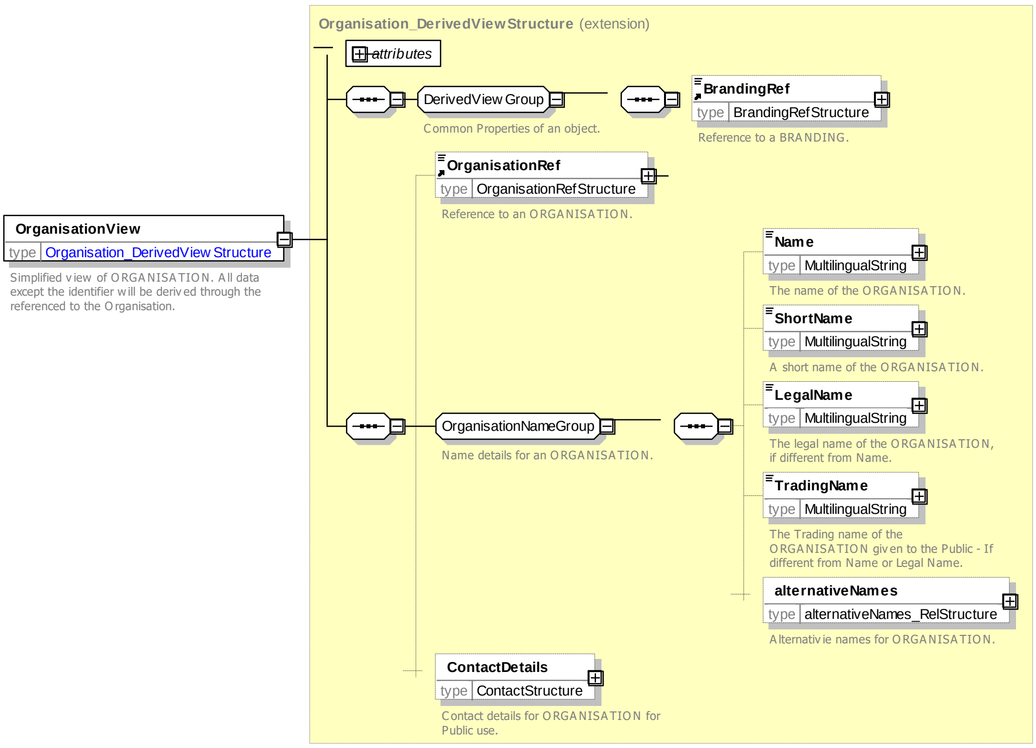


Figure 114 — *OrganisationView – XSD*

7.4.7.5 XML Examples of Organisations

For examples of ORGANISATIONs, See the Transport Organisation Section.

7.5 Generic Frames

As described above, VERSION FRAMEs are used to organise data elements for exchange. There are specific types of VERSION FRAME for each functional area of NeTEx, for example SITE FRAME, TIMETABLE FRAME, etc. These 'explicit' frames are described in the respective NeTEx functional parts, along with the elements they contain. There are in addition two types of explicit VERSION FRAME that have a general purpose and so are described here as part of the framework.

- COMPOSITE FRAME – A Container used to group other frames that meet the same validity conditions.
- GENERAL FRAME – A general purpose frame that can contain an arbitrary group of ENTITies.

7.5.1 Composite Frame

7.5.1.1 Composite Frame – Physical Model

The COMPOSITE FRAME is used to group other frames that have the same VALIDITY CONDITIONs.

A COMPOSITE FRAME may contain another COMPOSITE FRAME.

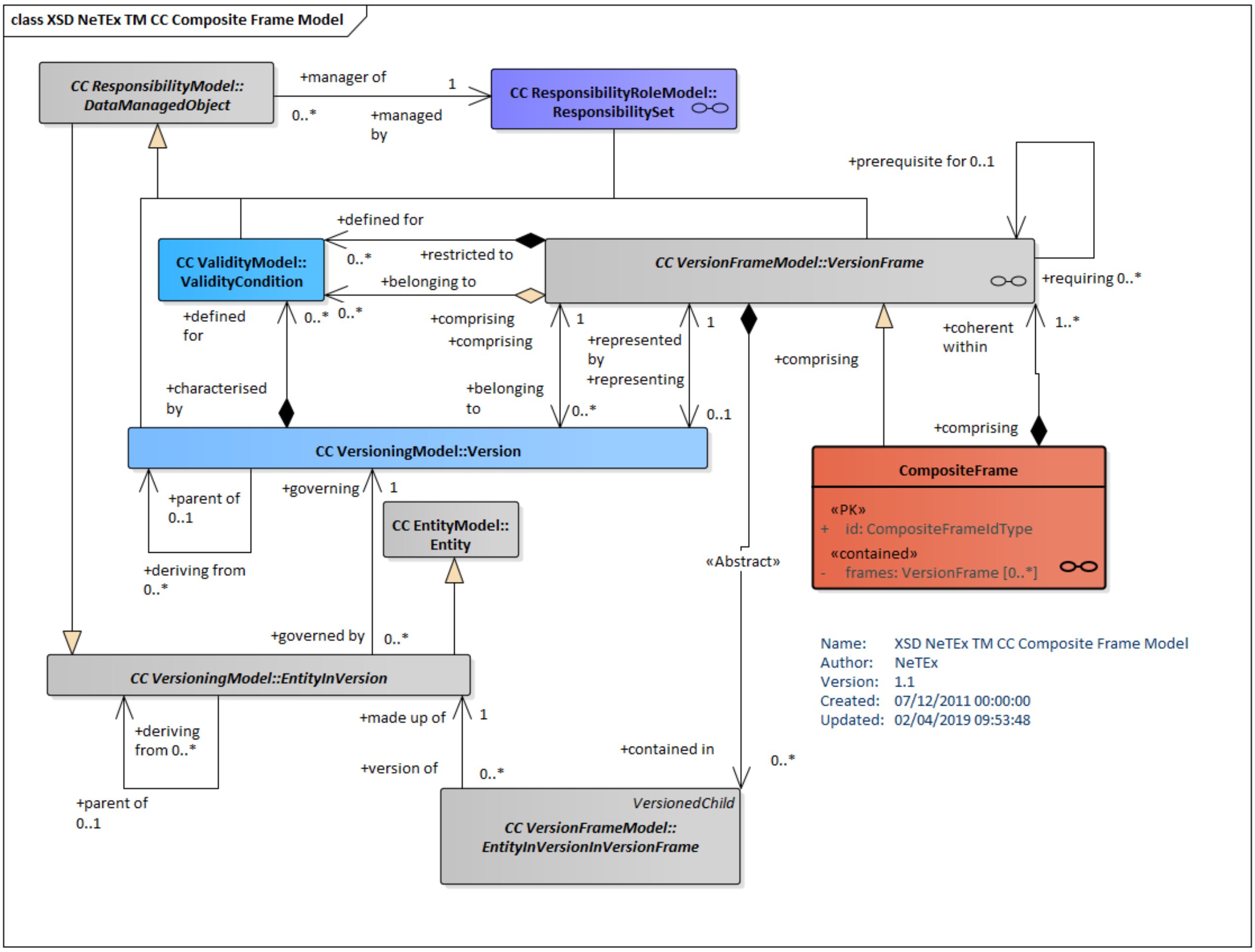


Figure 115 — Composite Frame – Physical MODEL (UML)

7.5.1.2 CompositeFrame – Attributes and XSD

7.5.1.2.1 CompositeFrame – Model Element

A set of VERSION FRAMEs to which the same VALIDITY CONDITIONs have been assigned.

Table 71 — *CompositeFrame – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionFrame*		COMPOSITE FRAME.inherits from VERSION FRAME
«PK»	***id***	*CompositeFrameIdType*	1:1	Identifier of COMPOSITE FRAME.
«cntd»	***frames***	*VersionFrame*	1:*	Frames contained in COMPOSITE FRAME.

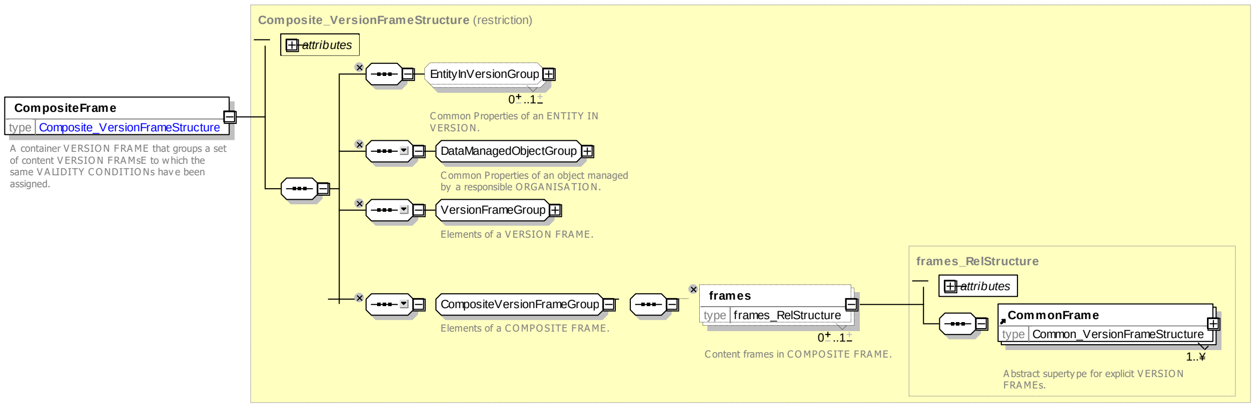


Figure 116 — *CompositeFrame – XSD*

7.5.2 General Frame

7.5.2.1 General Frame – Physical Model

The General FRAME is for general purpose use and may contain any type of NeTEx object, or references to any type of NeTEx object.

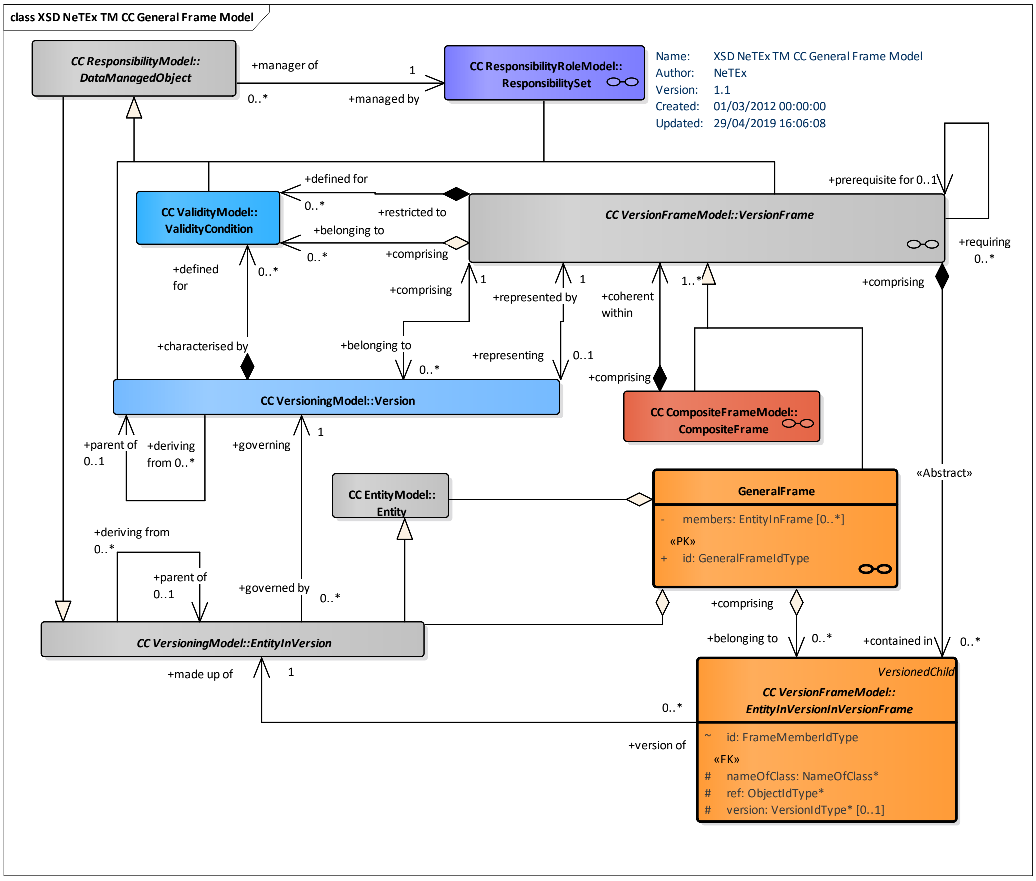


Figure 117 — General Frame – Physical Model (UML)

7.5.2.2 GeneralFrame – Attributes and XSD

7.5.2.2.1 GeneralFrame – Model Element

A set of data of any sort to which the same validity period has been assigned. A GENERAL FRAME may contain:

— Any NeTEx ENTITY (using the Entity_Entity element).

— Any NeTEx DATA MANAGED OBJECT.

Table 72 — *GeneralFrame – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionFrame*</u>	::>	GENERAL FRAME inherits from VERSION FRAME.
«PK»	***id***	*GeneralFrameIdType*	1:1	Identifier of GENERAL FRAME.
«cntd»	***members***	<u>*(EntityInFrame)*</u>	0:*	Members of GENERAL FRAME: a DATA MANAGED OBJECT, an ENTITY_ENTITY or reference to a DATA MANAGED OBJECT (***GeneralFrameMember***).

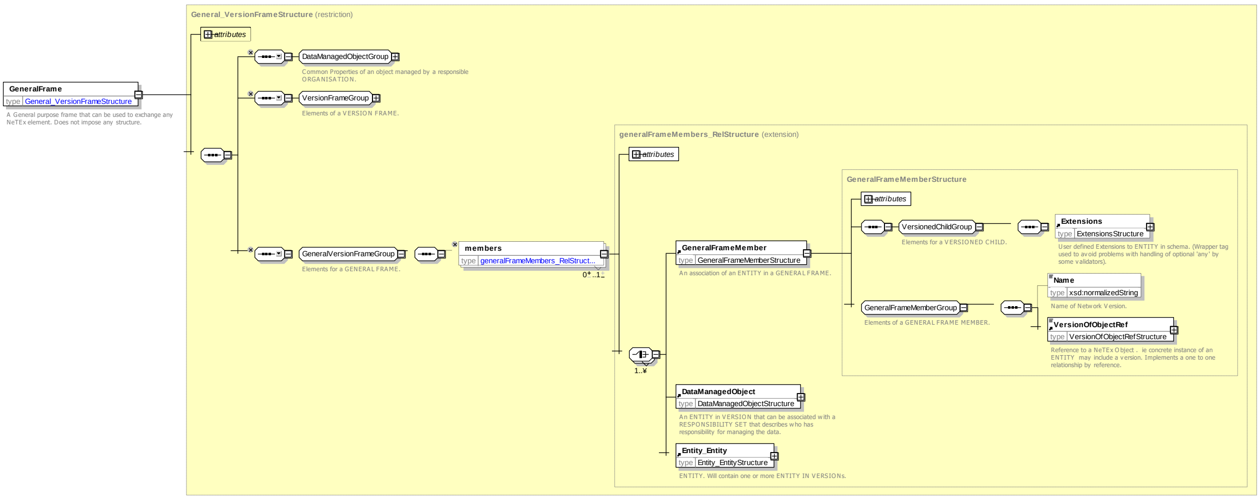


Figure 118 — *GeneralFrame – XSD*

7.5.2.3 XML Examples of Generic Frames

7.5.2.3.1 General Frame – XML Fragment Example

The following example shows the use of a GENERAL FRAME to exchange an arbitrary set of elements in any order.

EXAMPLE

```
        <GeneralFrame     version="004"     created="2010-05-21T10:30:51.0Z"     changed="2010-05-21T10:30:51.0Z" modification="revise" responsibilitySetRef="mybus:RS_10" id="mybus:ntwkf001_ALL">
            <Name>ntwkf001 experessed as a delta - only the removed stop is described - Baseline is version 3</Name>
            <members>
                <ServicePattern version="001" id="mybus::SP_001">
                    <Name>From A to C, version 1</Name>
                    <pointsInSequence>
                        <StopPointInJourneyPattern version="any" id="mybus:SP_001_01" order="1">
                            <ScheduledStopPointRef version="002" ref="mybus:SSP0001A"/>
                        </StopPointInJourneyPattern>
                        <StopPointInJourneyPattern version="any" id="mybus:SP_001_02" order="2">
                            <ScheduledStopPointRef version="002" ref="mybus:SSP0002B"/>
                        </StopPointInJourneyPattern>
                    </pointsInSequence>
                </ServicePattern>
                <ScheduledStopPoint version="any" id="mybus:SSP0001A">
                    <Name>Haltstelle A</Name>
                    <Description>Stop A</Description>
                </ScheduledStopPoint>
                <ServiceLink version="any" id="mybus:SL_AtoB01">
                    <Name>Version one of Link from A to B</Name>
                    <Distance>1.12</Distance>
                    <FromPointRef version="002" ref="mybus::SSP0001A"/>
                    <ToPointRef version="002" ref="mybus::SSP0002B"/>
                </ServiceLink>
                <ScheduledStopPoint version="any" id="mybus::SSP0002B">
                    <Name>Haltstelle B</Name>
                    <Description>Version one of stop B</Description>
                </ScheduledStopPoint>
            </members>
        </GeneralFrame>
```

7.5.2.3.2 Use of a General Frame for Delta – XML Fragment Example

The following example shows the use of a GENERAL FRAME to exchange delta changes.

EXAMPLE

```
        <GeneralFrame     version="005"     created="2010-05-21T10:30:51.0Z"     changed="2010-05-21T10:30:51.0Z" modification="revise" responsibilitySetRef="mybus:RS_10" id="mybus:ntwkf001_Delta">
            <Name>ntwkf001 experessed as a delta - only the removed stop is described - Baseline is version 3</Name>
            <BaselineVersionFrameRef version="003" ref="mybus:ntwkf001"/>
            <members modificationSet="changesOnly">
                <GeneralFrameMember modification="delete" id="mybus:ntwkf001_05">
                    <ServiceLinkRef version="002" ref="mybus:SL_BtoA01"/>
                </GeneralFrameMember>
                <GeneralFrameMember modification="revise" id="mybus:ntwkf001_02">
                    <ScheduledStopPointRef version="001" ref="mybus:SSP0003C"/>
                </GeneralFrameMember>
                <GeneralFrameMember modification="new" id="mybus:ntwkf001_06">
                    <ServiceLinkRef version="001" ref="mybus:SL_BtoC01"/>
                </GeneralFrameMember>
                <GeneralFrameMember modification="new" id="mybus:ntwkf001_07">
                    <ServicePatternRef version="001" ref="mybus:SP_001"/>
                </GeneralFrameMember>
            </members>
        </GeneralFrame>
```

7.6 Generic Framework Model

The NeTEx Generic Framework model defines common framework objects and relationships that are used as a basis for defining the elements of the NeTEx functional models. The framework defines common abstract supertypes that can be specialised to create the concrete elements of the NeTEx modules.

7.6.1 Generic Framework – Model Dependencies

The Framework Models extend the core NeTEx models for responsibility, versioning etc. so that all framework elements can be versioned and managed.

- The GROUPING model provides a means of grouping elements.
- The POINT & LINK model provides a model for defining 1D points and 2D links.
- The LINK SEQUENCE model provides a means of defining graphs of points and links such as are commonly found in layered PT models.
- The ZONE model provides a model for defining 2D zones (with prossible 1D point centroid).
- The PROJECTION model provides a means of defining mappings between different graphs of POINTs and LINKs.
- The PLACE model provides a model for defining named places and links between them.
- Three utility models provide standard definitions of base types:
  - The LOCATION Model defines location related elements such as coordinates. It references a core subset of the GML schema.
  - UNITs model defines reusable base types for common quantitative dimensions such as distance and weight.
  - The UTILITY TYPES model defines common reusable base types such as an email format, time zone, etc.

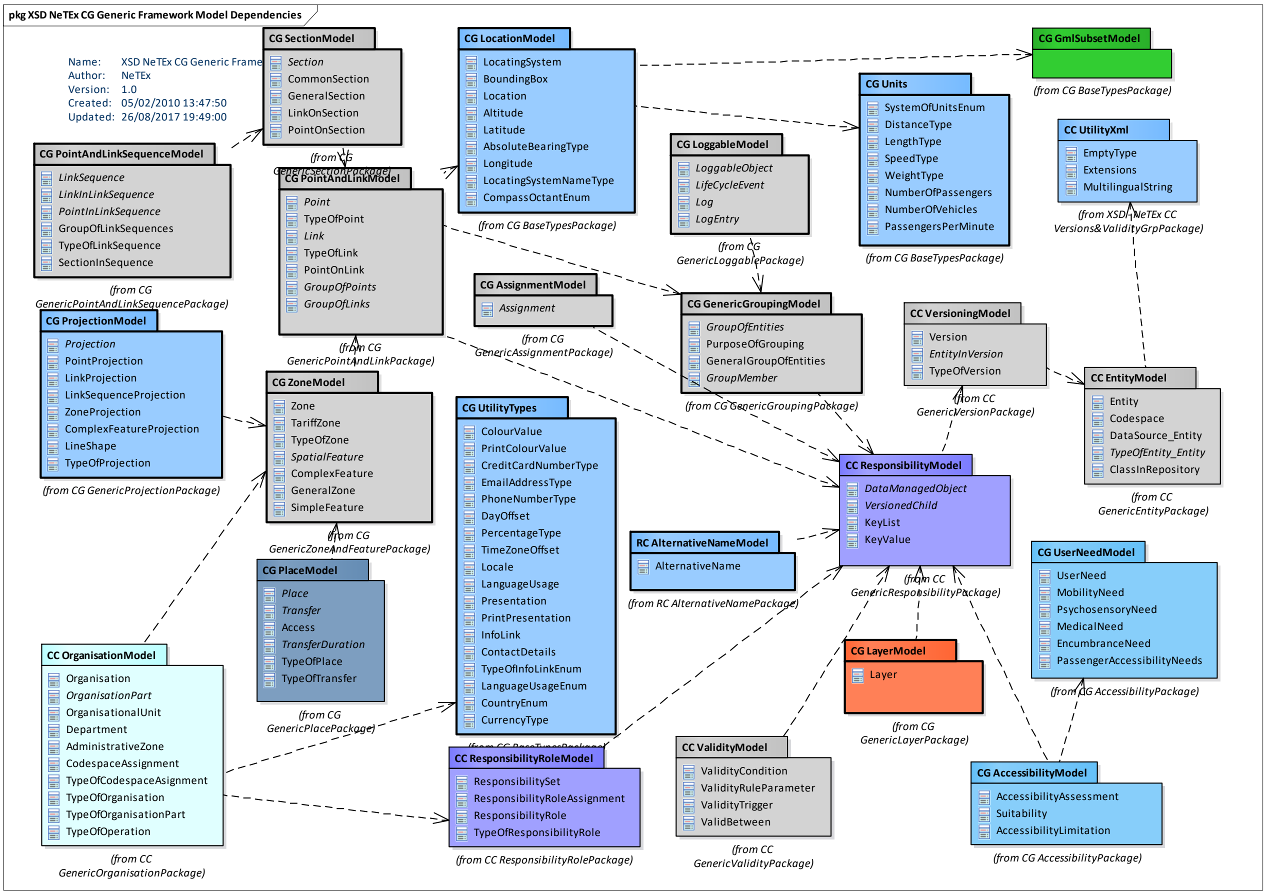


Figure 119 — General Framework – Model Dependencies (UML)

7.6.2 Unit & Utility Base Types

Some reusable base types and units are defined for NeTEx. As these definitions do not inform the conceptual model, they are only described at the physical and XSD level.

7.6.2.1 Unit & Utility Types – Model Dependencies

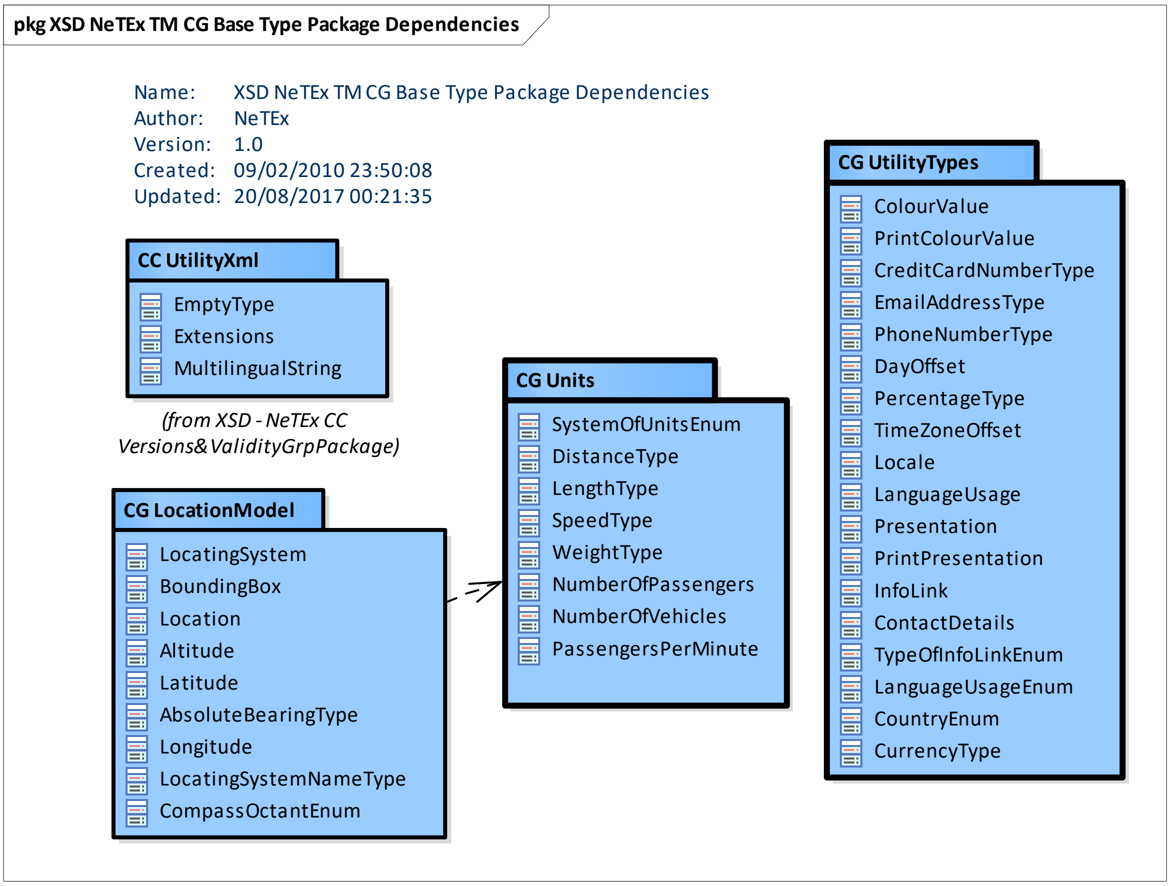


Figure 120 — NeTEx Base Types – PackageModel Dependencies (UML)

7.6.2.2 Core Base types– Physical Model

The core utility types provide standardised definitions of low level data types that are used throughout the framework – an internationalised string, an empty type and an extension type, used to wrap arbitrary user extensions.

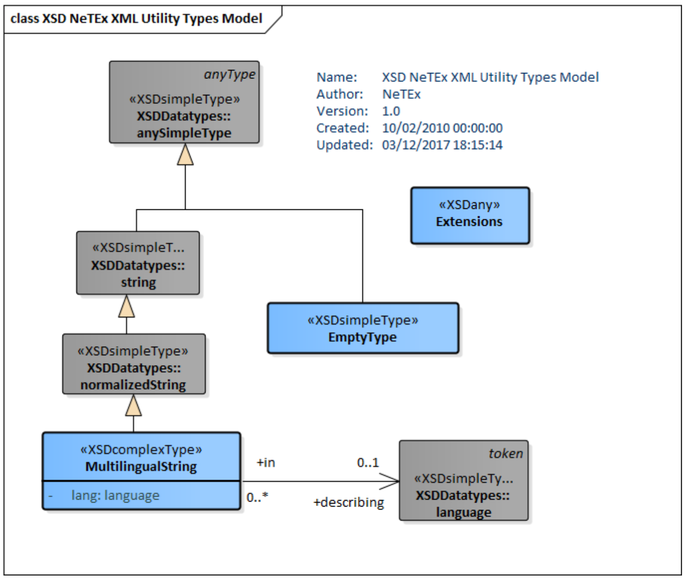


Figure 121 — NeTEx core base types – Physical Model (UML)

7.6.2.3 Unit Types – Physical Model

The UNIT types model defines reusable base types for common quantitative measures such as distance and weight.

Defining such type – as opposed to using undifferentiated integer or floating types for dimensioned amounts - enables strong typing, and facilitates consistency checking and interoperability at both the conceptual and implementation level.

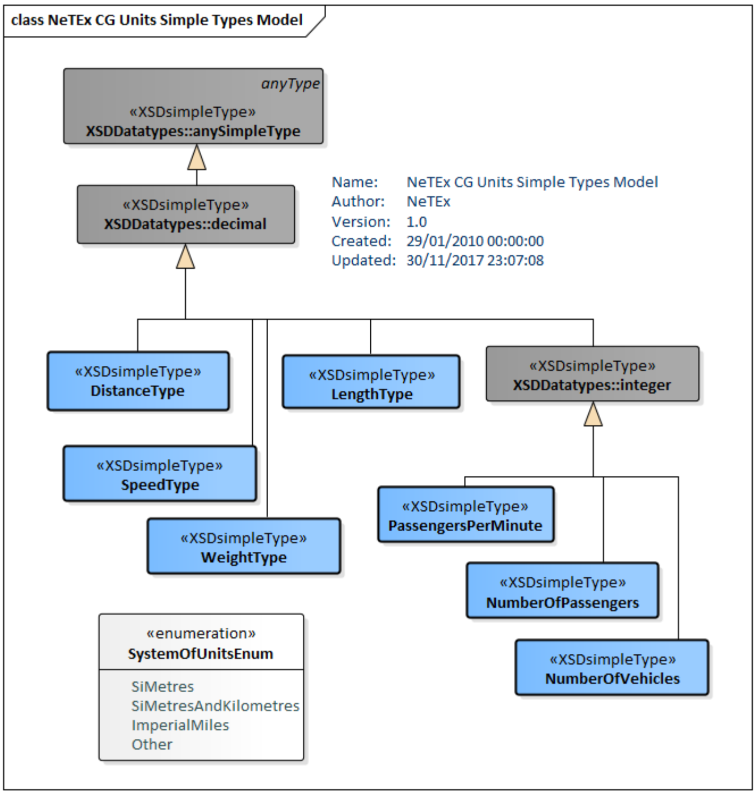


Figure 122 — NeTEx Simple Types – Physical Model (UML)

Four of them are generic and based on SI units:

Table 73 — *Generic units*

Name	Description	XML Base Type
LengthType	Distance in units. Default unit is SI metres as defined by http://www.ordnancesurvey.co.uk/xml/resource/units.xml#metres	*xsd:decimal*
DistanceType	Distance in units. Default unit is SI Kilometres Distance (Kilometres) as defined by http://www.ordnancesurvey.co.uk/xml/resource/units.xml#kilometres	*xsd:decimal*
WeightType	Weight. Default unit is kilos as defined by http://www.ordnancesurvey.co.uk/xml/resource/units.xml#kg	*xsd:decimal*
SpeedType	Speed in metres per second	*xsd:decimal*

Three other types describe cardinal dimensions for measurements specific to public transport.

Table 74 — Measurement units specific to public transport

Name	Description	XML Base Type
PassengersPerMinute	Number of passengers per minute – capacity	*xsd:integer*
NumberOfPassengers	Number of passengers – capacity	*xsd:integer*
NumberOfVehicles	Type for Number of Vehicles - capacity	*xsd:integer*

7.6.2.4 Unit Types – Attributes and XSD

7.6.2.4.1 *SystemOfUnits* – Allowed Values

Allowed values for default units to use ***SystemOfUnits*** **(***SystemOfUnitsEnum*). These dictate the dimensional units for *LengthType*, *DistanceType* and *WeightType*.

Table 75 — *SystemOfUnits* – Allowed Values

Value	**Description**
SiMetres	SI metric. - *DistanceType - Metres* - *LengthType* - Metres - *WeightType* - Kilos
SiMetresAndKilometres	SI - *DistanceType* - Kilometres - *LengthType* - Metres - *WeightType* - Kilos
other	Other system of units e.g. miles

7.6.2.5 Utility types– Physical Model

The utility types provide standardised definitions of a number of low level data types such as email address, time zone, etc.

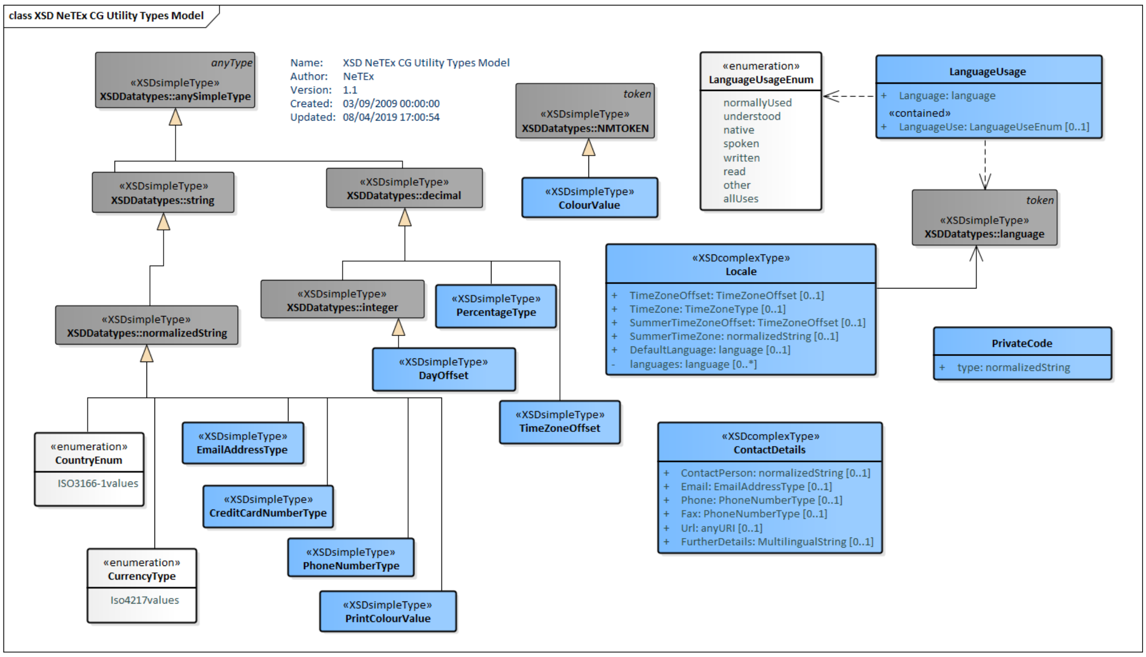


Figure 123 — NeTEx Basic Utility Types – Physical Model (UML)

7.6.2.6 Utility Types: Presentation– Physical Model

The PRESENTATION and PRINT PRESENTATION elements are used to specify colour and other presentation aspects for certain ENTITies.

An INFOLINK can be used to link certain ENTITies to external web pages.

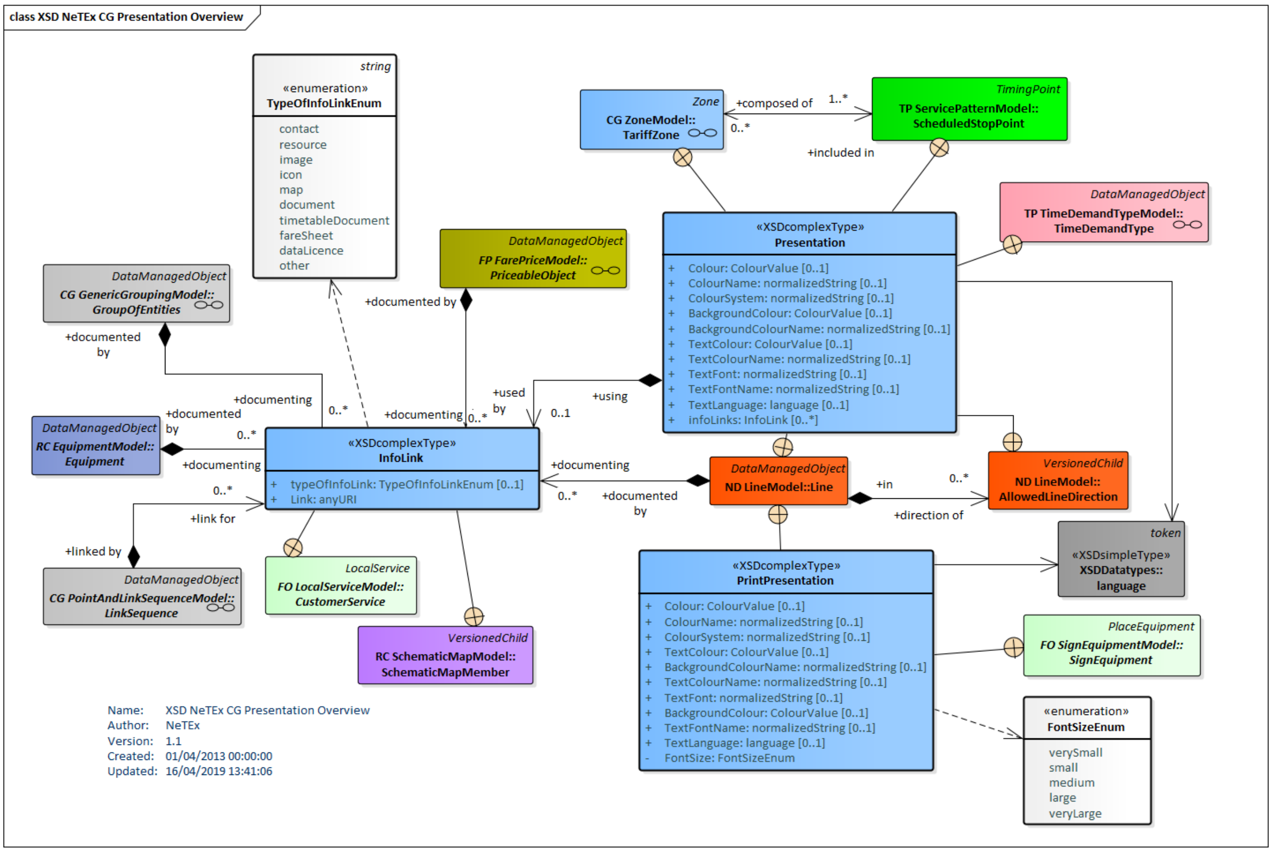


Figure 124 — NeTEx Presentation Utility Types – Physical Model (UML)

The available NeTEx utility types are shown in the following table. Some types are simple and are not described further. Others are complex and are described further below.

Table 76 — *Utility Types* – Elements

Type	Description	XML Base Type
Locale	Common locale dependent properties (time zone, language, etc.)	*Complex*
Presentation	Type describing common presentation properties.	*Complex*
PrintPresentation	Type describing common print presentation properties.	*Complex*
InfoLink	Link to any URI (Uniform Resource Identifier: string of characters used to identify a name or a resource on the Internet) referring any kind of object : web page, image, video, pdf file, etc.	*Complex*
InfoLinkEnum	Enumeration of uses of an Infolink link	*Enumeration*
FontSizeEnum	Enumeration of relative sizes of font	*Enumeration*
ColourValue	Type for a Colour Value as 3 hexadecimal RGB, red, green, and blue levels.	*xsd:hexBinary*
LanguageUsageList	Enumeration of uses of language	*Enumeration*

TimezoneOffset	Timezone offset for the local time from GMT. May be fractional.	*xsd:decimal*
PhoneNumberType	International phone number, country code plus number, *e.g. '+41 6756001' etc.*	*xsd:normalizedString*
EmailAddressType	Email description in standard format foo@bar.org etc.	*xsd:normalizedString*
Country	ISO country code (http://www.iso.org/iso/english_country_names_and_code_elements)	*xsd:normalizedString*
ContactDetails	Set of details for a contact: Email, Phone, Link and comment.	*Complex*

7.6.2.7 Utility Types – Attributes and XSD

7.6.2.7.1 Locale – Model Type

Parameters relating to locale to be used when presenting data.

Table 77 — *Locale* – Type

Classification	Name	Type	Cardinality	Description
	TimeZoneOffset	*TimeZoneOffset*	0:1	Time zone of Entity as offset in hours from GMT. plus or minus. May be decimal for fractional differences.
	TimeZone	*xsd:normalized-String*	0:1	Name of Time zone associated with LOCALE.
	SummerTimeZoneOffset	*TimeZoneOffset*	0:1	Time zone as an offset in hours or part hours from GMT. Can be plus or minus.
	SummerTimeZone	*xsd:normalized-String*	0:1	Name of Summer Time zone associated with LOCALE.
	DefaultLanguage	*xsd:language*	0:1	Default language of LOCALE.
	languages	*LanguageUsage*	0:*	Other languages supported in LOCALE.

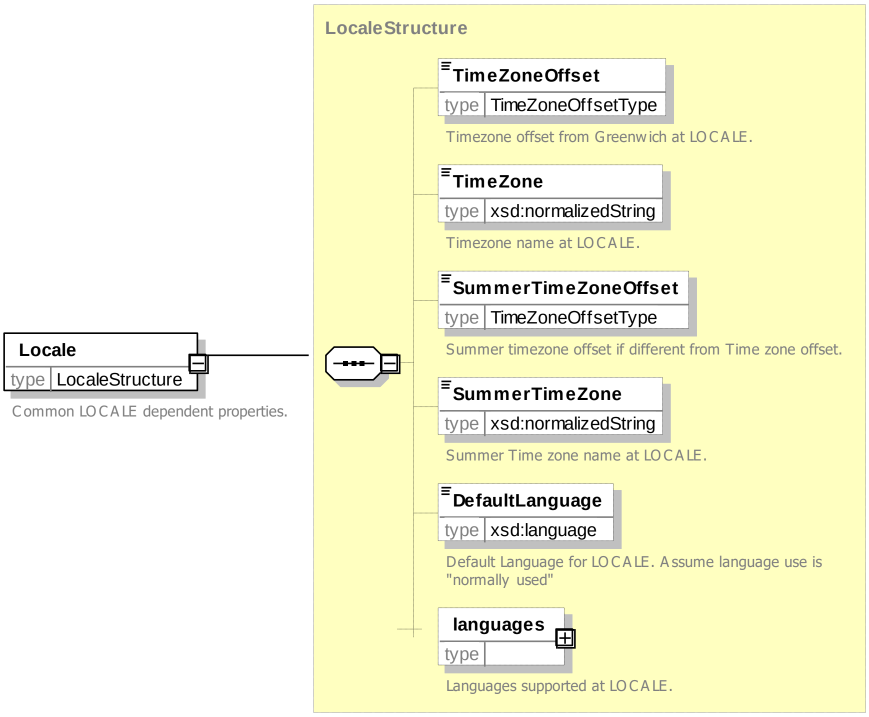


Figure 125 — *Locale* – Element

7.6.2.7.2 LanguageUsage – Model Type

A description of language usage available at the LOCALE, based on UN standard terms.

Table 78 — *LanguageUsage – Structure*

Classification	Name	Type	Cardinality	Description
«enum»	***Language***	*xsd:language*	1:1	Language described by this usage.
«cntd»	***LanguageUse***	*LanguageUsageEnum*	0:1	Usage of language available.

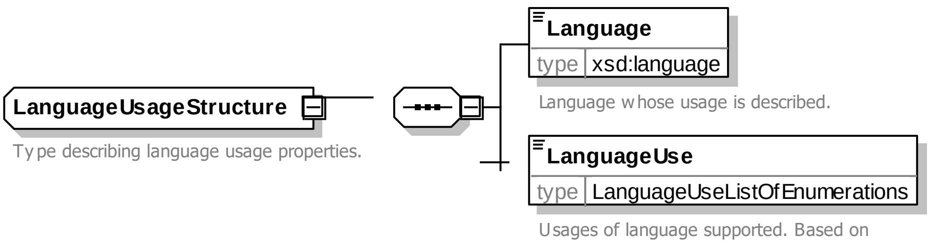


Figure 126 — *LanguageUsage* – Element

7.6.2.7.2.1 LanguageUsage – Allowed Values

Allowed values for ***LanguageUsage*** (*LanguageUsageEnum*),

Table 79 — *LanguageUsage* – Allowed Values

Value	Description
normallyUsed	Language is that normally used at location.
understood	Language is understood but not necessarily spoken or read.
native	Language is the native tongue
spoken	Language is spoken but not necessarily written
written	Language is written but not necessarily spoken
read	Language is read but not necessarily spoken or written
other	Other language usage
allUses	Language can be used in all ways

7.6.2.7.3 Presentation – Model Type

Set of PRESENTATION values used to control appearance of an element.

Table 80 — *Presentation* – Type

Classification	Name	Type	Cardinality	Description
	Colour	*ColourValueType*	0:1	Default RGB colour value to use for ENTITY in user interface, e.g. LINE colour.
	ColourName	*xsd:normalisedString*	0:1	Text name of colour to use for ENTITY in user interface, e.g. LINE colour.
	ColourSystem	*xsd:normalisedString*	0:1	Text name of colour system isued for ColourName and TextColourName. For example • RAL: https://en.wikipedia.org/wiki/RAL_colour_standard; • DIN: 6164 http://www.dtpstudio.de/atlas/farbsysteme/DIN 6164_bs00_3.htm; • Pantone (NB Pantone is a proprietary) ystem, not a standard, etc.
	Background-Colour	*ColourValueType*	0:1	Defaul RGB background colour value for text and graphics associated with ENTITY.
	Background-ColourName	*xsd:normalizedString*	0:1	Name (in ***ColourSystem***) of the background colour for text and graphics associated with ENTITY. +v1.1.
	TextColour	*ColourValueType*	0:1	Default RGB colour value to use for text for ENTITY in user interface, e.g. colour for name of LINE.
	TextColourName	*xsd:normalizedString*	0:1	Text name (in ***ColourSystem***) of colour to use for text for ENTITY in user interface, e.g. name of colour for name of LINE.
	TextFont	*xsd:normalizedString*	0:1	Identifier of font for text. +v1.1
	TextFontName	*xsd:normalizedString*	0:1	Name of font for text. +v1.1
	TextLanguage	*xsd:language*	0:1	Identifier of default language of text for ENTITY in user interface.

«cntd»	***infoLinks***	*InfoLink*	0:*	URL for image associated with entity e.g. icon.

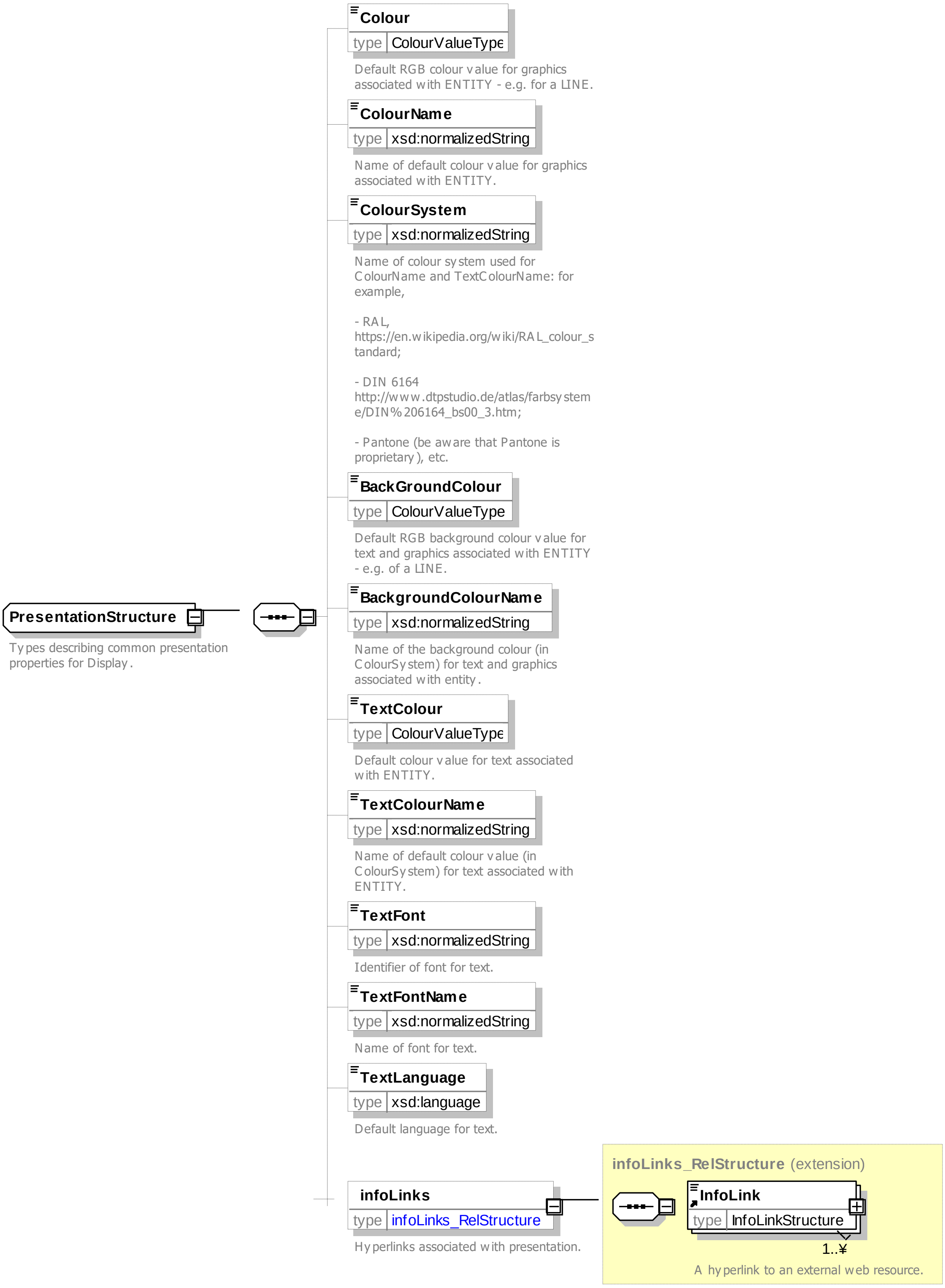


Figure 127 — *Presentation – XSD*

7.6.2.7.4 PrintPresentation – Model Type

Set of PRESENTATION values used to control appearance of an element in printed material.

Table 81 — *PrintPresentation* – Type

Classific ation	Name	Type	Cardinality	Description
	Colour	*ColourValueType*	0:1	Identifier of colour value to use for ENTITY in user interface, e.g. LINE colour.
	ColourName	*xsd:String*	0:1	Default colour value for printed text associated with entity
	TextColour	*ColourValueType*	0:1	Identifier of colour value to use for text for ENTITY in user interface, e.g. colour for name of LINE.
	ColourSystem	*xsd:normalisedString*	0:1	Text name of colour system used for ***ColourName*** and ***TextColourName***.
	Background-Colour	*ColourValueType*	0:1	Defaul RGB background colour value for printed text and graphics associated with ENTITY.
	Background-ColourName	*xsd:normalizedString*	0:1	Name (in ***ColourSystem***) of the background colour for printed text and graphics associated with ENTITY. +v1.1.
	ColourSystem	*xsd:normalisedString*	0:1	Text name of colour system isued for ColourName and TextColourName. See ***Presentation***/
	TextColourName	*xsd:normalizedString*	0:1	Name of Default colour value for printed text associated with entity, eg Pentone name.
	TextFont	*xsd:normalizedString*	0:1	Identifier of font for printed text. +v1.1
	TextFontName	*xsd:normalizedString*	0:1	Name of font for printed text. +v1.1
	TextLanguage	*xsd:language*	0:1	Identifier of default language of printed.
«enum»	***FontSize***	*FontSizeEnum*	0:1	Fomt size name – see below for allowed values.

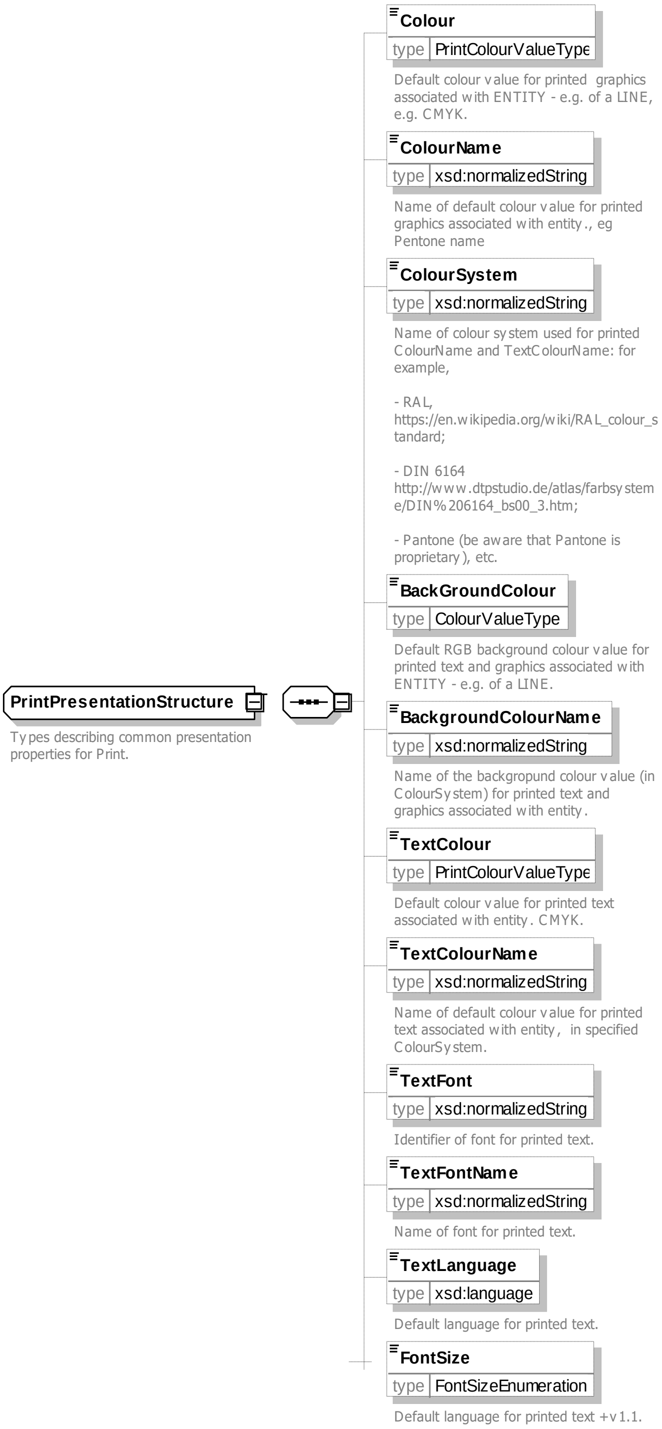


Figure 128 — *PrintPresentation – XSD*

7.6.2.7.4.1 FontSize – Allowed Values

Allowed values for ***FontSize*** (*FontSizeEnumeration)*.

Table 82 — *FontSize* – Allowed Values

Value	Description
verySmall	Very small
small	Small
medium	Medium
large	Large
veryLarge	Very large

7.6.2.7.5 InfoLink – Model Type

Info Link defines a URL with a content type.

Table 83 — *InfoLink* – Type

Classification	Name	Type	Cardinality	Description
	typeOfInfoLink	*TypeOfInfoLinkEnum*	0:1	Classification of INFO LINK. See allowed values below.
	Link	*xsd:anyURI*	1:1	URL associated with INFO LINK.

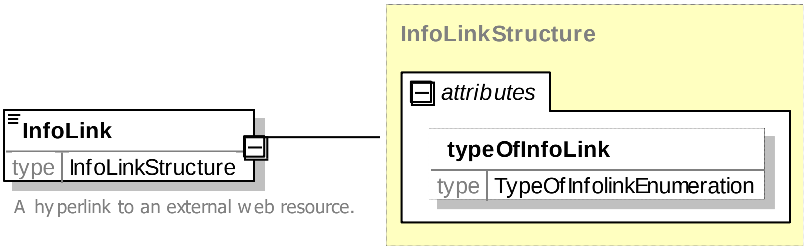


Figure 129 — *InfoLink – XSD*

7.6.2.7.5.1 TypeOfInfoLink – Allowed Values

Allowed values for ***TypeOfInfoLink*** (*TypeOfInfoLinkEnumeration)*

Table 84 — *TypeOfLink* – Allowed Values

Value	Description
contact	Link is a web page of contact details.
resource	Link is an online resource.
info	Link is an information page.
image	Link is an image in a well-known MIME type.
icon	k is an icon.
document	Link is an online document.
timetableDocument	Link is an online timetable document.
fareSheet	Link is an online fare sheet document.
map	Link is an online map document.
dataLicence	Link is a data licence.
other	Other Link.

7.6.2.7.6 ContactDetails – Model Element

Details of a contact.

Table 85 — *ContactDetails – Structure*

Classification	Name	Type	Cardinality	Description
	ContactPerson	*MultilingualString*	0:1	Name of contact person.
	Email	*EmailAddressType*	0:1	Email address in ISO format.
	Phone	*PhoneType*	0:1	Phone number.
	Fax	*PhoneType*	0:1	Fax number.

	Url	*xsd:anyURI*	0:1	Contact web site URL.
	FurtherDetails	*MultilingualString*	0:1	Further details of contact.

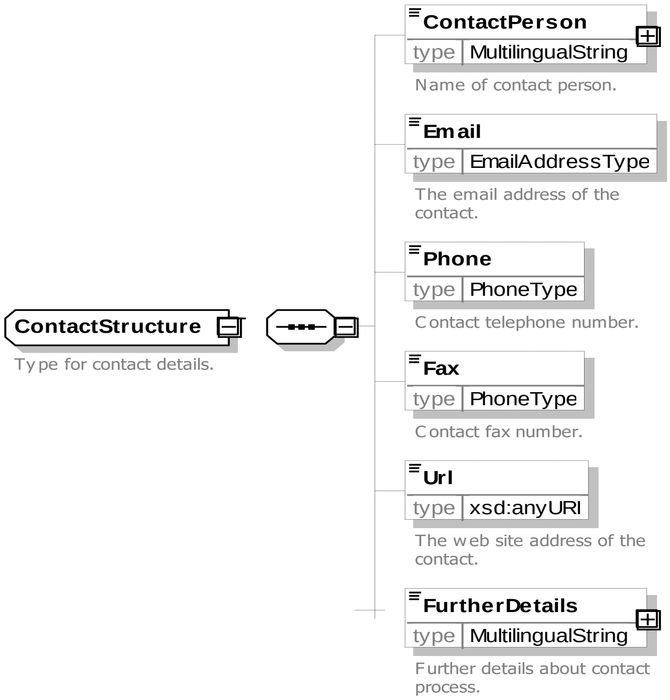


Figure 130 — *ContactDetails – XSD*

7.6.2.7.7 Country – Allowed Values

Allowed values for ISO country identifier (*CountryEnumeration)*.

Table 86 — *Country* – Allowed Values

Value	Description
ISO values	ISO values for code (ISO 3166-1).

7.6.2.7.8 KeyList – Model Element

Set of arbitrary key value pair that can be associated with any DATA MANAGED OBJECT. Provides an extension mechanism.

Table 87 — *KeyList – Element*

Classification	Name	Type	Cardinality	Description
	keyValues	*KeyValue*	1:*	List of KEY VALUE pairs.

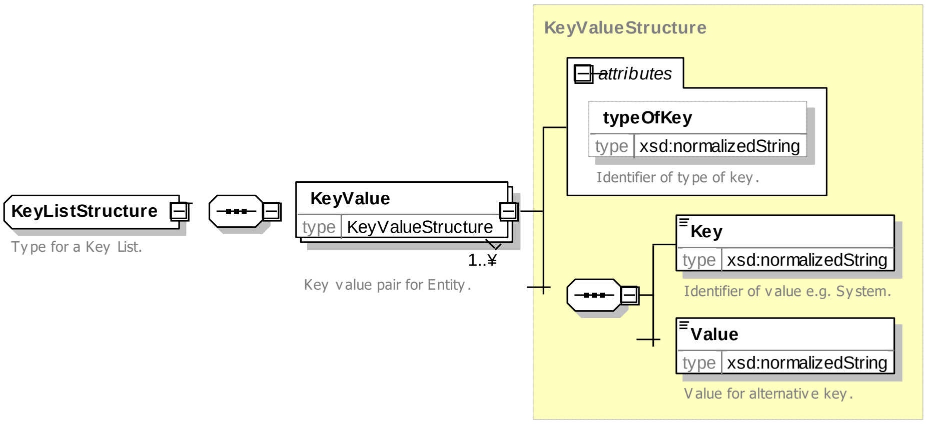


Figure 131 — *KeyList* & *KeyValue*– *XSD*

7.6.2.7.9 KeyValue – Model Element

An arbitrary key value pair which may be associated with a DATA MANAGED OBJECT.

Table 88 — *KeyList* – *Element*

Classification	Name	Type	Cardinality	Description
	typeOfKey	*xsd:normalizedString*	0:1	Type for KEY VALUE.
	Key	*xsd:normalizedString*	1:1	Key for KEY VALUE.
	Value	*xsd:normalizedString*	1:1	Value for KEY VALUE.

7.6.2.7.10 MultilingualString – Model Type

MultilingualString allows the definition of a string in a specific Natural Languages.

Table 89 — *MultilingualString* – Type

Classification	Name	Type	Cardinality	Description
	lang	*xsd:language*	0:1	Language in which the string is expressed.
	textIdType	*xsd:normalizedString*	0:1	Numeric text id associated with text.

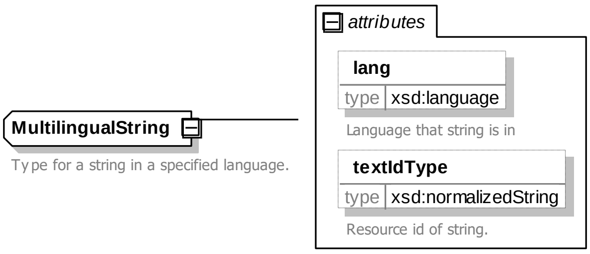


Figure 132 — *MultilingualString* – *XSD*

7.6.2.7.11 PrivateCode – Model Type

PrivateCode allows the association of an arbitrary code with certain elements. This can be used to maintaion compatibility with legacy and external systems.

Table 90 — *PrivateCode* – Type

Classification	Name	Type	Cardinality	Description
	type	*xsd:normalizedString*	0:1	Identifier of type of code.
	value	*xsd:normalizedString*	1:0	Value for private code.

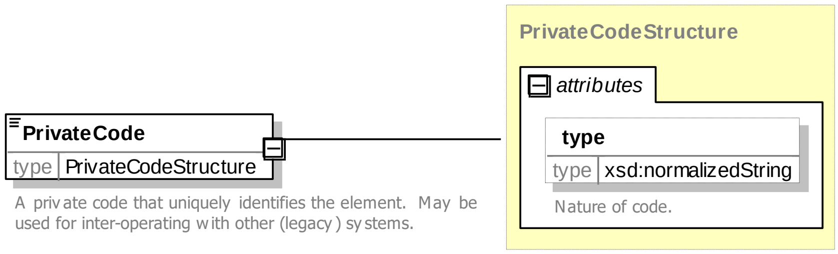


Figure 133 — *PrivateCode – XSD*

7.6.3 Location

7.6.3.1 LOCATION – Conceptual MODEL

The Location provides a representation of the basic coordinates of those entities that are located in space, for example, POINTs and LINKs and ZONEs.

NeTEx uses a subset of the OGC GML (Geographic Mark-up language) schema to defined coordinates. This allows different spatial reference systems (SRS) to be used to express the geographic location of objects of different shape (point, linestring, polygon, and multi-points if using the OGC normalised wording at http://www.opengeospatial.org/standards). The spatial reference defines an ellipsoid, a datum using that ellipsoid, and either a geocentric, geographic or projection coordinate system. The projection also always has a geographic coordinate system associated with it.

Some of the commonly used spatial reference systems are: 4326 - WGS 84 Long Lat, 4269 - NAD 83 Long Lat, 3395 - WGS 84 World Mercator but several hundred are available depending on the country, needed precision, purpose of the location, etc. See http://spatialreference.org/ for more details.

The location of a POINT is dependent on the LOCATING SYSTEM used. Given a LOCATING SYSTEM, every POINT may be located in this system by a LOCATION. One of the classical ways to locate a POINT is to assign coordinates to it. The LOCATION is hence defined by two coordinates in a two-dimensional representation and possibly by a third coordinate relating the point to a surface. Examples of coordinates are x- and y-values in a plane graphic or diagram, degree of longitude and latitude on a globe, GPS-coordinates, the angle to the North and the distance from an origin, etc.

Every LOCATION must be defined according to one and only one LOCATING SYSTEM and must be located at one and only one POINT. Any POINT may be located by one or more LOCATIONs, each of which may refer to only one LOCATING SYSTEM. The LOCATING SYSTEM may be specified implicitly by the context (e.g. on the VERSION FRAME, or on an individual LOCATION.

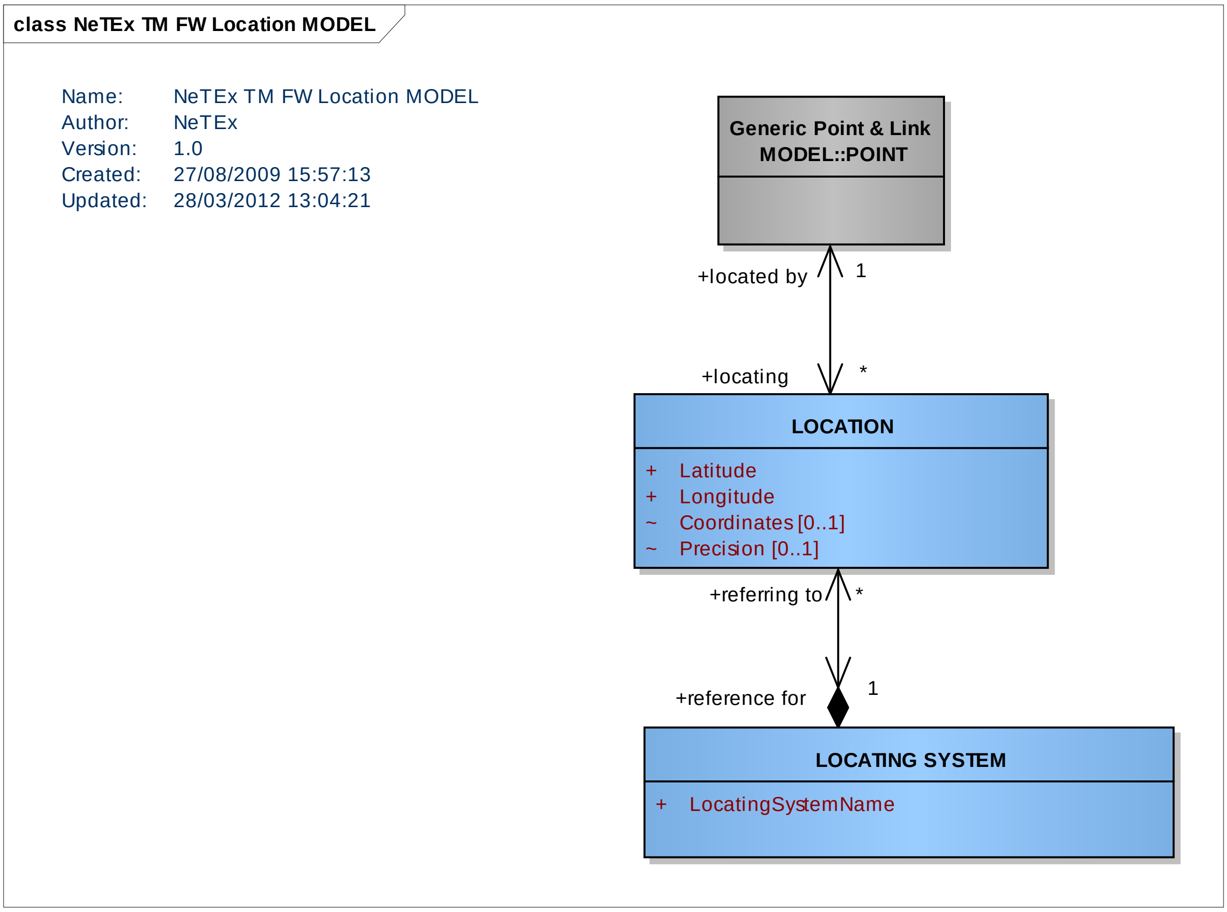


Figure 134 — Location – Conceptual MODEL (UML)

7.6.3.2 Location – Physical Model

The Location Physical model defines LOCATION and LOCATING system and a number of base types.

The NeTEx encoding has special support for an encoding of LOCATION as separate Longitude and Latitude coordinates using the widely used WGS84 geodetic location system (as used by the Global Positioning System, GPS), which can specify a location anywhere on earth. This allows values to be validated using an XML data type and the built-in capabilities of any XML validator.

Alternatively (or as well), locations may be encoded in the coordinates of another LOCATION SYSTEM as specified by a ***SrsName*** and in accordance with the GML standard. Coordinates specified in this way are not validated by the XML but must be checked by an application importing a NeTEx Document.

The LOCATING SYSTEM can be stated at different levels of specificity: a default *SrsName* can be specified at the VERSION FRAME to apply to all LOCATION in the frame which do not specify their own value.

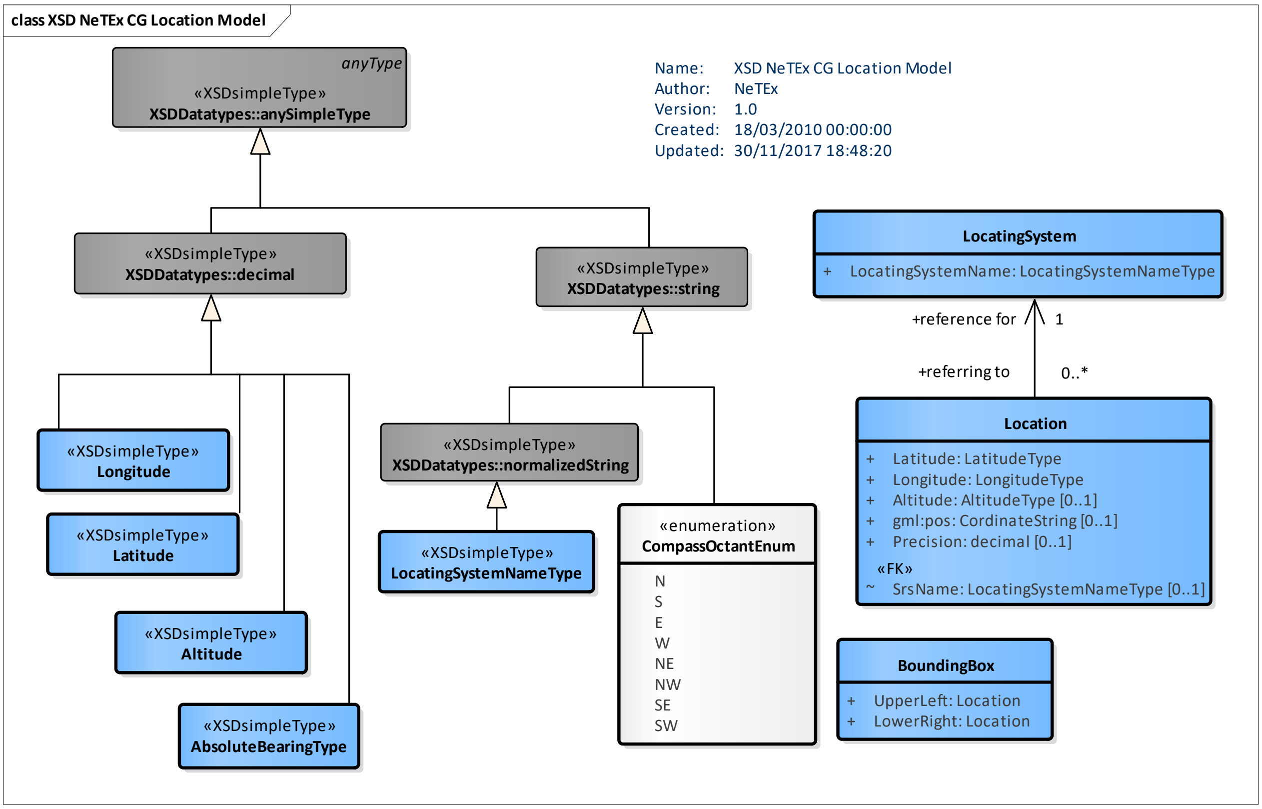


Figure 135 — Location – Physical Model (UML)

The ***srsName*** is usually composed of Author name and the unique Integer ID defined by an authority of the respective spatial reference system. For example, the common WGS 84 longitude-latitude locating system has an ***srsName*** identifier of '*EPSG:4326*'. Similarly the '*Massachusetts State Plane NAD 83 meters'* locating system would be referred to as '*EPSG:26986'*. Many commonly used Spatial Reference IDs have the authority *European Petroleum Survey Group (EPSG*). See http://www.epsg-registry.org/ or http://spatialreference.org/ for more details.

The coordinates themselves are provided as a standard ISO OGC coordinate string (see *gml:Coordinates Type* in http://www.opengeospatial.org/standards/gml).

Table 91 — *Location* – Types

Type	Description	XML Base Type
Latitude	Latitude from Equator: between -90 and + 90.	xsd:decimal
Longitude	Longitude from Greenwich: between -180 and 180 degrees.	xsd:decimal
LocatingSystemNameType	Name of locating System.	xsd:normalizedString
LocatingSystem	Description of System of Coordinates.	Not implemented in XSD
Location	The position of a POINT with a reference to a given LOCATING SYSTEM.	Complex

7.6.3.3 Location – Attributes and XSD

7.6.3.3.1 LocatingSystem – Model Element

The system used as reference for location and graphical representation of the network and other spatial objects.

The LOCATING SYSTEM is not implemented in the XSD, but isreferenced simply by name

Table 92 — LocatingSystem – Element

Classification	Name	Type	Cardinality	Description
	LocatingSystem Name	*LocatingSystemNameType*	1:1	Name of Locating system used for coordinates. Same as SrsName.

7.6.3.3.2 Location – Model Element

The position of a POINT with a reference to a given LOCATING SYSTEM (e. g. coordinates).

Table 93 — *Location – Element*

Classification	Name	Type	Cardinality	Description
«FK»	***srsName***	*LocatingSystemNameType*	0:1	GML id of Type of LOCATING SYSTEM used.
	Longitude	*LongitudeType*	1:1	Longitude of Location.
	Latitude	*LatitudeType*	1:1	Latitude of Location.
	Altitude	*AltitudeType*	0:1	Altitude of Location.
	Coordinates	*CoordinateString gml:pos*	0:1	GML coordinates providing location in a specified Location system.
	Precision	*xsd:decimal*	0:1	Precision of coordinates.

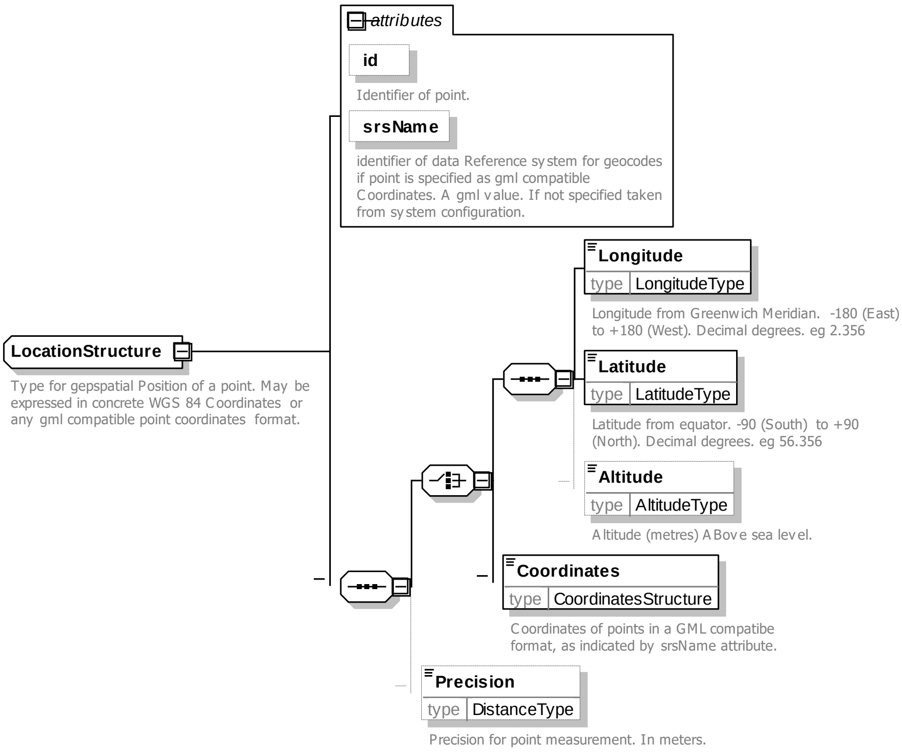


Figure 136 — *Location – XSD*

7.6.3.3.3 CompassOctant – Allowed Values

Allowed values for Compass Octants (*CompassOctantEnumeration)*

Table 94 — *CompassOctant* – Allowed Values

Value	Description
N	North
S	South
E	East
W	West
NE	North East
NW	North West
SE	South East
SW	South West

7.6.3.4 *XML Examples of encoding a Location*

7.6.3.4.1 Encoding a Point Location with WSG84 coordinates – XML Example

The following example shows the use of a *Location* to specify the coordinates of a point.

The first example uses WGS84 latitude and longitude. No explicit srsName needs to be specified.

EXAMPLE

```
<ServiceFrame version="1" id="bar: SF01">
    <Name>Stops for winter timetable for route 24 </Name>
    <routePoints>

        <RoutePoint version="any" id=" bar: 01
            <Name>Point in Road near Stop A</Name>
            <Location>
    <Longitude>53.0000</Longitude>
    <Latitude>0.1000</Latitude>
            </Location>
        </RoutePoint>
        <RoutePoint version="any" id=" bar: 02>
            <Name>Point in Road near Stop B</Name>
            <Location>
    <Longitude>53.0000</Longitude>
    <Latitude>0.2000</Latitude>
            </Location>
```

```
        </RoutePoint>
```

7.6.3.4.2 Encoding a Point Location with Grid coordinates – XML Example

The second example uses UK OS grid coordinates,
EXAMPLE

```
<StopPlace version="any" created="2006-09-11T15:42:00" id="mybus:StopPlace:SP003C">
    <Name>Place C</Name>
    <Centroid>
    <Location>
        <gml:pos srsName="UKOS">524811     170666 </gml:pos>
    </Location>
    </Centroid>
```

7.6.3.4.3 Encoding a Line with WSG84 coordinates – XML Example

The third example uses WGS84 coordinates encoded with a srSName to encode a GML Line String:

EXAMPLE

```
    <ServiceJourney version="any" id="mygtfsxm:ServiceJourney:AB2o">
        <LinkSequenceProjection version="any" id="mygtfsxm:AB2o">
            <gml:LineString srsName="WGS84" gml:id="AB2o">
    <gml:pos>53.00 1.00</gml:pos>
    <gml:pos>53.10 1.10</gml:pos>
    <gml:pos>53.20 1.20</gml:pos>
            </gml:LineString>
        </LinkSequenceProjection>
    </ServiceJourney>
```

7.6.4 Generic Grouping

7.6.4.1 Introduction

There is often a need in public transport to group objects into a set, for example a group of lines, group of points, etc. Some kinds of grouping are very frequent; others are specific to a particular local situation. NeTEx provides an explicit grouping mechanism that can be used for the more commonly found cases, such as GROUP OF LINEs, and a generic grouping mechanism that can be used to group any kind of object.

Grouping may be very useful in situations like:

— Defining a bus network by grouping a set of LINEs together,

— Defining a bus night network built from a subset of the LINEs in a network (or a subset of several networks),

— Defining a group of SCHEDULED STOP POINTs all belonging to the same town,

— etc.

7.6.4.2 GROUPING – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

One or more ENTITies of any type may be grouped using a GROUP of ENTITies.

Objects like POINT, LINK, and LINK SEQUENCE may be grouped by the corresponding entities GROUP OF POINTs, GROUP OF LINKs, and GROUP OF LINK SEQUENCEs respectively.

Each of these groups can be classified by a PURPOSE OF GROUPING. Such a group is the association of specific elements of a given type into a group needed for a particular functional purpose (for example, WIRE ELEMENTs having a specific power supply type). The PURPOSE OF GROUPING refers to the functional purpose for which the associated groups of elements are defined.

Some other types of ENTITY also have an inherent grouping semantic. For example STOP AREA (or also indeed ZONE) incorporates the generic concept of a grouping of POINTs.

The assignment of elements to groups of such elements is represented by many-to-many relationships. An ENTITY can belong to more than one group, either or for the same or a different PURPOSE OF GROUPING.

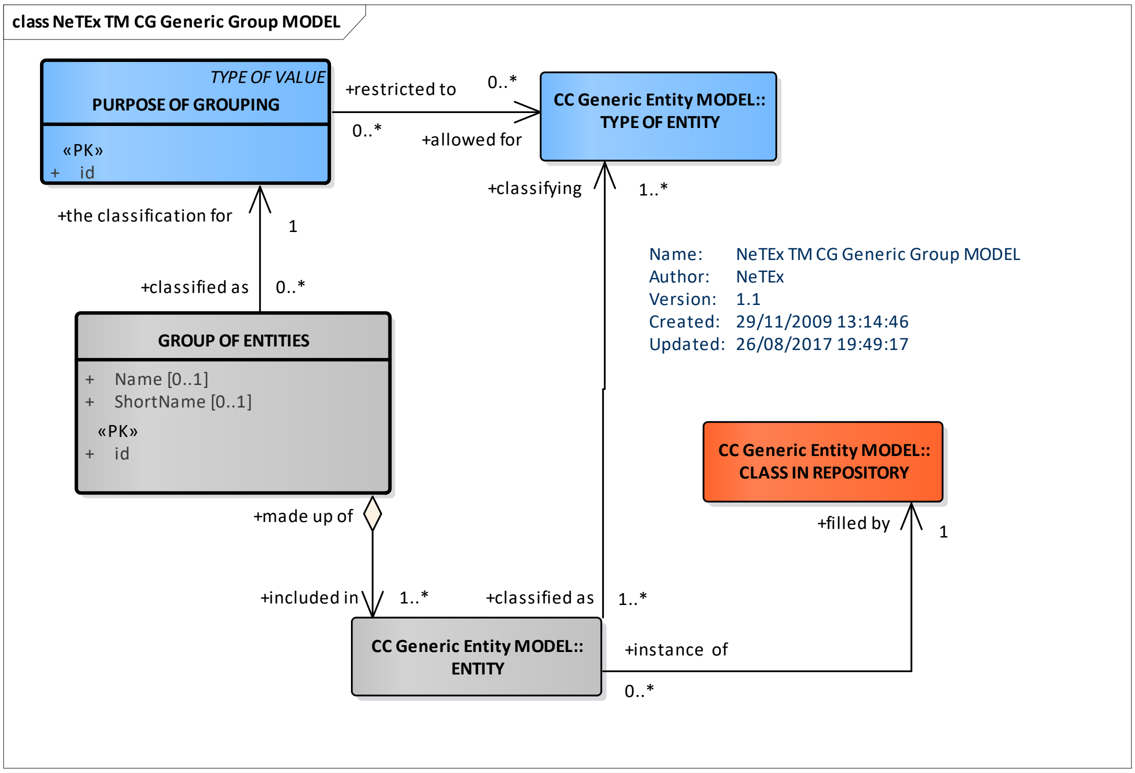


Figure 137 — Generic Grouping – Conceptual MODEL (UML)

7.6.4.2.1 Classifying Groups

The PURPOSE of GROUPING can be used to explain the purpose of arbitrary groupings of elements and to specify the allowed members.

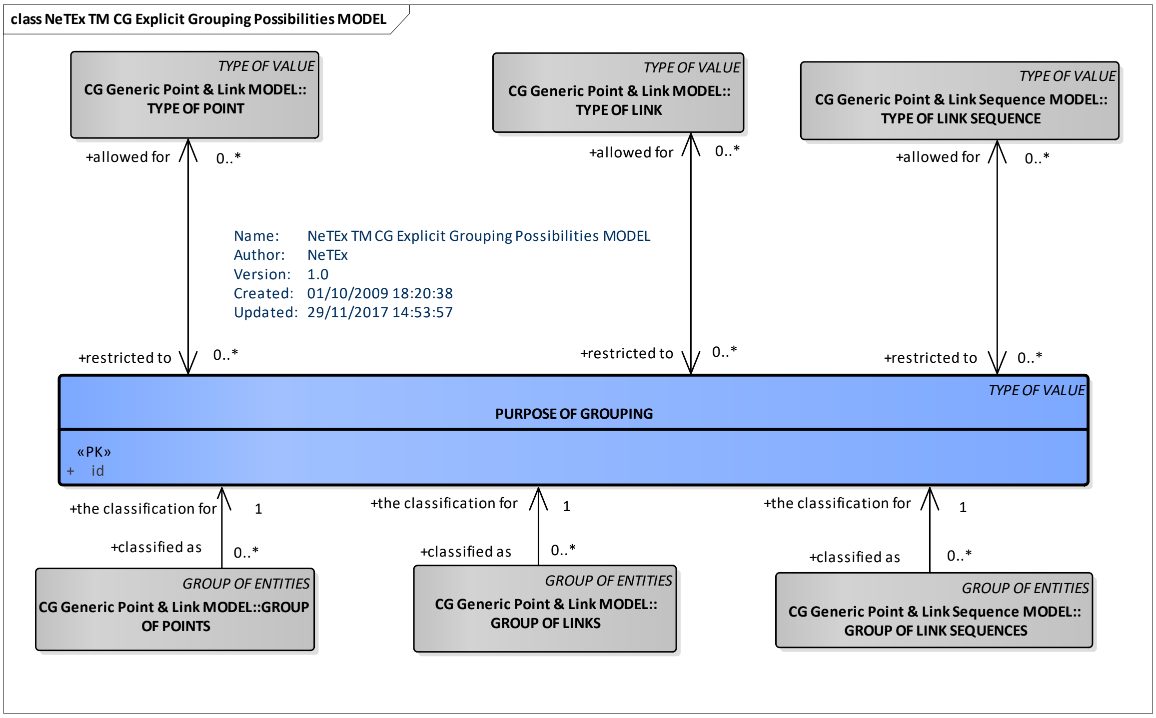


Figure 138 — Classifying Groups – Conceptual MODEL (UML)

7.6.4.3 Grouping – Physical Model

The element ***GroupOfEntities*** which implements GROUP of ENTITies in the Physical model provides common attributes such as Name that are present on all groups. ***GroupOfEntities*** is abstract (as it is shared with other group classes with different types of member), but a concrete specialisation, ***GeneralGroupOfEntities,*** can contain entities of any type.

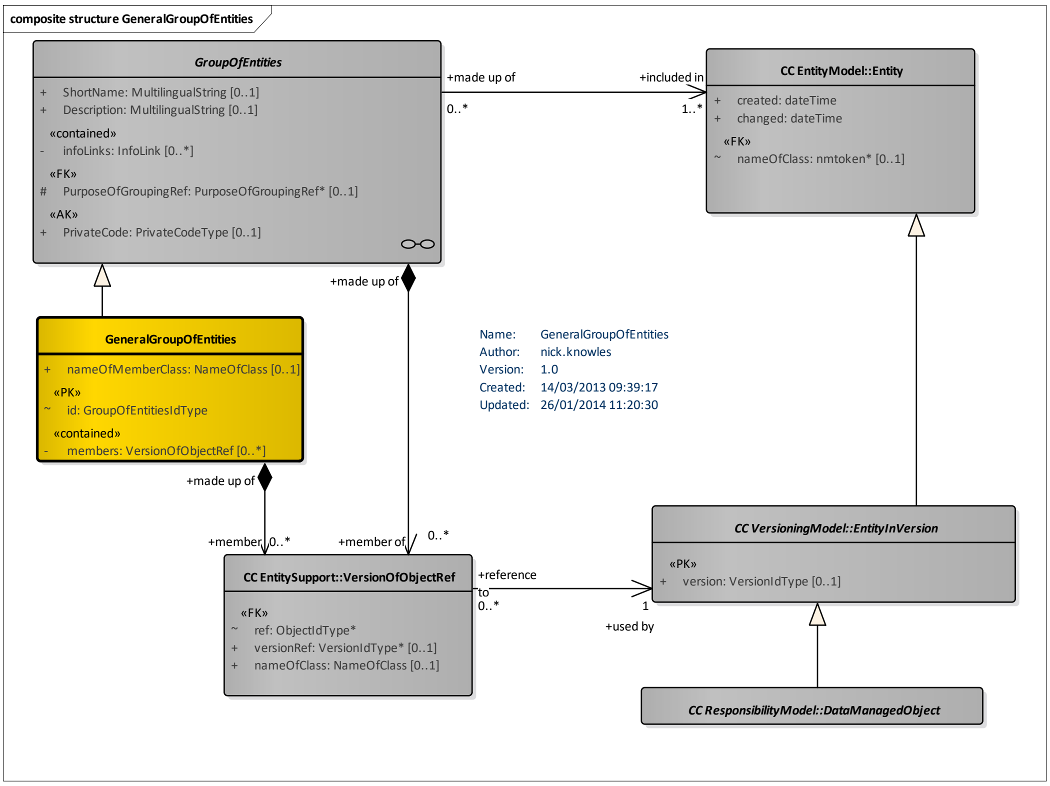


Figure 139 — Generic Grouping – Physical Model (UML)

7.6.4.3.1 Classifying Groups – Physical Model

It is best practice always to use the type of generic grouping element that is most specific to the element being grouped, as this allows for type checking. For example, GROUP OF LINEs should be used to group a collection of LINEs and not GENERAL GROUP OF ENTITIes.

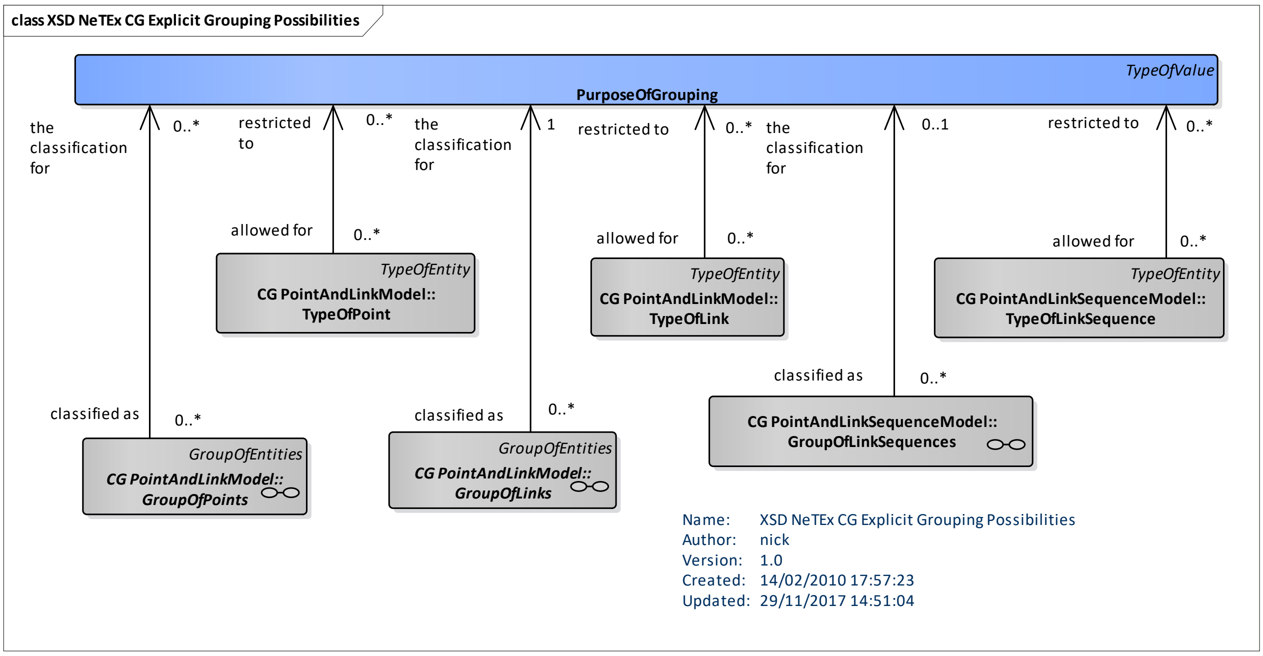


Figure 140 — Explicit Grouping – Physical Model (UML)

7.6.4.3.2 Explicit Specialisations of Group Of Entities – Physical Model

The following figure shows the explicit specialisations of GROUP OF ENTITIes available in NeTEx.

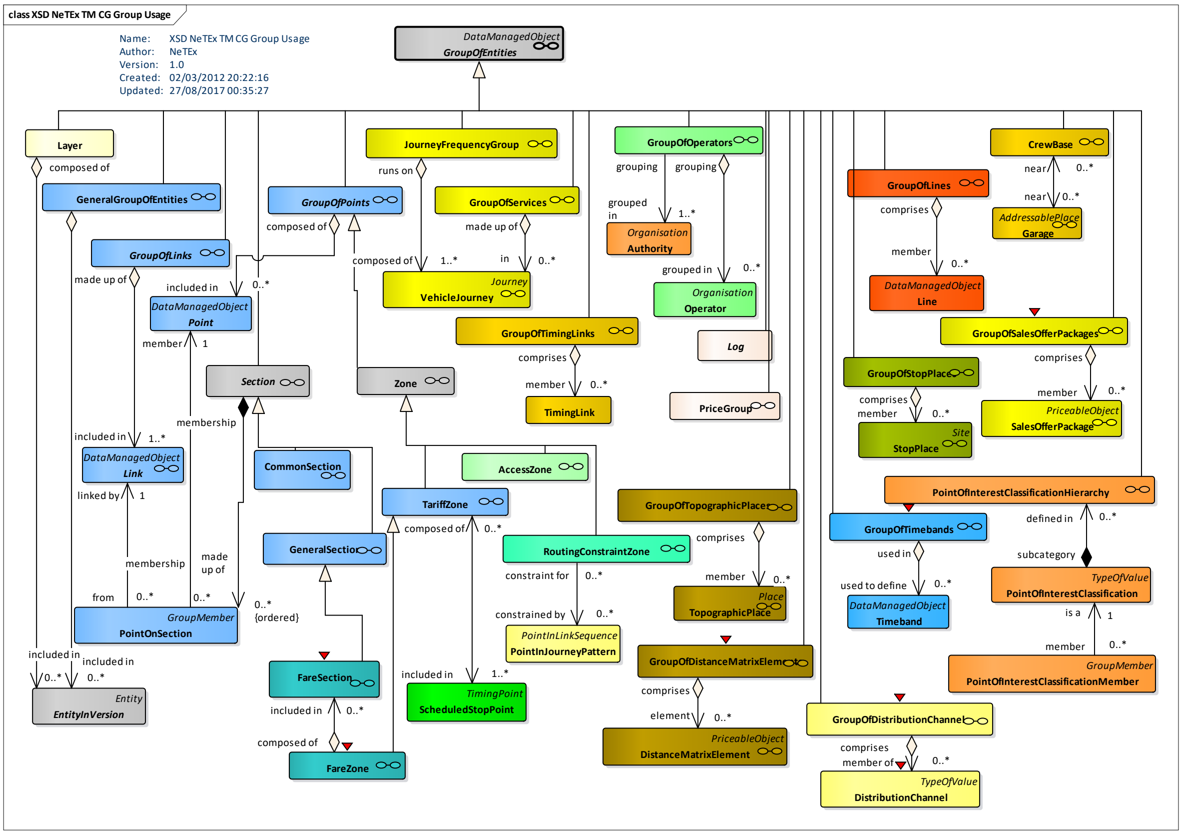


Figure 141 — Explicit Groups Usage – Physical Model (UML)

7.6.4.4 Grouping– Attributes and XSD

7.6.4.4.1 GroupOfEntities – Model Element

A set of ENTITies grouped together according to a PURPOSE OF GROUPING, e.g. grouping of stops known to the public by a common name.

Table 95 — *GroupOfEntities – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	GROUP OF ENTITies inherits from DATA MANAGED OBJECT.
«PK»	***id***	*GroupOfEntitiesIdType*	1:1	Identifier of GROUP OF ENTITies.
	Name	*MultilingualString*	0:1	Name of GROUP OF ENTITies.
	ShortName	*MultilingualString*	1:1	Short Name of GROUP OF ENTITies.
	Description	*MultilingualString*	0:1	Description of GROUP OF ENTITies.
«FK»	***PurposeOfGrouping-Ref***	*PurposeOfGroupingRef*	0:1	Reference to the PURPOSE OF GROUPING of GROUP OF ENTITies.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Private code associated of GROUP OF ENTITies.

«cntd»	***infoLinks***	*InfoLink*	0:*	InfoLinks associated with LINK SEQUENCE. +v1.1
«ctd»	***(members)***	*(VersionOfObjectRef) | GroupMember*	1:1	Reference to ENTITY included in group. Each concrete specialisation must add its own set restricted to the correct type.

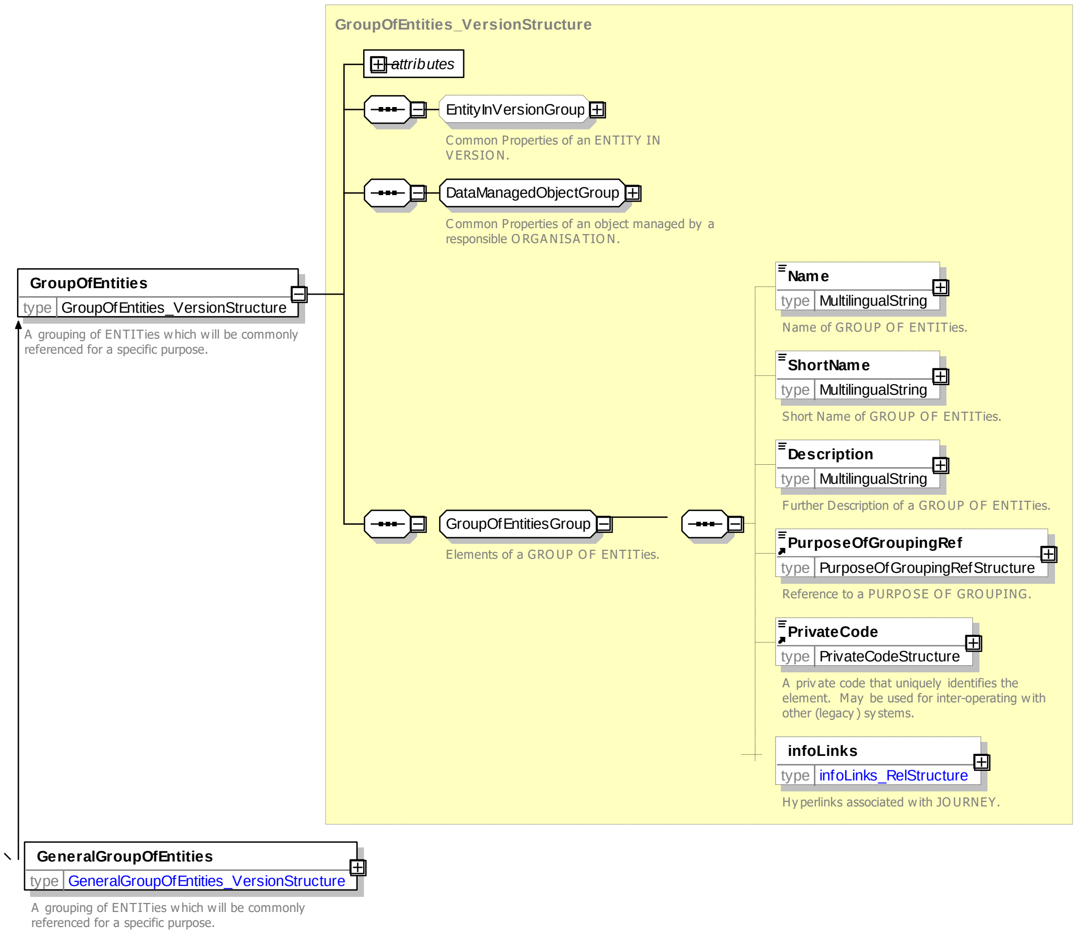


Figure 142 — *GroupOfEntities – XSD*

7.6.4.4.2 PurposeOfGrouping – Model Element

Functional purpose for which GROUPs of elements are defined. The PURPOSE OF GROUPING may be restricted to one or more types of the given object.

Table 96 — *PurposeOfGrouping – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	PURPOSE OF GROUPING inherits from TYPE OF VALUE.
«PK»	***id***	*PurposeOfGroupingIdType*	1:1	Identifier of PURPOSE OF GROUPING.

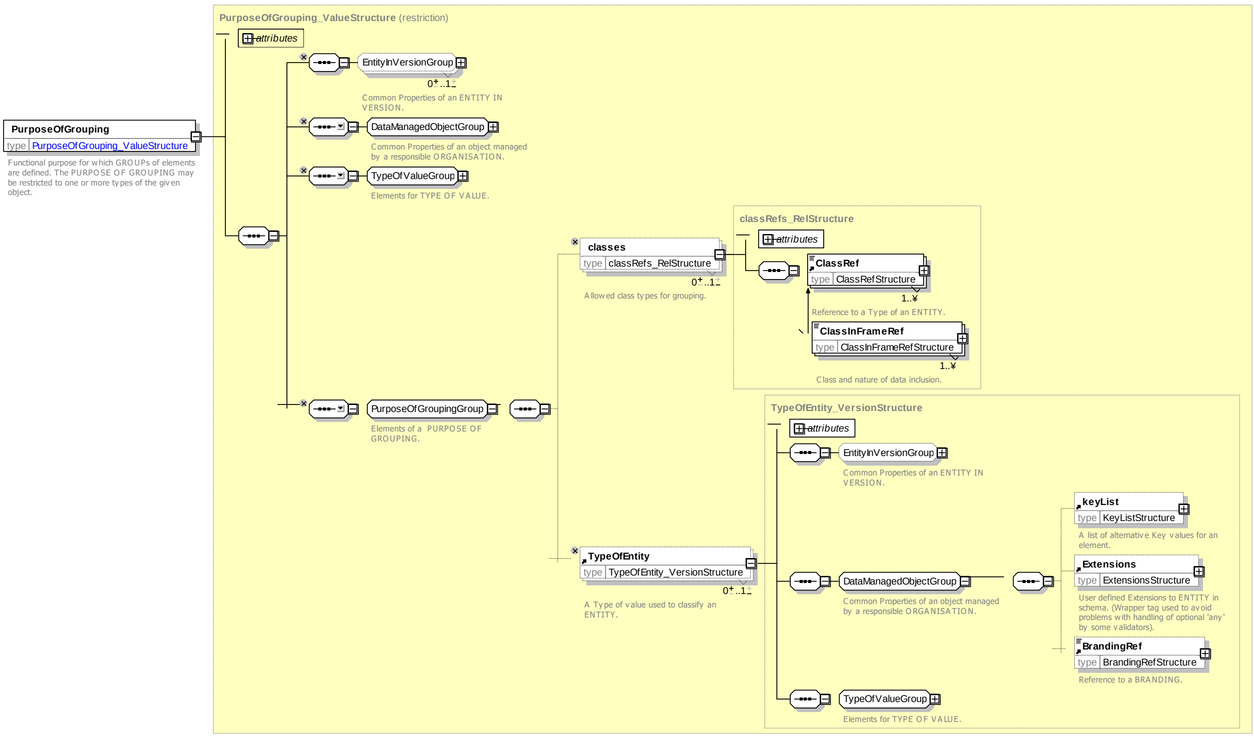


Figure 143 — *PurposeOfGrouping – XSD*

7.6.4.4.3 GeneralGroupOfEntities – Model Element

Represents an arbitrary grouping of entities of any type.

GeneralGroupOfEntities is concrete and may mix up entities of different types

Table 97 — *GeneralGroupOfEntities – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GENERAL GROUP OF ENTITies inherits from GROUP OF ENTITies.
«cntd»	***members***	*(VersionOfObjectRef)*	0:*	References to members of GROUP OF ENTITies.

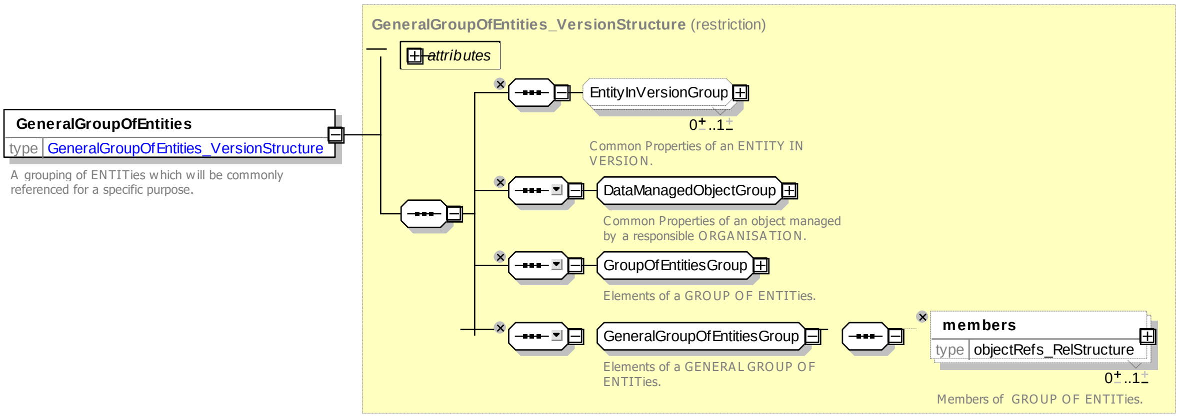


Figure 144 — *GeneralGroupOfEntities – XSD*

7.6.4.4.4 GroupMember – Model Element

An association that records membership of a group by an entity. This is for general purpose use.

Table 98 — *GroupMember – Element*

Classification	Name	Type	Cardinality	Description
«PK»	***id***	*GroupMemberIdType*	0:1	Name of GROUP MEMBER.
«FK»	***ParentRef***	*(GroupObjectRef)*	0:1	Reference to parent GROUP OF ENTITies.
«FK»	***MemberRef***	Member*ObjectRef*	1:1	Reference to an ENTITY that is a member of GROUP.
«FK»	***MemberVersionRef***	*VersionRef*	0:1	Reference to VERSION of referenced member.

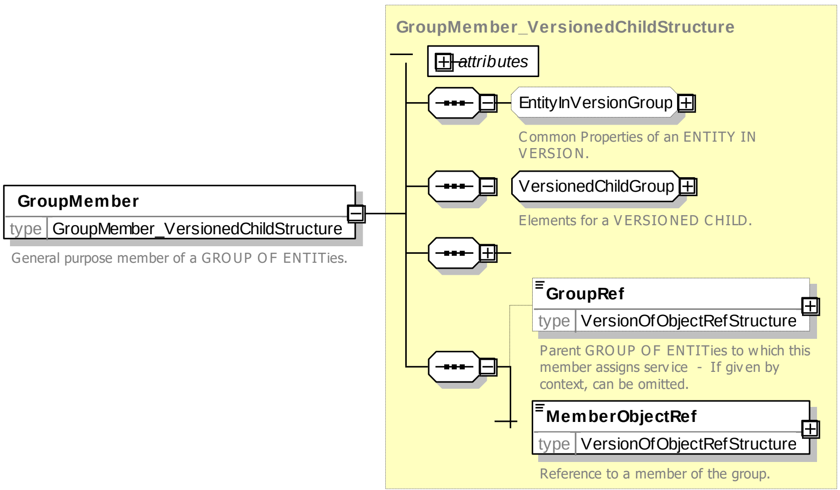


Figure 145 — *GroupMember – XSD*

7.6.4.5 *XML Examples of Grouping*

7.6.4.5.1 General Grouping – XML Example

The following example shows the use of a GENERAL GROUP OF ENTITies to define some arbitrary code values. A PURPOSE OF GROUPING is defined to specify the reason for the grouping.

EXAMPLE

```
        <PurposeOfGrouping version="any" id="mybus:Pog1">
                <Name>My Group of Scheduled Stop Points and Links</Name>
                <classes>
        <ClassRef nameOfClass="ScheduledStopPoint"/>
        <ClassRef nameOfClass="ServiceLink"/>
                </classes>
            </PurposeOfGrouping>
::>…
    <GeneralGroupOfEntities version="001" id="mybus:GGE001">
        <Name>My Group</Name>
        <PurposeOfGroupingRef version="any" ref="mybus:Pog1"/>
        <members>
            <ScheduledStopPointRef ref="mybus:SP0001A"/>
            <ScheduledStopPointRef ref="mybus:SSP0002B"/>
            <ScheduledStopPointRef ref="mybus:SSP0003CA"/>
            <ServiceLinkRef ref="mybus:SL_AtoB01"/>
        </members>
    </GeneralGroupOfEntities>
```

7.6.4.5.2 Explicit Grouping – XML Example

The following fragment shows the use of a GROUP OF LINEs to group the London underground subsurface lines:

EXAMPLE

```
    <lines>
        <Line version="002" created="2011-01-17T09:30:47.0Z" id="lul:CIR">
            <Name>Circle Line</Name>
            <TransportMode>metro</TransportMode>
            <Presentation><ColourName>yellow</ColourName></Presentation>
            </Line>
        <Line version="002" created="2011-01-17T09:30:47.0Z" id="lul:DIS">
            <Name>District Line</Name>
            <TransportMode>metro</TransportMode>
            <Presentation><ColourName>green</ColourName></Presentation>
        </Line>
        <Line version="002" created="2011-01-17T09:30:47.0Z" id="lul:HAM">
            <Name>Hammersmith &amp; City Line</Name>
            <TransportMode>metro</TransportMode>
            <Presentation><ColourName>pink</ColourName></Presentation>
        </Line>
        <Line version="002" created="2011-01-17T09:30:47.0Z" id="lul:MET">
            <Name>Metropolitan Line</Name>
            <TransportMode>metro</TransportMode>
            <Presentation><ColourName>maroon</ColourName></Presentation>
        </Line>
    </lines>
::>.
    <groupsOfLines>
        <GroupOfLines version="any" id="lul:GroupOfLines:lngp01">
            <Name>Subsurface lines</Name>
            <members>
                <LineRef version="002" ref="lul:CIR"/>
                <LineRef version="002" ref="lul:DIS"/>
                <LineRef version="002" ref="lul:HAM"/>
                <LineRef version="002" ref="lul:MET"/>
            </members>
        </GroupOfLines>
    </groupsOfLines>
```

7.6.4.5.3 Explicit Grouping Stop Area– XML Example

The following fragment shows the use of three STOP AREAs to group the SCHEDULED STOP POINTs of a Station.

EXAMPLE

```
<stopAreas>
    <StopArea version="any" id="napt:9100WIMBLDN ">
        <Name>Wimbledon Rail Area for MDV</Name>
        <members>
            <ScheduledStopPointRef version="any" ref="napt:9100WIMBLDN10"/>
        </members>
    </StopArea>
    <StopArea version="any" id="napt:940GZZCRWIM">
        <Name>Wimbledon Tube</Name>
        <members>
            <ScheduledStopPointRef version="any" ref="napt:9400ZZLUWIM"/>
        </members>
        <ParentStopAreaRef version="any" ref="napt:9100WIMBLDN "/>
    </StopArea>
    <StopArea version="any" id="napt:490G00272P">
        <Name>Wimbledon Bus</Name>
        <members>
            <ScheduledStopPointRef version="any" ref="napt:490000272P"/>
            <ScheduledStopPointRef version="any" ref="napt:490000272C"/>
            <ScheduledStopPointRef version="any" ref="napt:490000272D"/>
            <ScheduledStopPointRef version="any" ref="napt:490000272L"/>
        </members>
        <ParentStopAreaRef version="any" ref="napt:9100WIMBLDN-MDV02"/>
    </StopArea>
```

7.6.5 Generic Point & Link

7.6.5.1 Generic POINT & LINK – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

One of the most important aspects of information systems dealing with public transport is the representation of the networks over which the services are operated. Such a representation describes the objects composing a network (e.g. stations, lines, etc.) using simplified and conventional topological objects: points, links and for some purposes, zones. Specific roles are assigned to these simple objects, according to the functional purpose of the description.

The representation is chosen to be independent of the underlying geospatial context in which the network resides, but may be projected onto it or other spatial contexts using a projection model – see later.

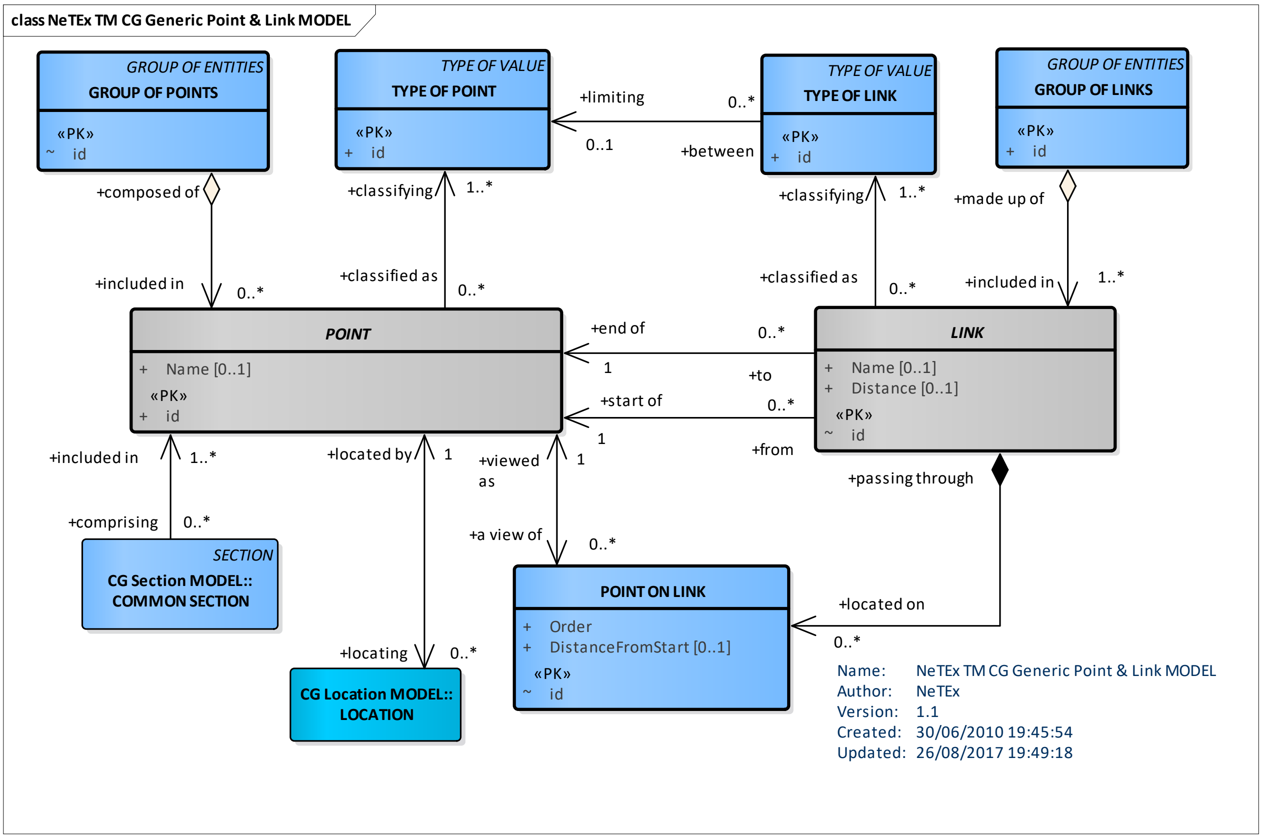


Figure 146 — Generic Point & Link – Conceptual MODEL (UML)

7.6.5.1.1 Point – ENTITY

Topological descriptions of the spatial structure of a public transport network are generally built with points. Thus, an entity POINT is defined as the most basic entity of the network model. A POINT represents a 0-dimension node of the network. It marks the location of bus stops, parking places or other types of POINTs.

A TYPE OF POINT is defined as an entity to describe common roles played by a number of POINTs. Each POINT is functionally classified as being of one or more types, according to the specific information needs of a particular functional domain.

Certain TYPEs of POINT are regarded as important enough to be additionally represented by a separate entity. The most important of these are the SCHEDULED STOP POINT, TIMING POINT and ROUTE POINT entities, described in a further section. Other examples are ROAD JUNCTION, ACTIVATION POINT, etc. The types if not explicitly defined by an entity of the reference model may be specified by the generic entity TYPE OF POINT. Any POINT may be of more than one type. For example, the same POINT may be a ROUTE POINT, a TIMING POINT and may be an ACTIVATION POINT as well.

7.6.5.1.2 Link – ENTITY

Between two POINTs of any type, a LINK may be defined to store spatial information (e.g. the distance a vehicle will cover crossing this link). LINKs represent 1-dimensional connections between POINTs. There must be no LINKs without one limiting POINT at each end. Two relationships between the POINT and the LINK entity specify the limiting POINTs of a LINK.

A LINK is **oriented** from its start POINT to its end POINT. This order has to be interpreted in a rather abstract sense of an artificial orientation, to be possibly used for expressions like “left of” or “right of” the LINK. The order does not necessarily express the direction of the traffic flow, for instance, which must be defined by appropriate entities, relationships or attributes, depending on the functional context. LINKs are usually not used as standalone objects, but through specialised object (by inheritance): if the orientation of the LINK has a specific meaning, or if the LINK is bidirectional, the specialised object will carry specific attribute to express it.

The network structures used by different functions may be subject to different conditions and constraints. In some structures, the ordered connection between two POINTs may have to be unique. This means that there cannot be more than one LINK between the two same end POINTs. In other words, such a link is logically a straight line (if it has a curvilinear shape, it is only for a drawing purpose). In other structures, there may be two or more different connections between the same start and end POINTs. This means that such alternative LINKs follow different paths between the two POINTs, therefore have different shapes. In this case, the shape is implicitly associated to a LINK. For instance, between two TIMING POINTs, there may be different TIMING LINKs if the vehicles covering these links follow different paths (ROUTEs).

The LINK entity is therefore identified by an own ID attribute, which allows multiple LINKs between the same pair of POINTs (e.g. TIMING POINTs). This ID does not represent explicitly the path followed by the LINK. The projection mechanism allows to indicate an exact geospatial path. For applications in which a LINK must be identified by its limiting POINTs, these may be used as an alternative unique key, or the artificial ID may be implemented as a combination of the two end point identifiers.

The entity TYPE OF LINK expresses the different functional roles of a LINK. For instance, this classification may include a distinction between “commercial links” to be used for passenger carrying journeys and “connecting links” to be used by dead runs or turnarounds at the terminals. It may be useful to express a difference between LINKs with separate bus lanes and without, or to describe activation specifications (to control announcements, ticketing devices, etc.) that are identical for each LINK of a given TYPE OF LINK.

Each LINK is functionally classified as being of one or more types, according to the specific information needs of a particular functional domain. As for the TYPE OF POINT, certain TYPEs OF LINK are explicitly defined by an entity in the reference model (e.g. SERVICE LINK, TIMING LINK, ROUTE LINK).

In most cases, LINKs of a given type must be only between POINTs of a corresponding TYPE OF POINT. For instance, SERVICE LINKs must only be used to connect STOP POINTs (however, other TYPEs OF LINK may be defined between STOP POINTs). An optional relationship between TYPE OF LINK and TYPE OF POINT expresses that only points of the specified type must be used as limits for links of a given type (or several types).

7.6.5.1.3 Point on Link – ENTITY

It is often necessary to define POINTs that are simply located on a LINK of a certain type. For instance, on a LINK defined for activation of traffic light priority, some intermediate points may be necessary, at which a vehicle should confirm or cancel the priority request. If a platform is described by a LINK, it may be necessary to define the different coach stopping positions as POINTs on this LINK. Such a POINT ON LINK is a POINT that is defined on a LINK belonging to the same layer.

Each POINT ON LINK is identified by the LINK it is located on and by an order on that LINK. The distance from the start point of the LINK is only optional information, but the order attribute ensures that all the intermediate POINTs ON a LINK are sequenced in a unique way.

7.6.5.1.4 Group of Points & Group of Links – ENTITies

The Conceptual MODEL schema also shows two explicit grouping mechanisms: GROUP OF POINTs and GROUP OF LINKs (already introduced in the section on Grouping Model).

A GROUP OF POINTs may be used to describe a central or complex station, consisting of all SCHEDULED STOP POINTs serving the whole area of this station, or any important interchange area. In such a case, the PURPOSE OF GROUPING of the GROUP OF POINTs will limit the grouped POINTs to SCHEDULED STOP POINTs (TYPE OF POINT). This allows one to use classical STOP AREAs to describe limited sets of STOP POINTs (e.g. a couple of bus SCHEDULED STOP POINTs close to the station) and to add one or two other hierarchical levels of STOP POINT groups (e.g. the upper group associating all SCHEDULED STOP POINTs of the station, whatever the transport mode serving them).

Passenger information functions, in particular information on interchanges may use such GROUPs OF POINTs in various ways.

A GROUP OF POINTS may also be used to describe an operational cluster, consisting of STOP POINTs, RAILWAY JUNCTION, ACTIVATION POINTs, and POINTs of any other type. All these POINTs are located close to each other and operationally belong to an object known by a particular name (e.g. train station, from the operational point of view).

A GROUP OF LINKs may be all LINKs in a tunnel, all LINKs in an urban area, etc.

7.6.5.1.5 Explicit Specialisations of Point

The following figure shows the explicit specialisations of POINT found in NeTEx.

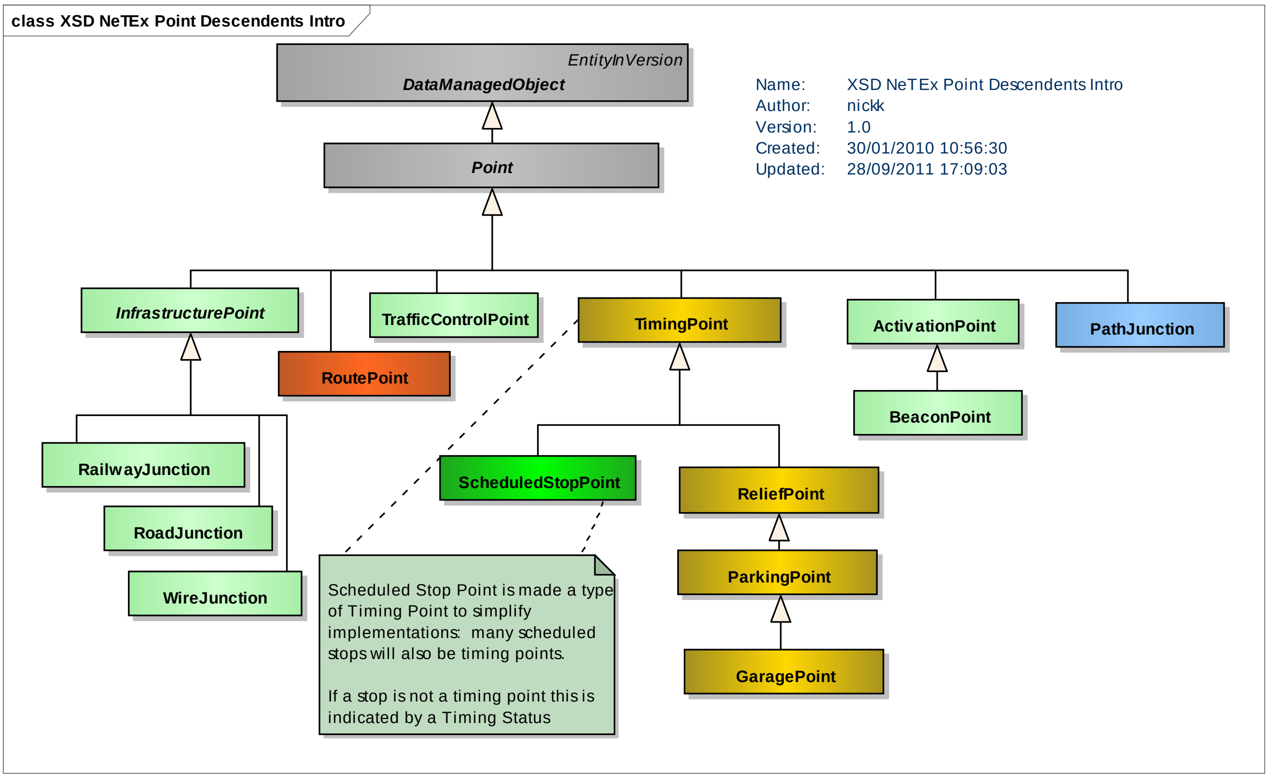


Figure 147 — Explicit Points Overview – Physical Model (UML)

7.6.5.1.6 Explicit Specialisations of Link

The following figure shows the explicit specialisations of LINK found in NeTEx.

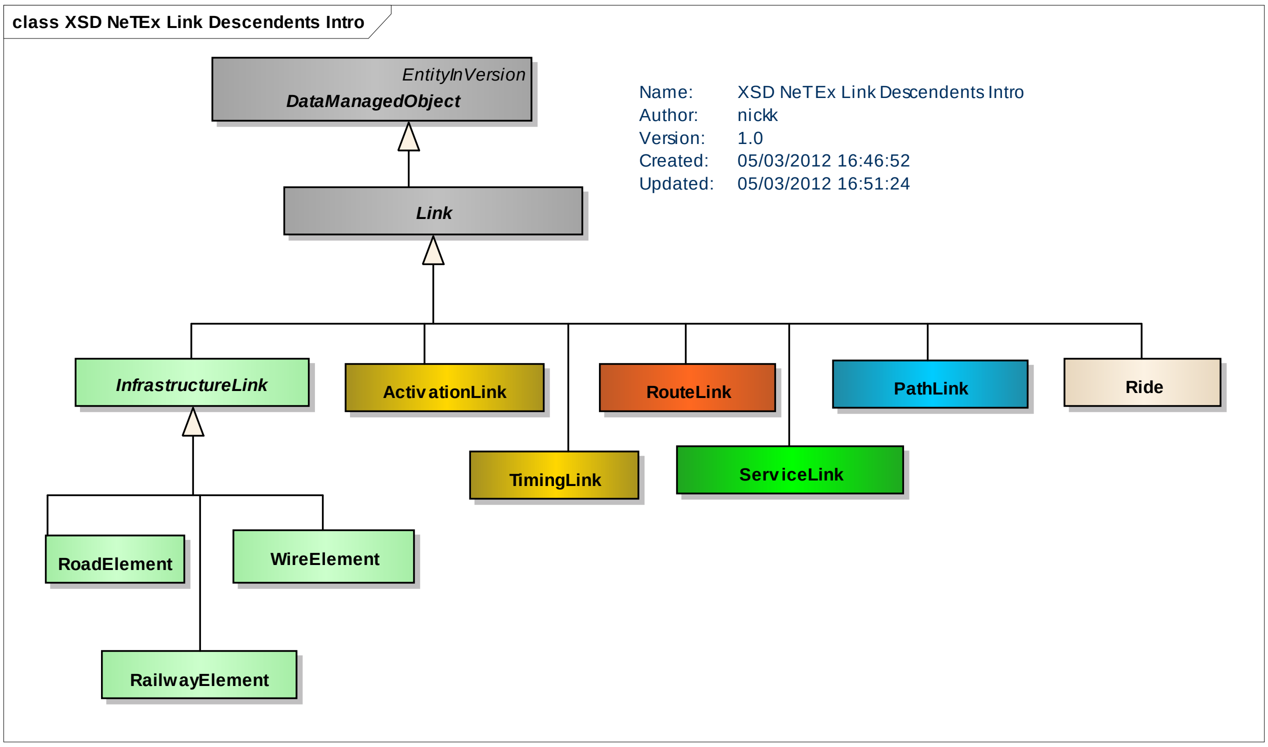


Figure 148 — Explicit Links Overview – Physical Model (UML)

7.6.5.2 Generic Point & Link – Physical Model

The Point & Link Physical model provides abstract classes which can be generalised to create concrete point and link types such as TIMING POINT, SCHEDULED STOP POINT, TIMING LINK, SERVICE LINK, etc.

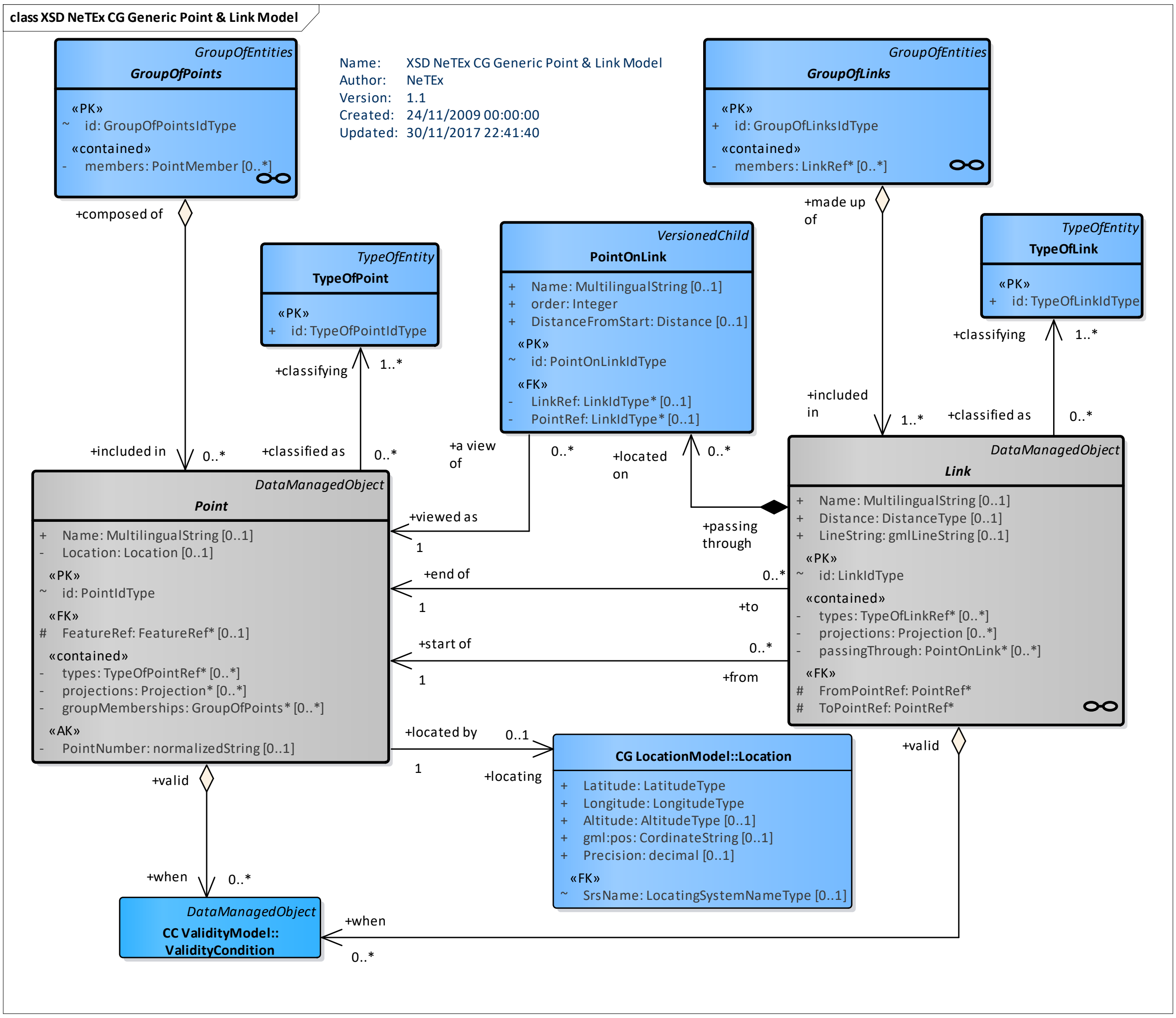


Figure 149 — Generic Point & Link – Physical Model (UML)

7.6.5.3 Generic Point & Link – Attributes and XSD

7.6.5.3.1 Point – Model Element

A 0-dimensional node of the network used for the spati*al description of the network. POINTs may be located by a LOCATION in a given LOCATING SYSTEM.*

Table 99 — *Point – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	POINT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*PointIdType*	1:1	Identifier of POINT.
	Name	*MultilingualString*	0:1	Name of POINT.

	Location	*Location*	0:1	Location of POINT.
	PointNumber	*xsd:normalizedString*	0:1	Arbitrary alternative identifier for the POINT.
«cntd»	***types***	*TypeOfPointRef*	0:*	Reference to one or more a TYPE OF POINT.
«cntd»	***projections***	*Projection*	0:*	Projections for POINT.
«cntd»	***group-Memberships***	*GroupOfPoints*	0:*	GROUPs of POINTs to which POINT belongs.

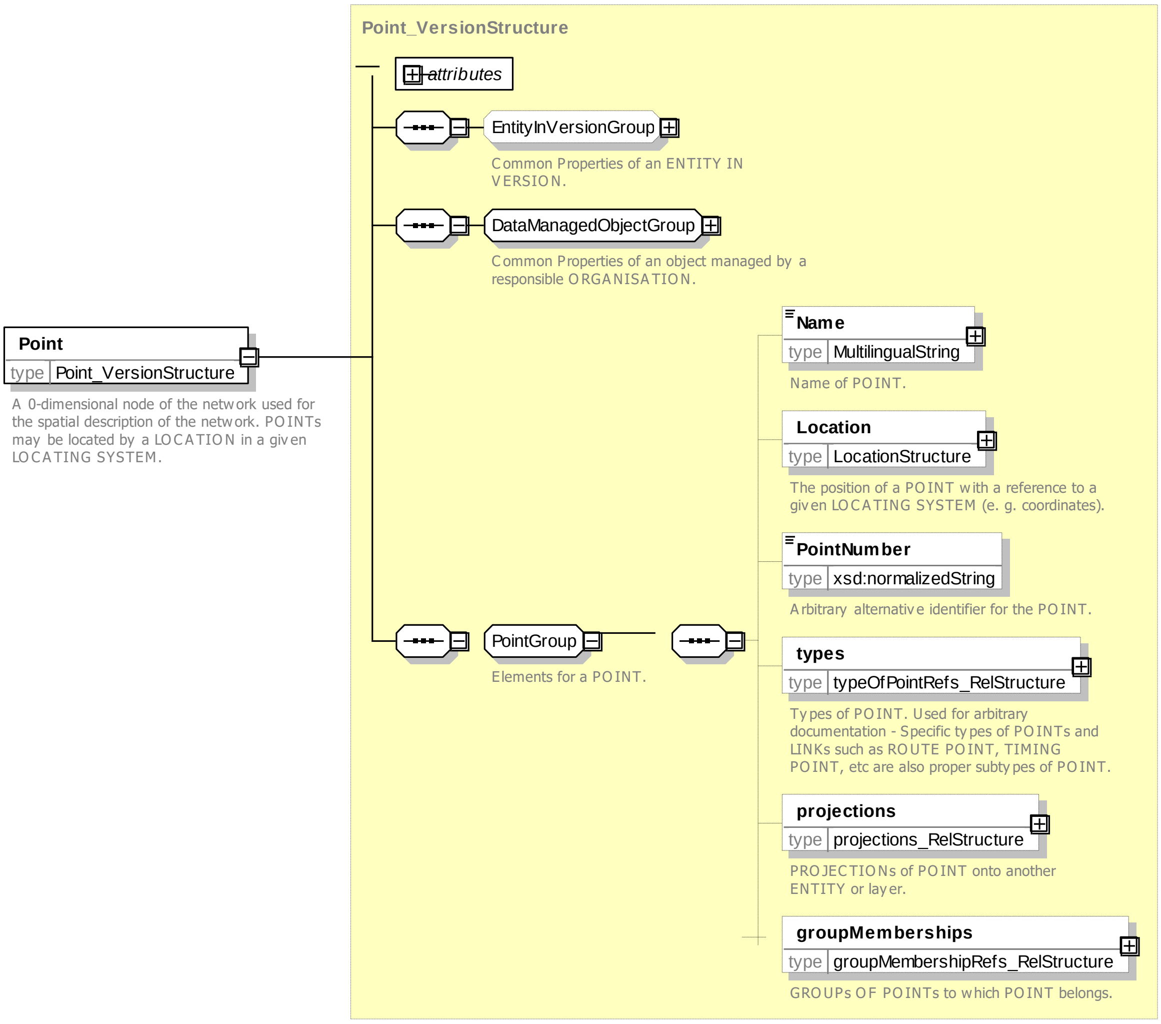


Figure 150 — *Point – XSD*

7.6.5.3.2 TypeOfPoint – Model Element

A classification of POINTs according to their functional purpose.

Table 100 — *TypeOfPoint – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	*TypeOfEntity*	::>	TYPE OF POINT inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfPointIdType*	1:1	Identifier of TYPE OF POINT.

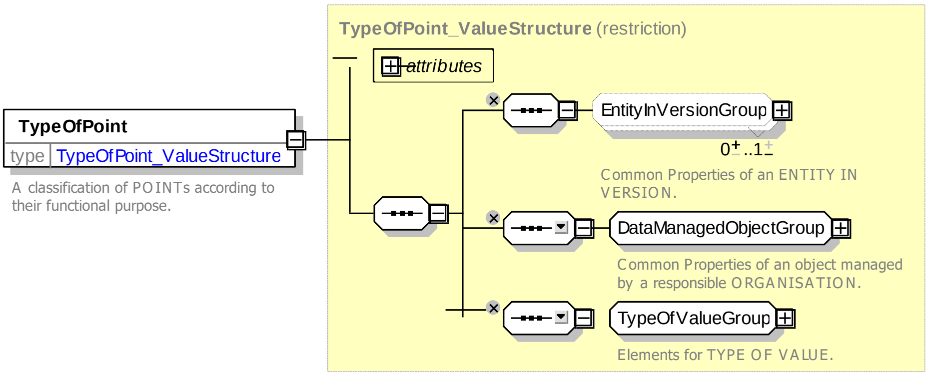


Figure 151 — *TypeOfPoint– XSD*

7.6.5.3.3 Link – Model Element

An oriented spatial object of dimension 1 with view to the overall description of a network, describing a connection between two POINTs.

Link is abstract – the ***FromPointRef*** & ***ToPointRef*** are added on the concrete subclasses.

Table 101 — *Link – Element*

Classification	**Name**	**Type**	**Cardinality**	**Description**
::>	::>	*DataManagedObject*	::>	LINK inherits from DATA MANAGED OBJECT.
«PK»	***id***	*LinkIdType*	1:1	Identifier of LINK.
	Name	*MultilingualString*	0:1	Name of LINK.
	Distance	*DistanceType*	0:1	Distance along LINK.
«cntd»	***types***	*TypeOfLinkRef*	0:*	Reference to one or more TYPEs OF LINK.
«cntd»	***LineString***	*gml:LineString*	0:1	GML linestring of the LINK.
«cntd»	***projections***	*(Projection)*	0:*	Projections for LINK.
«cntd»	***passingThrough***	*PointOnLink*	0:*	POINTs on LINK that LINK passes through.
«FK»	(***FromPointRef***)	*(PointRef)*	1:1	Reference to point at which LINK starts. Each Concrete subtype must add.
«FK»	(***ToPointRef***)	*(PointRef)*	1:1	Reference to point at which LINK ends. Each Concrete subtype must add.

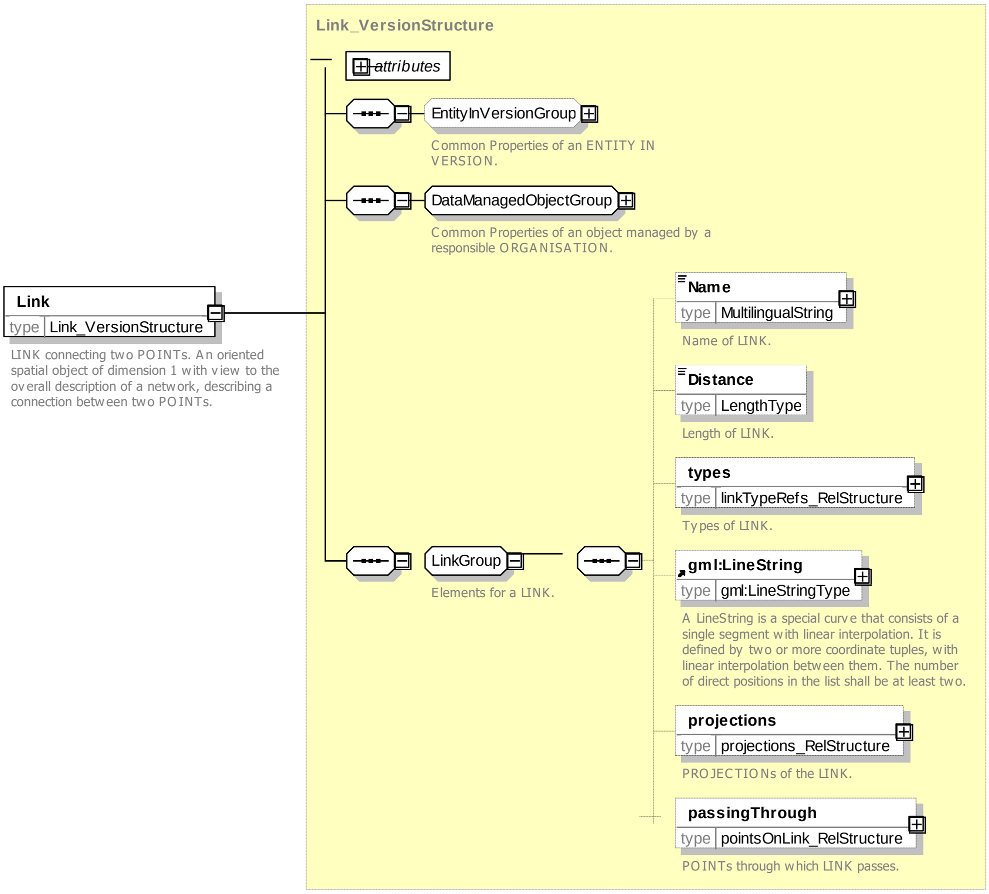


Figure 152 — *Link – XSD*

7.6.5.3.4 TypeOfLink – Model Element

A classification of LINKs to express the different functional roles of a LINK.

Table 102 — *TypeOfLink – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF LINK inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfLinkIdType*	1:1	Identifier of TYPE OF LINK.

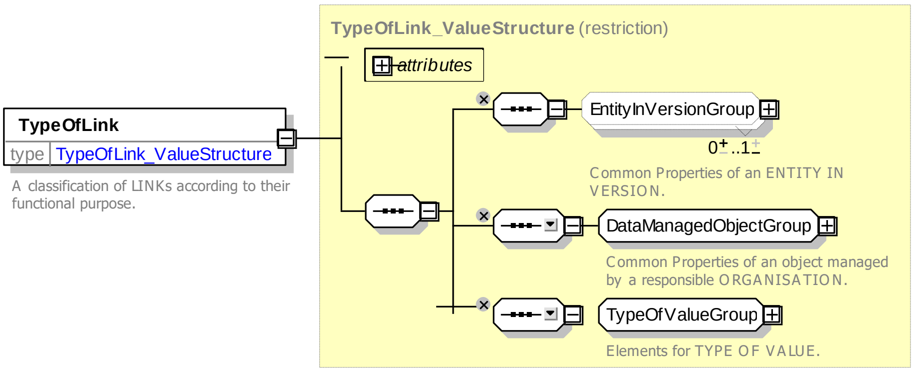


Figure 153 — *TypeOfLink – XSD*

7.6.5.3.5 PointOnLink – Model Element

A POINT on a LINK which is not needed for LINK definition, but may be used for other purposes, e.g. for purposes of AVM or passenger information, or for driver information.

Table 103 —*PointOnLink – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>		POINT ON LINK inherits from VERSIONED CHILD.
«PK»	***id***	*PointOnLinkIdType*	1:1	Identifier of POINT ON LINK.
«atr»	***order***	*Integer*	1:1	Order of POINT along link relative to other POINTs ON LINK.
	Name	*MultilingualString*	0:1	Name of POINT ON LINK.
«FK»	***LinkRef***	*(LinkRef)*	0:1	Reference to parent LINK on which POINT ON LINK is found. If given by context can be omitted.
	DistanceFromStart	*Distance*	1:1	Distance of POINT along LINK.
«FK»	a ***PointRef***	*(PointRef)*	1:1	Reference to POINT that is on LINK.
«ctnd»	b ***Point***	*(Point)*	1:1	POINT that is on LINK.

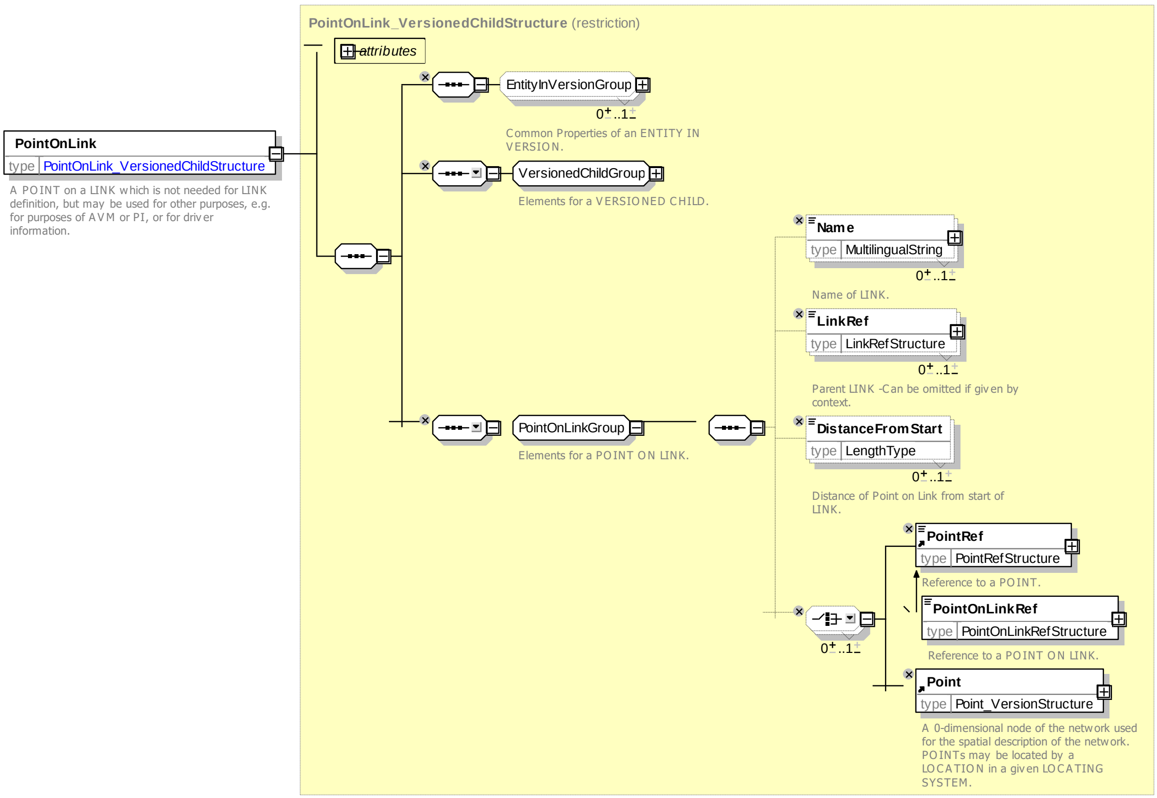


Figure 154 — *PointOnLink – XSD*

7.6.5.3.6 GroupOfPoints – Model Element

A grouping of POINTs.

A STOP AREA represents a specialisation of a GROUPs OF POINTs - a grouping of SCHEDULED STOP POINTs for specific TRANSPORT MODEs.

Table 104 — *GroupOfPoints – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF POINTs inherits from GROUP OF ENTITies.
«PK»	***id***	*GroupOfPointsIdType*	1:1	Identifier of GROUP OF POINTs.
«cntd»	***members***	*(PointRef)*	0:*	Member POINTs of GROUP OF POINTs.

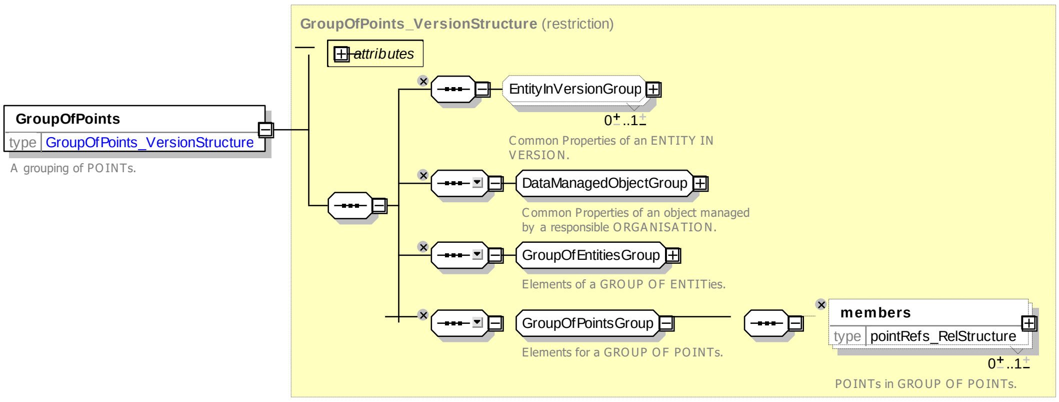


Figure 155 — *GroupOfPoints – XSD*

7.6.5.3.7 GroupOfLinks – Model Element

A grouping of LINKs. e.g. a specific GROUP OF LINKs might be managed by the same AUTHORITY.

Table 105 — *GroupOfLinks – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF LINKs inherits from GROUP OF ENTITies.
«PK»	***id***	*GroupOfLinksIdType*	1:1	Identifier of GROUP OF LINKs.
«cntd»	***members***	*(LinkRef)*	0:*	Member LINKs of GROUP OF LINKs.

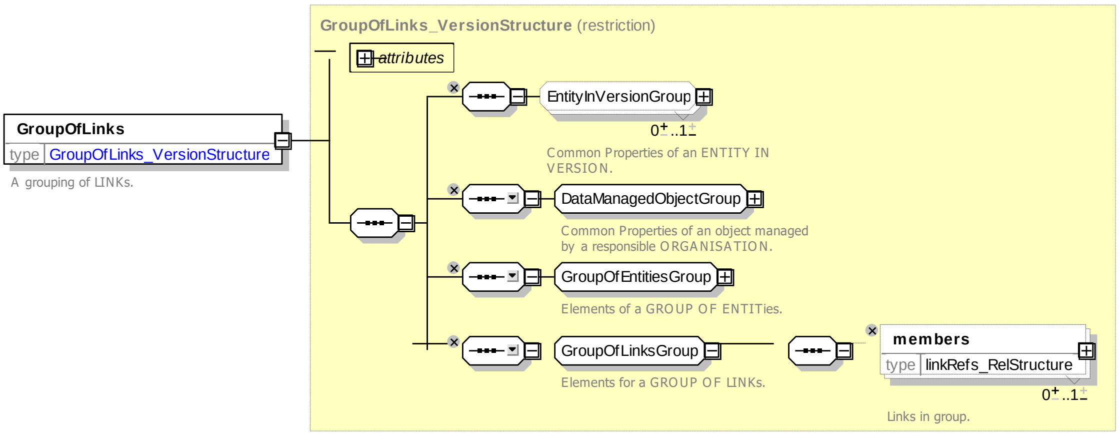


Figure 156 — *GroupOfLinks – XSD*

7.6.6 Generic Section

7.6.6.1 Generic SECTION – Conceptual MODEL

The generic SECTION describes an ordered sequence either of POINTs or of LINKs, defining a partial path through the network and having some common property. The link and point representations are equivalent as the links run between the points of the links.

Specialised examples include COMMON SECTION and LINE SECTION, and FARE SECTION. A GENERAL SECTION is a general-purpose instance of a SECTION that can be used for a sequence of links of any TYPE.

A LINK SEQUENCE can be considered to be an ordered sequence of SECTIONS; sections are thus sequences of links between intermediate points that may have additional meaning attached to them, as in the case of a COMMON SECTION, or a FARE SECTION. A SECTION IN LINK SEQUENCE represents an ordered instance of a SECTION within a LINK SEQUENCE

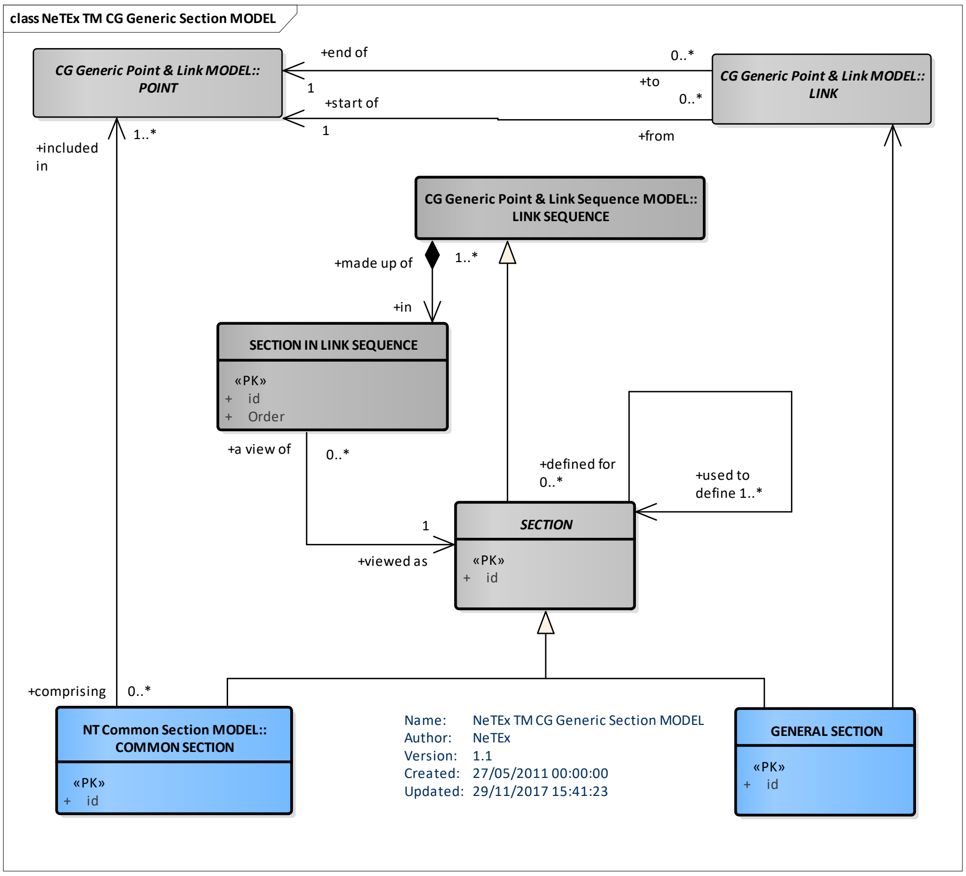


Figure 157 — General Section – Conceptual MODEL (UML)

7.6.6.2 Generic Section – Physical Model

The Point & Link Physical Model introduces two new objects for technical implementation purpose: ***LinkOnSection*** (a specialison of ***LinkInLinkSequence***) and ***PointOnSection*** (a specialisation of ***PointInLinkSequence***). These objects implement the many to many relationships between SECTION and POINT & SEQUENCE respectively.

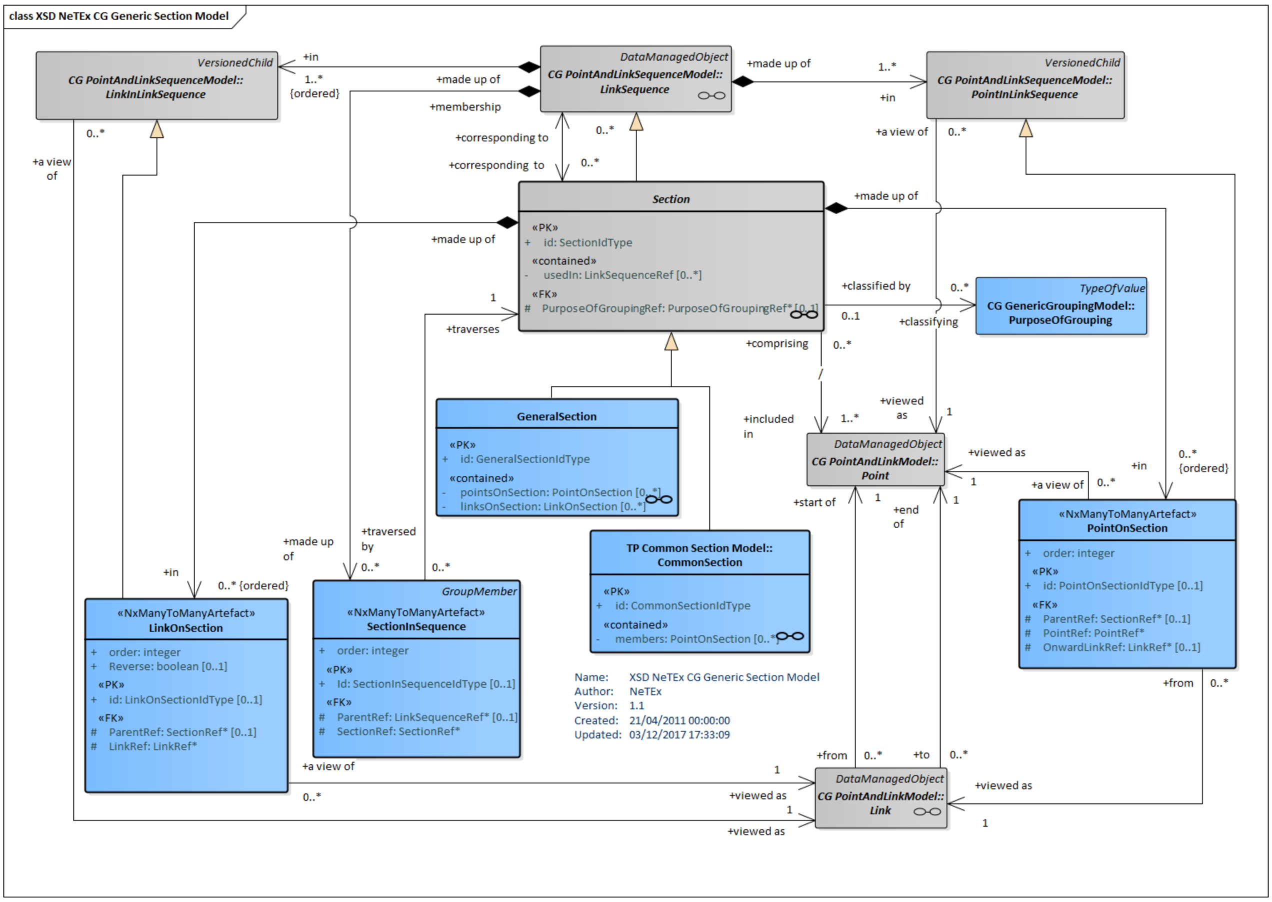


Figure 158 — General Section – Physical Model (UML)

7.6.6.3 Generic Section – Attributes and XSD

7.6.6.3.1 Section – Model Element

A part of a public transport network where the ROUTEs of several JOURNEY PATTERNs are going in parallel and where the synchronisation of SERVICE JOURNEYs may be planned and controlled with respect to commonly used LINKs and SCHEDULED STOP POINTs. COMMON SECTIONs are defined arbitrarily and need not cover the total lengths of topologically bundled sections.

Table 106 —*Section – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	SECTION inherits from LINK SEQUENCE.
	id	*SectionIdType*	1:1	Identifier of SECTION.
«FK»	***PurposeOf-Grouping***	*PurposeOfGroupingRef*	0:1	Classification of SECTION with PURPOSE OF GROUPING.
«cntd»	***usedIn***	*(LinkSequenceRef)*	0:*	LINK SEQUENCEs that use this SECTION.

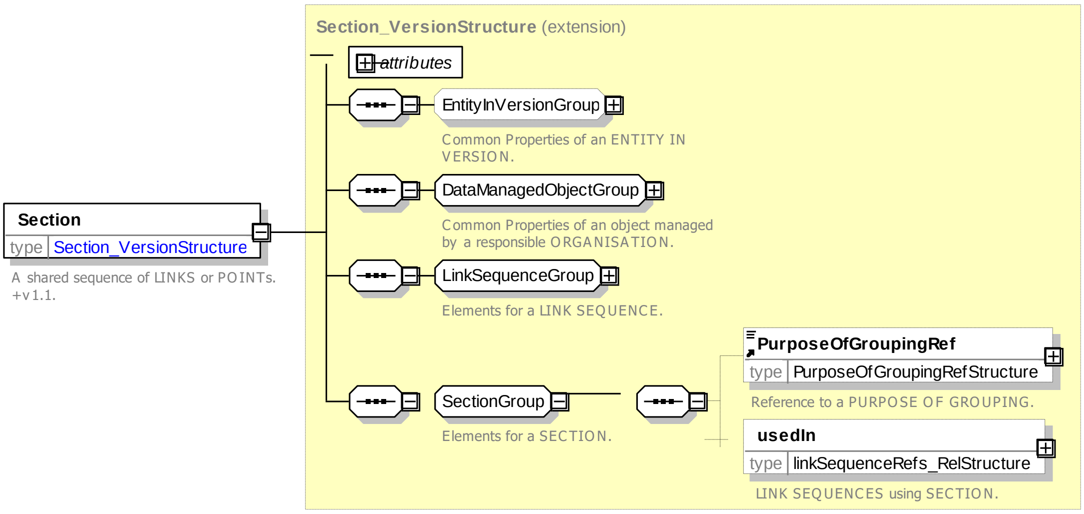


Figure 159 — *Section – XSD*

7.6.6.3.2 SectionInSequence– Model Element

A SECTION that is an ordered part of a LINK SEQUENCE - as specialisation of LINK IN LINK SEQUENCE.

Table 107 — SectionInSequence – *Element*

Classification	Name		Type	Cardinality	Description
::>	::>		*VersionedChild*	::>	SECTION IN SEQUENCE inherits from LINK IN LINK SEQUENCE.
«PK»	***id***		*SectionInSequenceIdType*	0:1	Identifier of SECTION IN SEQUENCE.
			CHOICE		Refernce or embedded
«FK»	a	***SectionRef***	*(LinkSequenceRef)*	0:1	Refernce to SECTION of SECTION IN SEQUENCE.
«ctnd»	b	***Section***	*(Section)*	0:1	SECTION of SECTION IN SEQUENCE

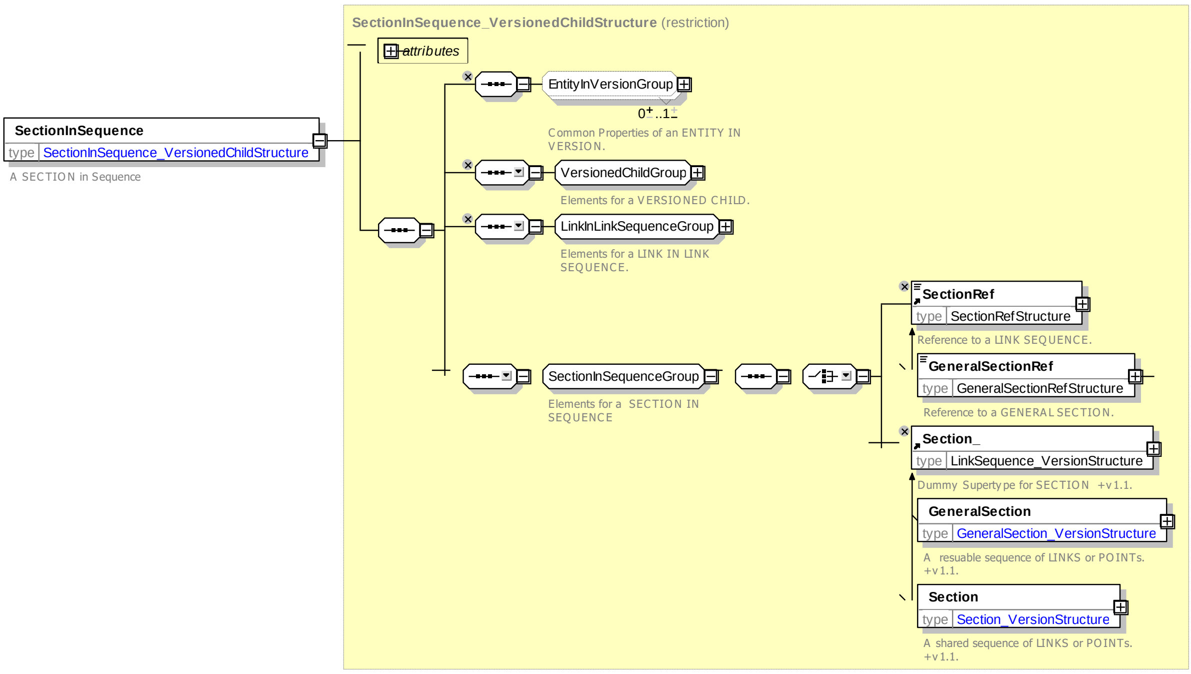


Figure 160 —*SectionInSequence – XSD*

7.6.6.3.3 PointOnSection – Model Element

POINT member of a COMMON SECTION.

Table 108 —*PointOnSection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*LinkInLinkSequence*	::>	POINT ON SECTION inherits from POINT IN LINK SEQUENCE.
«PK»	***id***	*PointOnSectionIdType*	0:1	Identifier of POINT ON SECTION.
«FK»	***PointRef***	*(PointRef)*	1:1	Reference to a POINT that belongs to POINT ON SECTION.
«FK»	***LinkRef***	*(LinkRef)*	1:1	Reference to a LINK that belongs to POINT ON SECTION.
	reverse	*xsd:boolean*	0:1	Whether navigation of LINK is considered to be in reverse.

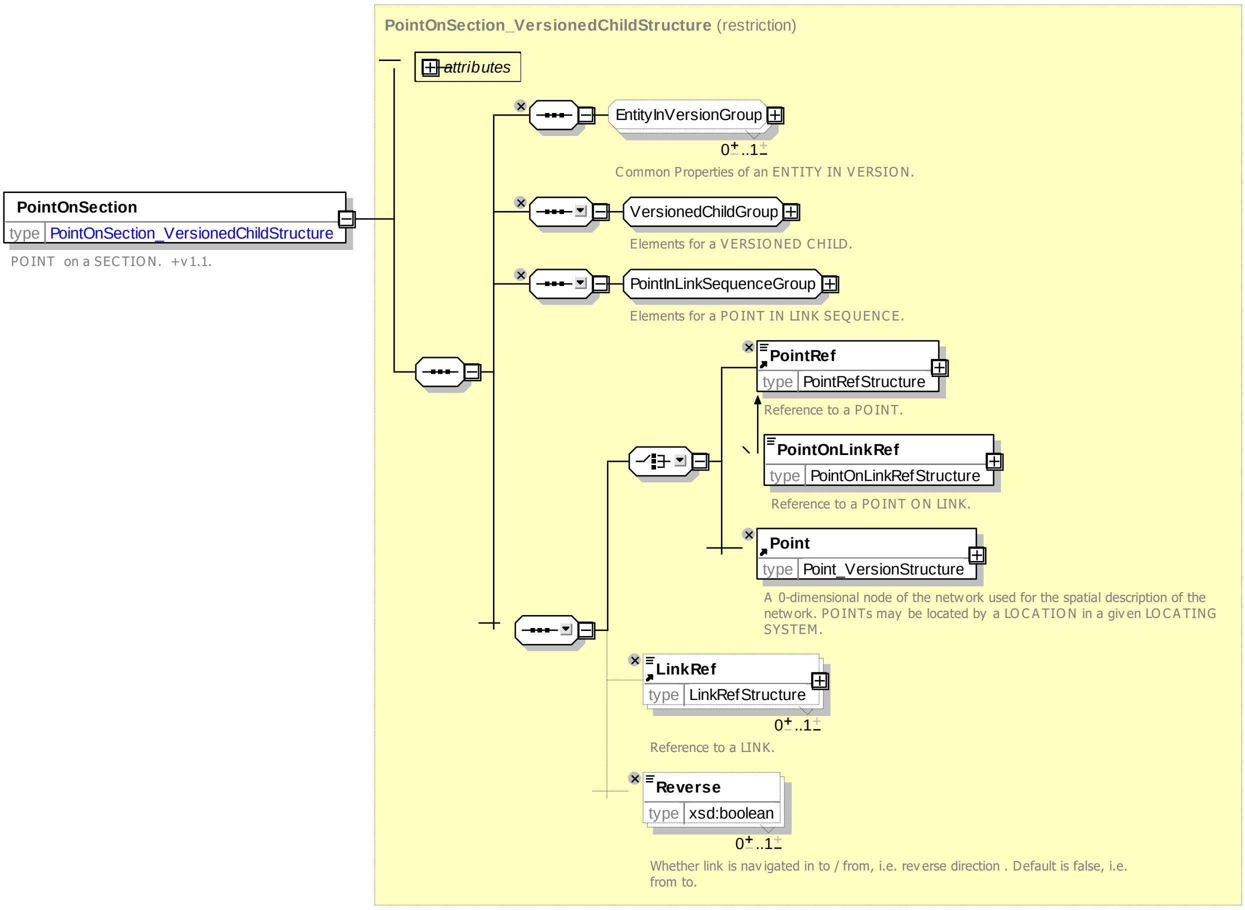


Figure 161 —*PointOnSection – XSD*

7.6.6.3.4 LinkOnSection – Model Element

LINK member of a COMMON SECTION.

Table 109 —*LinkOnSection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*LinkInLinkSequence*	::>	POINT ON SECTION inherits from LINK IN LINK SEQUENCE.
«PK»	***id***	*LinktOnSectionIdType*	0:1	Identifier of LINK ON SECTION.
«FK»	***LinkRef***	*(LinkRef)*	1:1	Reference to a LINK that belongs to LINK ON SECTION.
	reverse	*xsd:boolean*	0:1	Whether navigation of LINK is considered to be in reverse.

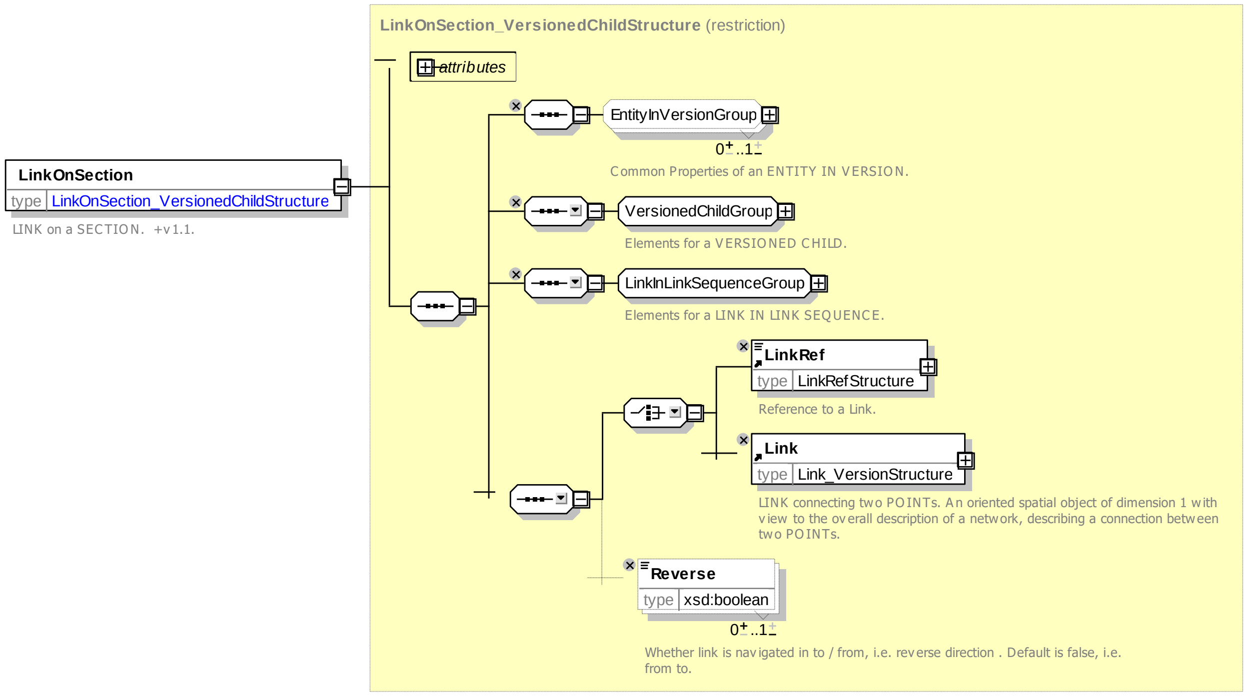


Figure 162 —*LinkOnSectionMember – XSD*

7.6.6.3.5 GeneralSection– Model Element

GENERAL SECTION is a general-purpose instance of a SECTION that can be used for a sequence of links of any TYPE.

Table 110 — GeneralSection – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Section*</u>	::>	GENERAL SECTION inherits from SECTION.
«PK»	***id***	*GeneralSectionIdType*	0:1	Identifier of GENERAL SECTION.
«cntd»	***pointsOnSection***	<u>*PointOnSection*</u>	0:*	POINTs on SECTION that make up this SECTION.
«cntd»	***linksOnSection***	<u>*LinkOnSection*</u>	0:*	LINKs on SECTION that make up this SECTION.

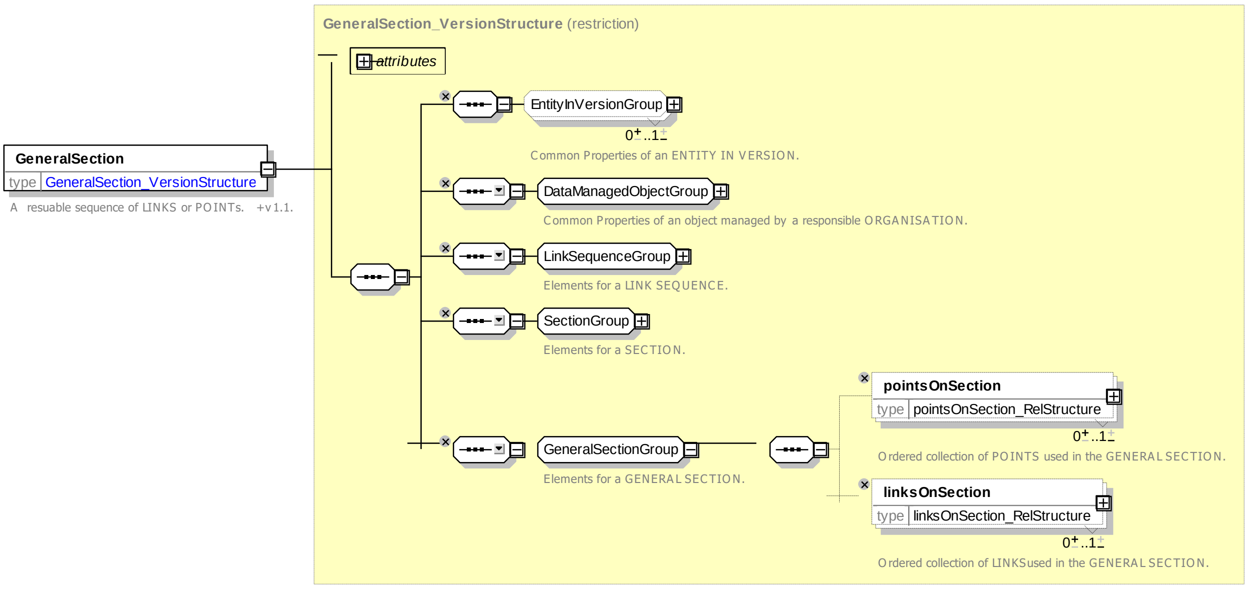


Figure 163 —*GeneralSection – XSD*

7.6.6.4 XML Example of General Section

The following fragment shows an example of a GENERAL SECTION.

```
<generalections>
        <!--- ===SCommon Section ========= -->

        </ GeneralSection > <GeneralSection nameOfLinkClass="ServiceLink" version="1" id="txc:Route_Section@11534">
                <PurposeOfGroupingRef version="any" ref="txc: _"/>
                <linksOnSection>
                    <LinkOnSection order="1" version="1" id="txc:11534_1">
                        <ServiceLink version="1" id="txc:11534_1">
                            <Distance>340</Distance>
                            <gml:LineString srsName="UKOS" gml:id="ID_11534_1">
                                <gml:pos>441325 275498</gml:pos>
                                <gml:pos>441332 275504</gml:pos>
                                <gml:pos>441350 275518</gml:pos>
                                <gml:pos>441611 275638</gml:pos>
                                <gml:pos>441623 275641</gml:pos>
                            </gml:LineString>
                            <FromPointRef version="1" ref="napt:4200F080500"/>
                            <ToPointRef version="1" ref="napt:4200F080501"/>
                        </ServiceLink>
                    </LinkOnSection>
                    <LinkOnSection order="2" version="1" id="txc:ServiceLink:11534_2">
                        <ServiceLink version="1" id="txc:ServiceLink:11534_2">
                            <Distance>1923</Distance>
                            <!-- <Direction>outbound</Direction> -->
                            <gml:LineString srsName="UKOS" gml:id="ID_11534_2">
                                <gml:pos>441623 275641</gml:pos>
                                <gml:pos>441630 275642</gml:pos>
                                <gml:pos>441689 275646</gml:pos>
                                <gml:pos>441703 275647</gml:pos>
                                <gml:pos>441723 275648</gml:pos>
                                <gml:pos>441838 275642</gml:pos>
                            </gml:LineString>
                            <FromPointRef version="1" ref="napt:4200F080501"/>
                            <ToPointRef version="1" ref="napt:4200F011200"/>
                        </ServiceLink>
                    </LinkOnSection>

…. etc
```

7.6.7 Generic Point & Link Sequence

7.6.7.1 Generic POINT & LINK SEQUENCE – Conceptual MODEL

The LINK SEQUENCE MODEL defines a set of POINTs and/or LINKs making up a path through a network.

It allows a path to be described as a sequence of points, a sequence of links, or both; both views are relevant for different use cases. LINK SEQUENCE is specialised by specific types of PATTERN, such as JOURNEY PATTERN, TIMING PATTERN etc.

All LINK SEQUENCEs have common properties – such as an over distance, some of which can be derived from the individual links.

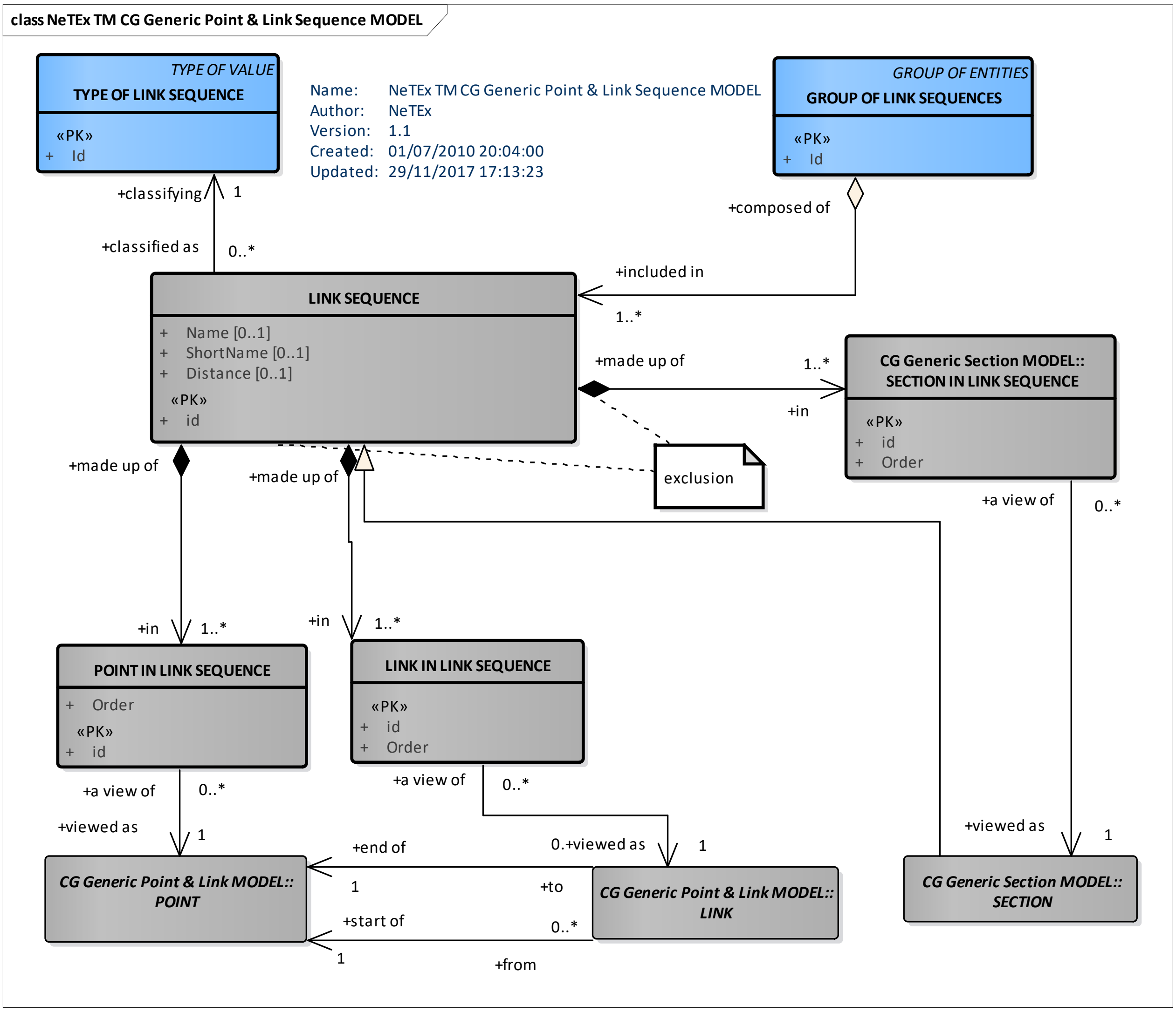


Figure 164 — Generic Point & Link Sequence – Conceptual MODEL (UML)

7.6.7.2 Generic Point & Link Sequence – Physical Model

The Physical Model implements LINK SEQUENCE as a framework element which can be further specialised by child instances.

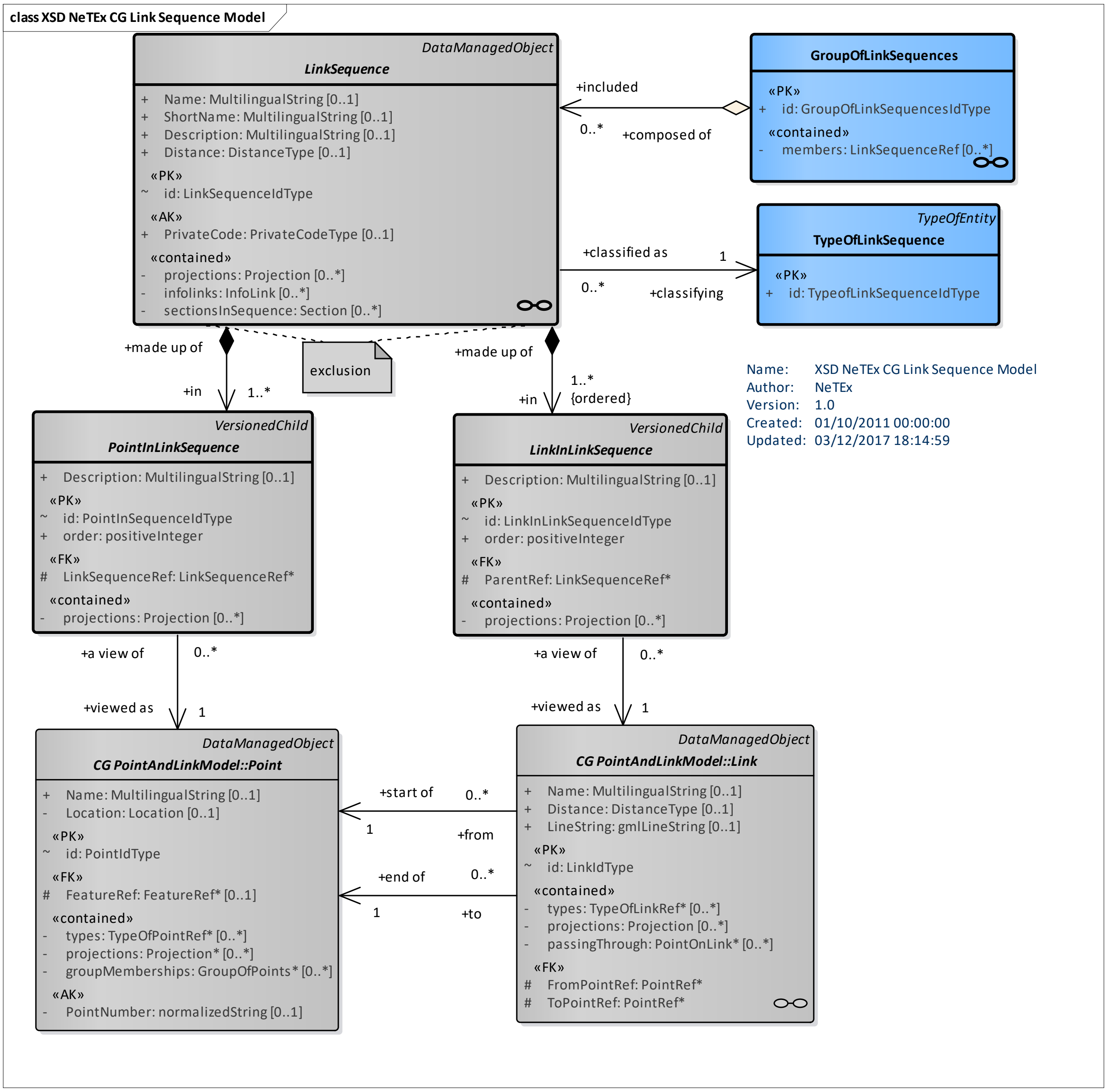


Figure 165 — Generic Point & Link Sequence – Physical Model (UML)

7.6.7.2.1 Explicit Link Sequences

The following figure summarises the specialisations of LINK SEQUENCEs in NeTEx.

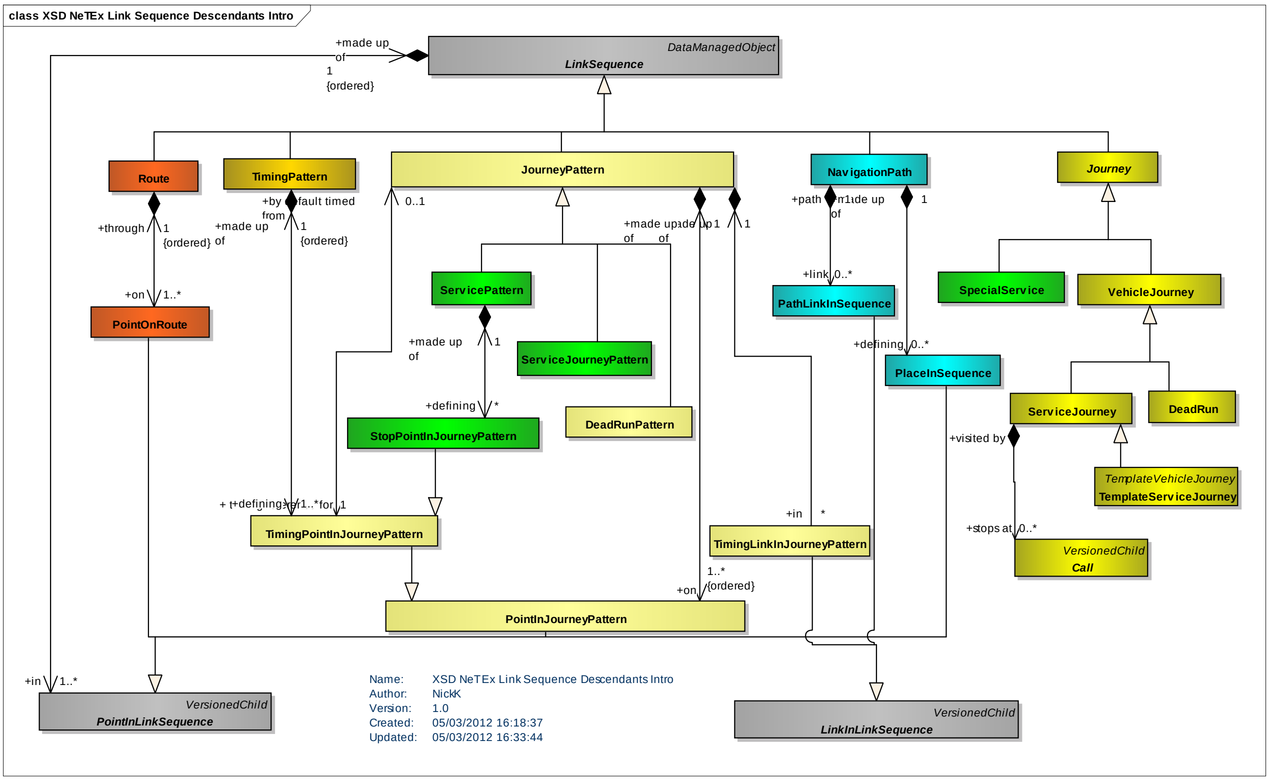


Figure 166 — Explicit Link Sequences Overview – Physical Model (UML)

7.6.7.3 Generic Point & Link Sequence – Attributes and XSD

7.6.7.3.1 LinkSequence – Model Element

An ordered sequence either of POINTs or of LINKs, defining a path through the network.

Table 111 — *LinkSequence – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	DataManagedObject	::>	LINK SEQUENCE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*LinkSequenceIdType*	1:1	Identifier of LINK SEQUENCE.
	Name	*MultilingualString*	0:1	Name of LINK SEQUENCE.
	Description	*MultilingualString*	0:1	Description of LINK SEQUENCE. +v1.1
	ShortName	*MultilingualString*	0:1	Short Name of LINK SEQUENCE.
	Distance	*DistanceType*	1:1	Distance of LINK SEQUENCE. Units are as specified for Frame. The default is '*SiMetres*'.
	PrivateCode	*PrivateCodeType*	0:1	Private Code of LINK SEQUENCE.
«cntd»	***projections***	*Projection*	0:*	PROJECTIONs for LINK SEQUENCE. +v1.1

«cntd»	***infoLinks***	*InfoLink*	0:*	InfoLinks associated with LINK SEQUENCE. +v1.1
«cntd»	***sectionsIn-Sequence***	*SectionInSequence*	0:*	SECTIONS that make up route. Can be used as an alternative to points in Seqnece. POINTS and LINKS must be of same type as sequence, eg ROUTE, ROUTE POINT, ROUTE LINK.

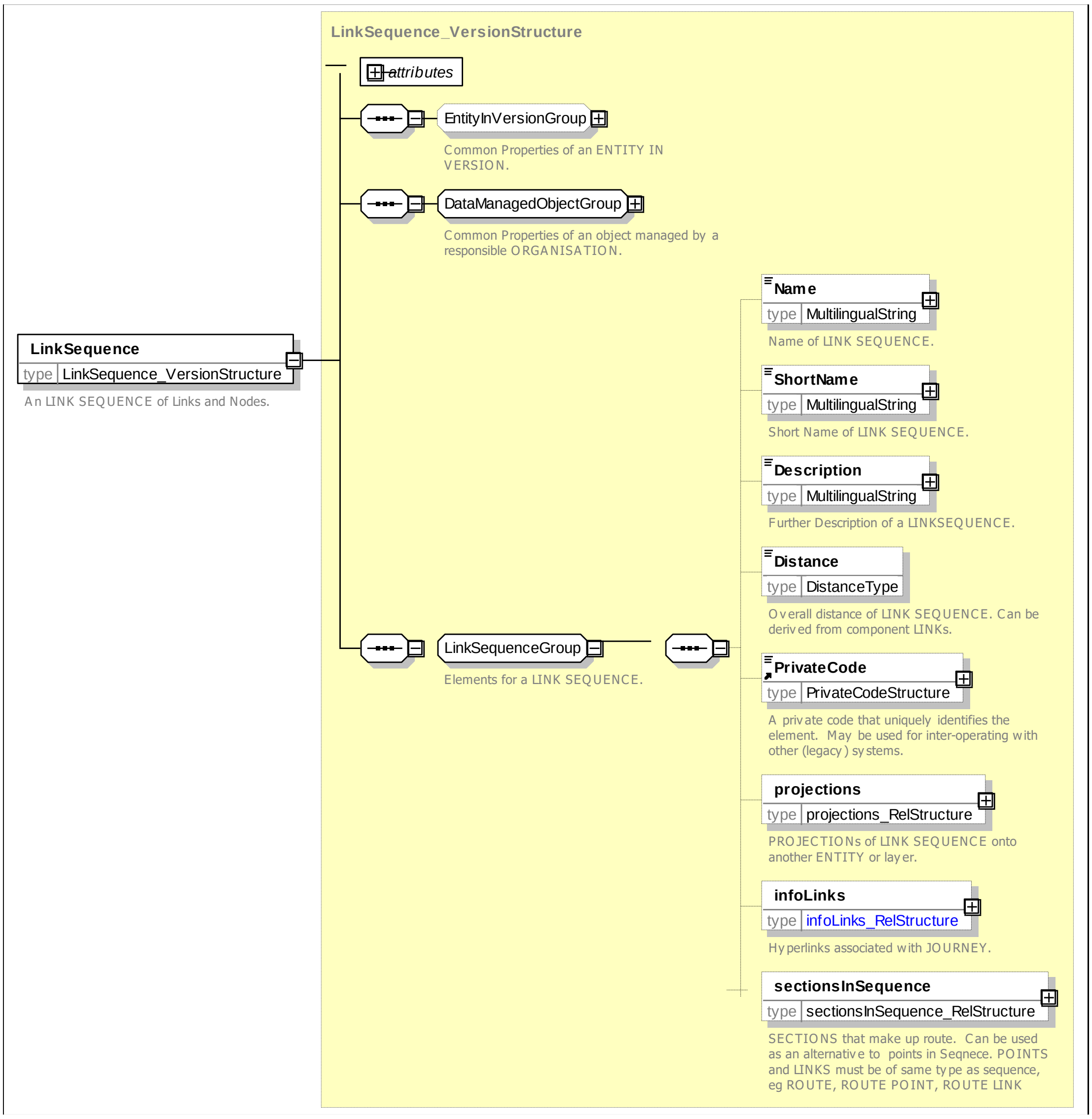


Figure 167 — *LinkSequence – XSD*

7.6.7.3.2 LinkInLinkSequence – Model Element

The order of a LINK in a LINK SEQUENCE to which it belongs.

Table 112 — *LinkInLinkSequence – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	<u>*VersionedChild*</u>	::>	LINK in LINK SEQUENCE inherits from VERSIONED CHILD.
«PK»	***id***	*LinkInLinkSequence-IdType*	1:1	Identifier of LINK in LINK SEQUENCE.
«atr», «PK»	***order***	*xsd:positiveInteger*	1:1	Order of LINK in LINK SEQUENCE.
«FK»	***LinkSequenceRef***	*(LinkSequenceRef)*	1:1	Parent LINK SEQUENCE of LINK in LINK SEQUENCE.
«ctnd»	***projections***	<u>*Projection*</u>	0:*	Projections of LINK in LINK SEQUENCE.
	Description	*MultilingualString*	0:1	Description of LINK in LINK SEQUENCE. +v1.1

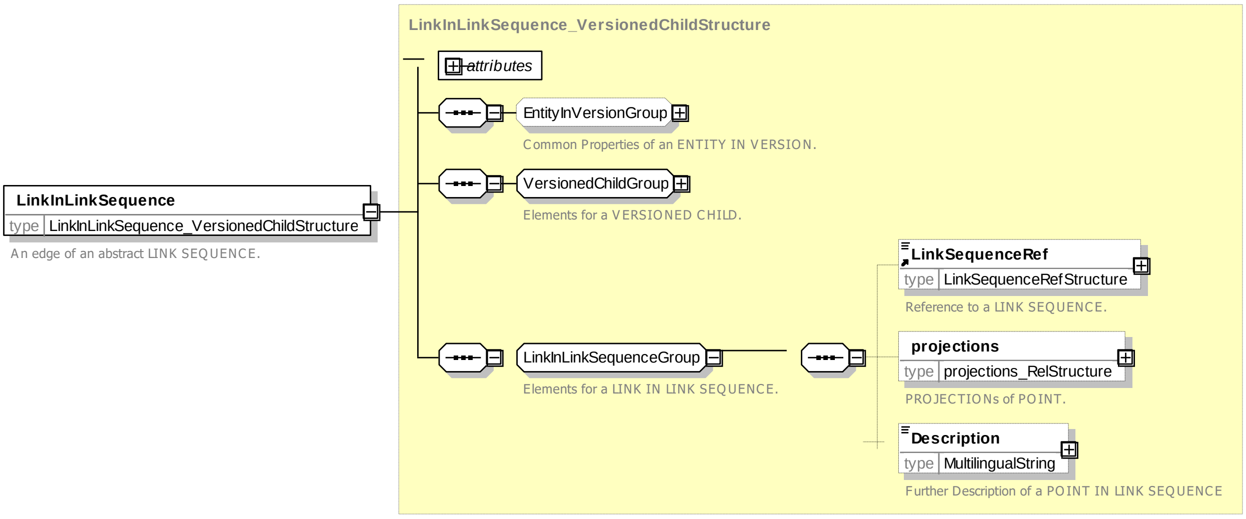


Figure 168 — *LinkInLinkSequence – XSD*

7.6.7.3.3 PointInLinkSequence – Model Element

A POINT in a LINK SEQUENCE indicating its order in that particular LINK SEQUENCE.

Table 113 — *PointInLinkSequence – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	POINT in LINK SEQUENCE inherits from VERSIONED CHILD.
«PK»	***id***	*PointInSequenceIdType*	1:1	Identifier of POINT in LINK SEQUENCE.
«atr»	***order***	*xsd:positiveInteger*	1:1	Order of POINT in LINK SEQUENCE within sequence.
«FK»	***LinkSequenceRef***	*(LinkSequenceRef)*	1:1	Reference to LINK SEQUENCE containing POINT in LINK SEQUENCE.

«containment»	***projections***	*Projection*	0:*	Projections of POINT in LINK SEQUENCE.
	Description	*MultilingualString*	0:1	Description of POINT in LINK SEQUENCE. +v1.1

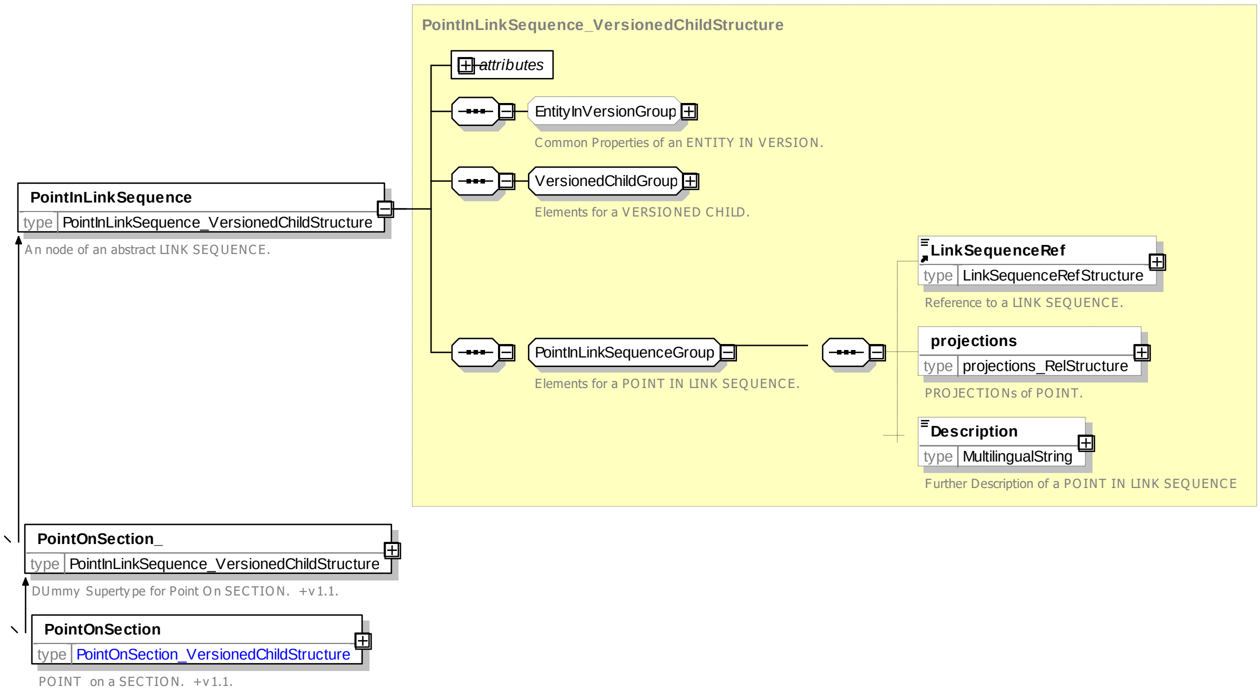


Figure 169 — *PointInLinkSequence – XSD*

7.6.7.3.4 GroupOfLinkSequences – Model Element

A grouping of LINK SEQUENCEs.

Table 114 — *GroupOfLinkSequences – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF LINK SEQUENCEs inherits from GROUP OF ENTITies.
«PK»	***id***	*GroupOfLinkSequencesIdType*	1:1	Identifier of GROUP OF LINK SEQUENCEs.
«cntd»	***members***	*(LinkSequenceRef)*	0:*	Member of a GROUP OF LINK SEQUENCEs.

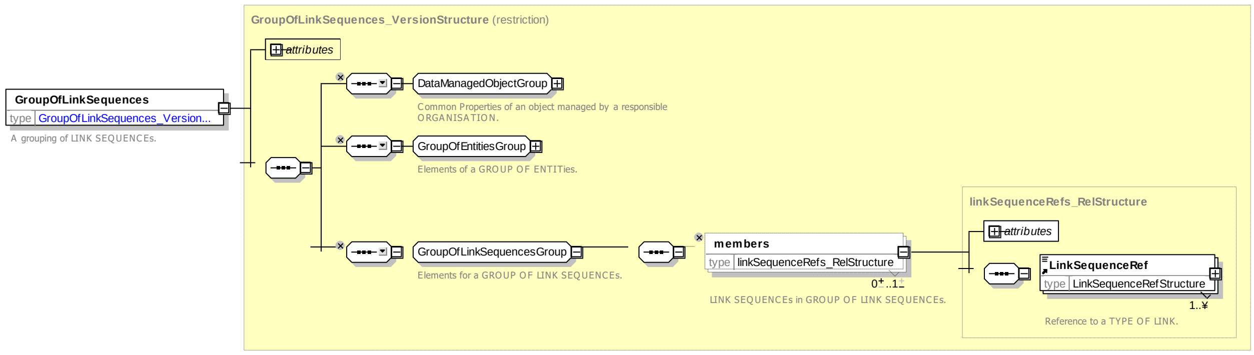


Figure 170 — *GroupOfLinkSequences – XSD*

7.6.7.3.5 TypeOfLinkSequence – Model Element

A classification of LINK SEQUENCEs used to define the different functions a LINK SEQUENCE may be used for. E.g. ROUTE, JOURNEY PATTERN, road, TRIP PATTERN, border line etc.

Table 115 — *TypeOfLinkSequence – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF LINK SEQUENCE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeofLinkSequenceIdType*	1:1	Identifier of TYPE OF LINK SEQUENCE.

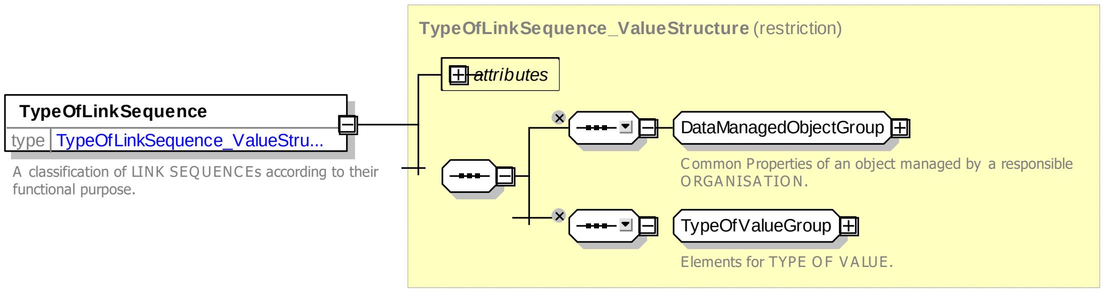


Figure 171 — TypeOfLinkSequence – XSD

7.6.8 Generic Zone and Feature

7.6.8.1 Generic ZONE and FEATURE – Conceptual MODEL

7.6.8.1.1 Zone – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

A ZONE is a two-dimension object used in the network description (e.g. administrative area, tariff zone, flexible transport zone). ZONEs are classified according to a TYPE OF ZONE.

A ZONE may be defined by a GROUP OF POINTs belonging to the ZONE. For instance, a TARIFF ZONE is composed of SCHEDULED STOP POINTs.

A ZONE may also be defined as a geometric area, bordered by a LINK SEQUENCE. In such a case, this LINK SEQUENCE has to be a closed one (i.e. the first and last POINTs IN LINK SEQUENCE must be a view of the same POINT).

A ZONE may be recursive, and include other smaller ZONEs. This is expressed by the reflexive relationship on ZONE. For example, a STOP PLACE may include other STOP PLACEs, or an ADMINISTRATIVE ZONE other ADMINISTRATIVE ZONEs.

A ZONE may be represented by a single POINT (names centroid). Within one particular layer, this representing POINT has to be unique. This also defines a special kind of point: when a reference is done to the centroid of a ZONE, this accounts for a reference to the ZONE itself (the centroid POINT only being a representative of the ZONE). This is an important issue for flexible transport modelling (as a stop is often a ZONE for FTS).

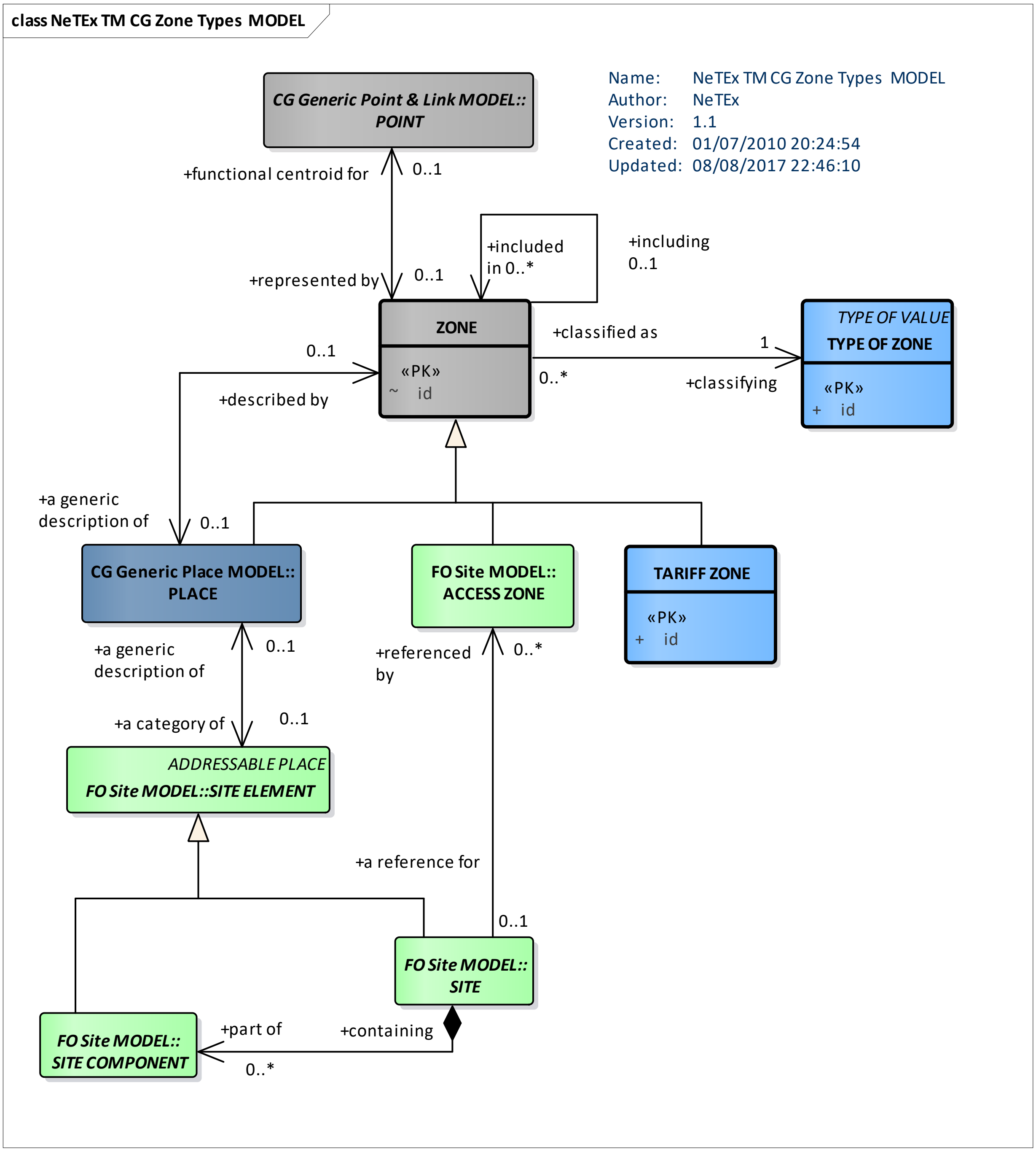


Figure 172 — Zone – Conceptual MODEL (UML)

7.6.8.1.2 Feature – Conceptual MODEL

It is often necessary to define a group of objects of different types in a simpler representation, omitting the details. For instance, a train station composed of tracks, platforms, vending machines, etc., or a depot composed of halls, parking areas, lanes, maintenance facilities, etc., are viewed in some layers as single POINTs. This is described by the entity COMPLEX FEATURE (named by analogy with the GDF standard and usual GIS wording).

A COMPLEX FEATURE is composed of one or more SIMPLE FEATUREs. A SIMPLE FEATURE is identical to an instance of either a POINT, a LINK, or a ZONE.

A COMPLEX FEATURE usually combines elements of different kinds as POINTs, LINKs, ZONEs (each of them not necessarily of the same type), and even other COMPLEX FEATUREs. It should not be mixed up with a group of elements (e.g. GROUP OF POINTs), combining elements of one single type only (e.g. one GROUP OF LINKs may be all LINKs in a tunnel, which is not a COMPLEX FEATURE).

As a ZONE, a COMPLEX FEATURE may be represented by a single POINT.

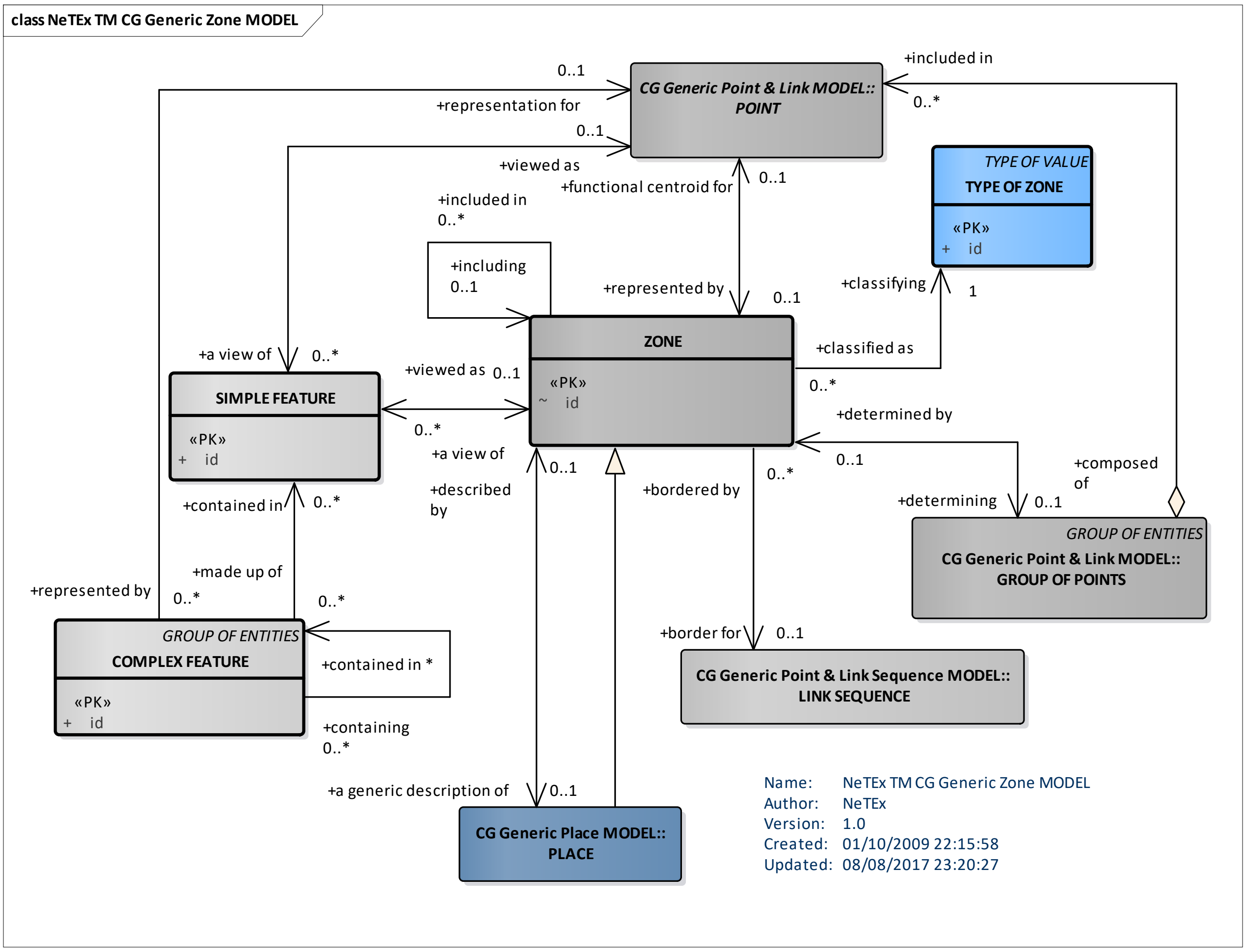


Figure 173 — Generic Zone and Feature – Conceptual MODEL (UML)

7.6.8.2 Generic Zone and Feature model – Physical Model

The ZONE Physical Model is similar to the ZONE Conceptual MODEL. It additionally models ZONE as a specialisation of the GROUP OF POINTs object in order to model the fact that a ZONE can contain (or be built from) a set of POINTs.

The figure also shows the main explicit specialisations of ZONE found in NeTEx (STOP AREA, PLACE, TARIFF ZONE, ACCESS ZONE, etc.)

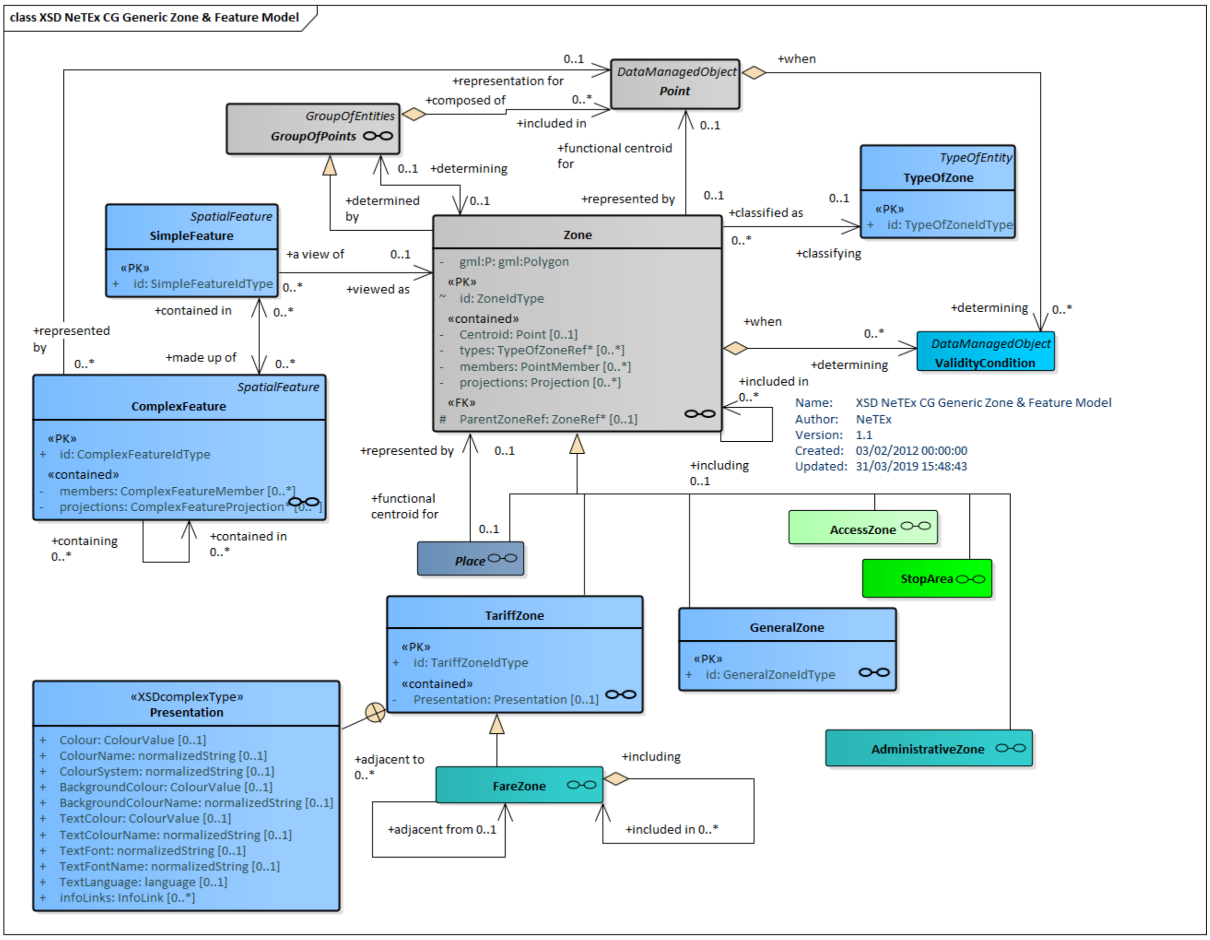


Figure 174 — Generic Zone & Feature – Physical Model (UML)

7.6.8.3 Generic Zone & Feature – Attributes and XSD

7.6.8.3.1 Zone – Model Element

A two-dimensional PLACE within the service area of a public transport operator (administrative zone, TARIFF ZONE, ACCESS ZONE, etc.).

Table 116 — *Zone – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*GroupOfPoints*</u>	::>	ZONE inherits from GROUP OF POINTs.
«PK»	***id***	*ZoneIdType*	1:1	Identifier of ZONE.
«cntd»	***members***	*(PointRef)*	0:*	List of points within ZONE.
«cntd»	***types***	*TypeOfZoneRef*	0:*	Type of ZONE.
«cntd»	***Centroid***	<u>*SimplePoint*</u>	0:1	Centre POINT of ZONE.

	gml:Polygon	*gml:Polygon*	0:1	Polygon associated with zone.
«cntd»	***projections***	*Projection*	0:*	List of PROJECTIONs of ZONE.
«FK»	***ParentZoneRef***	*ZoneRef*	0:1	Reference to parent of ZONE.

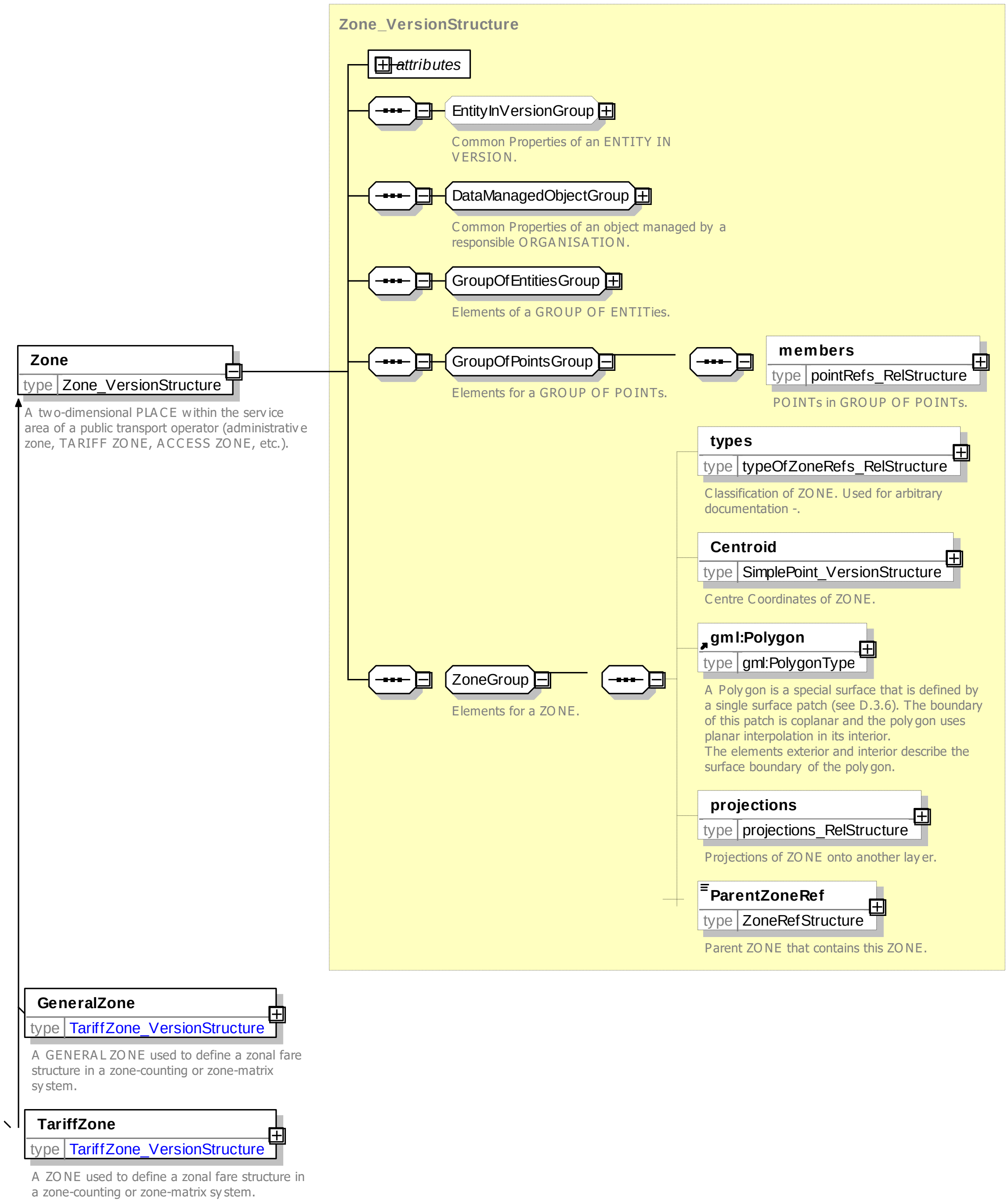


Figure 175 — *Zone – XSD*

7.6.8.3.2 TariffZone – Model Element

A ZONE used to define a zonal fare structure in a zone-counting or zone-matrix system.

Table 117 — *TariffZone – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Zone*	::>	TARIFF ZONE inherits from ZONE.
	id	*TariffZoneIdType*	1:1	Identifier of TARIFF ZONE.
«cntd»	***Presentation***	*Presentation*	0:1	Presentation features such as colour associated with TARIFF ZONE. +V1.1
«cntd»	***Printed-Presentation***	*PrintedPresentation*	0:1	Printed Presentation features such as colour associated with TARIFF ZONE. +V1.1

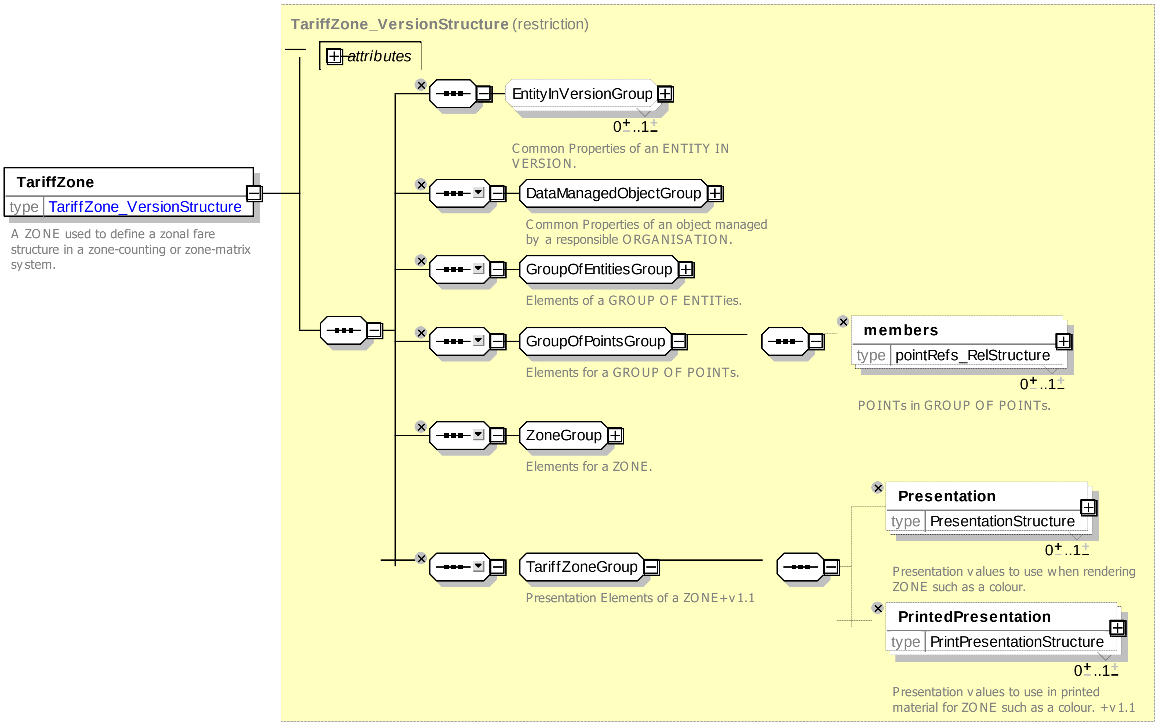


Figure 176 — *TariffZone – XSD*

7.6.8.3.3 GeneralZone – Model Element

A ZONE used to define a zonal area for any arbitrary purpose.

Table 118 — *GeneralZone – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Zone*	::>	GENERAL ZONE inherits from ZONE.
	id	*ZoneIdType*	1:1	Identifier of GENERAL ZONE.

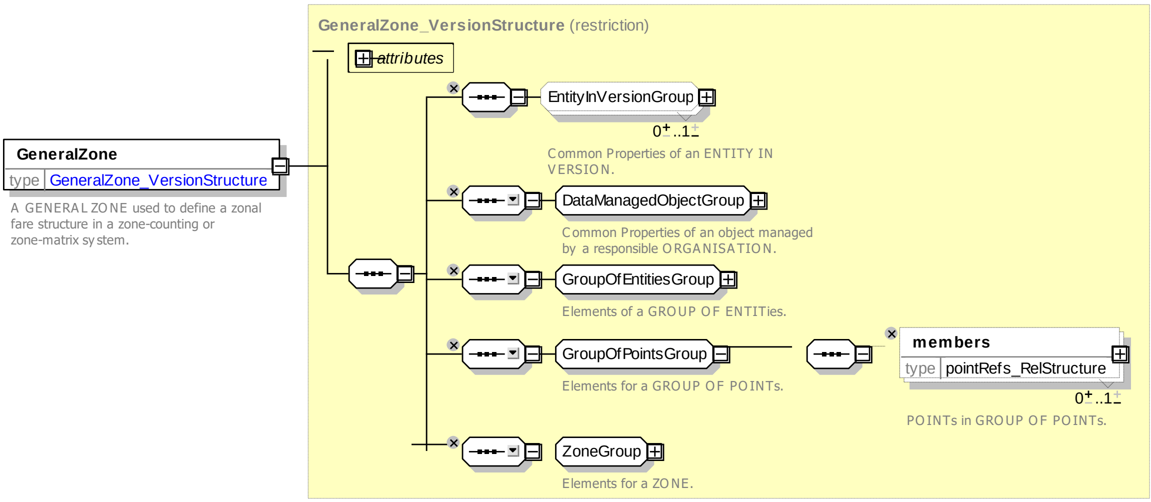


Figure 177 — *GeneralZone – XSD*

7.6.8.3.4 TypeOfZone – Model Element

A classification of ZONEs. E.g. TARIFF ZONE, ADMINISTRATIVE ZONE.

Table 119 — *TypeOfZone – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF ZONE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfZoneIdType*	1:1	Identifier of TYPE OF ZONE.

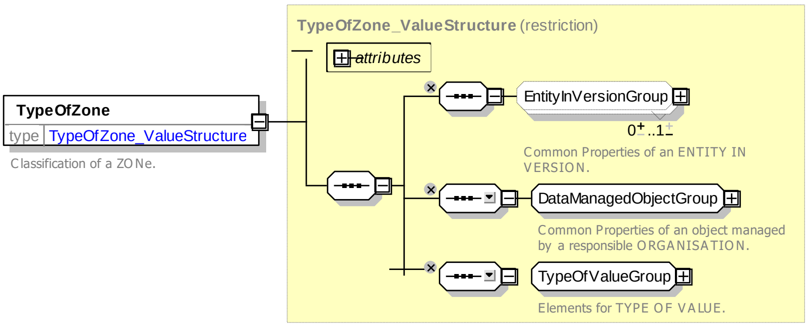


Figure 178 — *TypeOfZone – XSD*

7.6.8.3.5 SpatialFeature – Model Element

An abstract representation of elementary objects related to the spatial representation of the network POINTs (0-dimensional objects), LINKs (1-dimensional objects) and ZONEs (2-dimensional objects) may be viewed as SIMPLE FEATUREs.

Table 120 — *SpatialFeature – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	<u>*GroupOfPoints*</u>	::>	SPATIAL FEATURE inherits from GROUP OF POINTS.
«PK»	***id***	*FeatureIdType*	1:1	Identifier of Simple feature

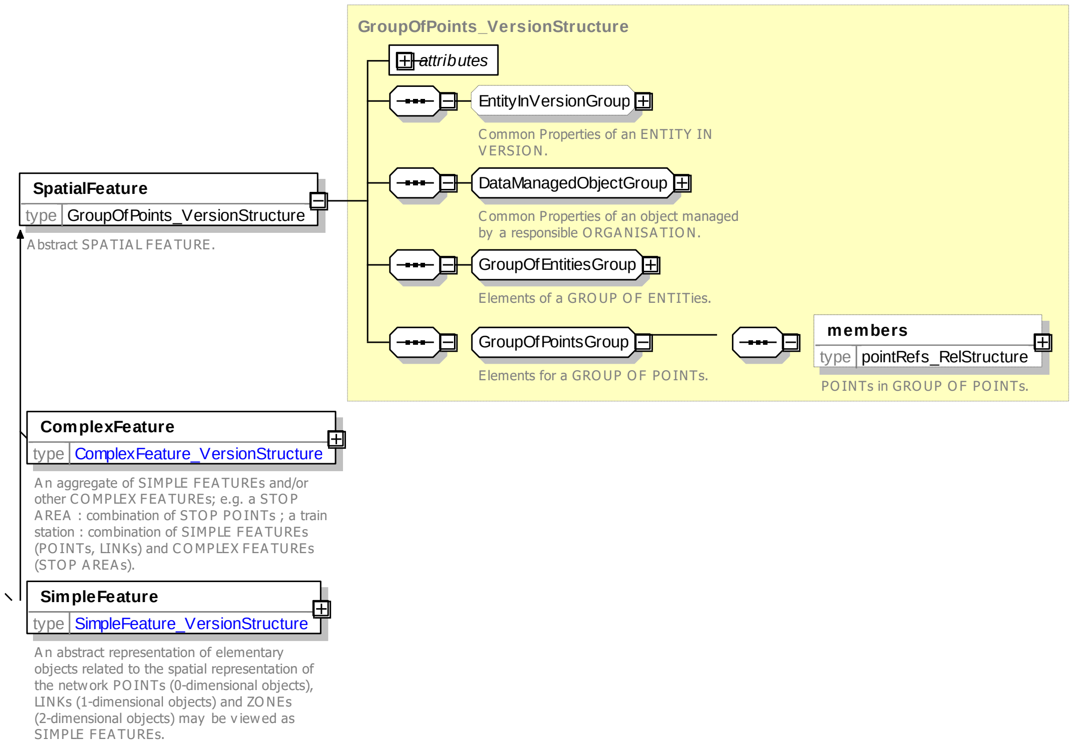


Figure 179 — *SpatialFeature – XSD*

7.6.8.3.6 ComplexFeature – Model Element

An aggregate of SIMPLE FEATUREs and/or other COMPLEX FEATUREs; e.g. a STOP AREA : combination of SCHEDULED STOP POINTS ; a train station : combination of SIMPLE FEATUREs (POINTs, LINKs) and COMPLEX FEATUREs (STOP AREAs).

Table 121 — *ComplexFeature – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*SpatialFeatureGroup-OfPoints*</u>	::>	COMPLEX FEATURE inherits from SPATIAL FEATUREPOINTs.
«PK»	***id***	*ComplexFeatureIdType*	1:1	Identifier of COMPLEX FEATURE.
«cntd»	***GroupOfEntitiesRef***	*(GroupOfEntitiesRef)*	0:1	References to GROUP OF ENTITies represented by this feature.
«cntd»	***Feature-Members***	<u>*ComplexFeatureMember*</u>	0:*	List of references to SIMPLE FEATUREs that are members of COMPLEX FEATURE.

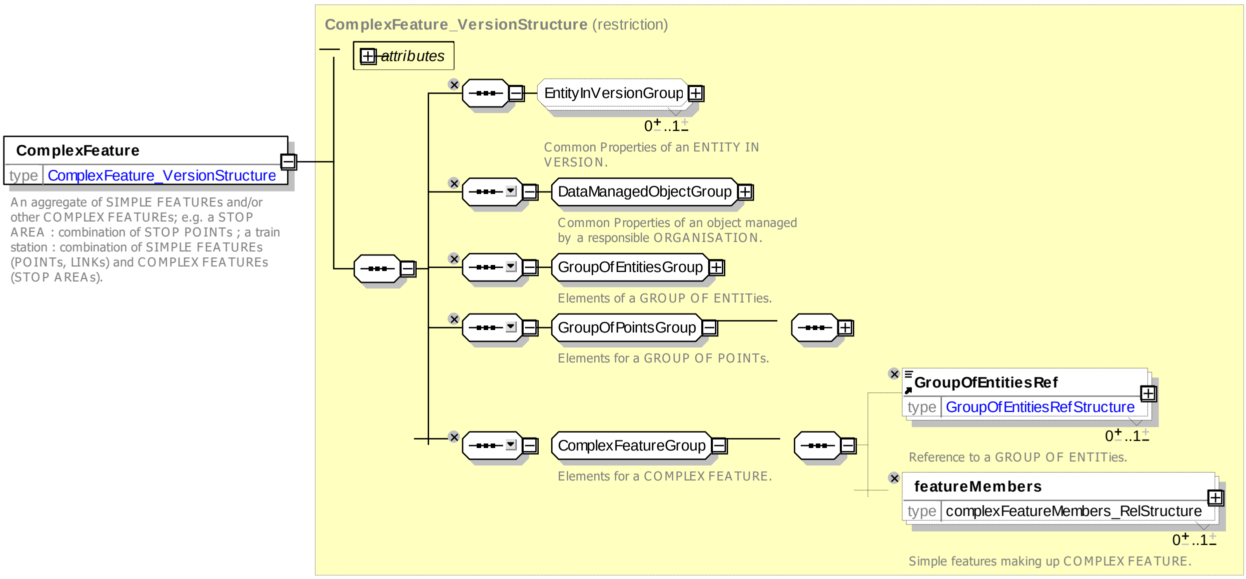


Figure 180 — *ComplexFeature – XSD*

7.6.8.3.7 SimpleFeature – Model Element

An abstract representation of elementary objects related to the spatial representation of the network POINTs (0-dimensional objects), LINKs (1-dimensional objects) and ZONEs (2-dimensional objects) may be viewed as SIMPLE FEATUREs.

Table 122 — *SimpleFeature – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfPoints*	::>	SIMPLE FEATURE inherits from GROUP OF POINTs.
«PK»	***id***	*SimpleFeatureIdType*	1:1	Identifier of SIMPLE FEATURE.
«FK»	***ZoneRef***	*ZoneRef*	0:1	Reference to ZONE that this feature represents.

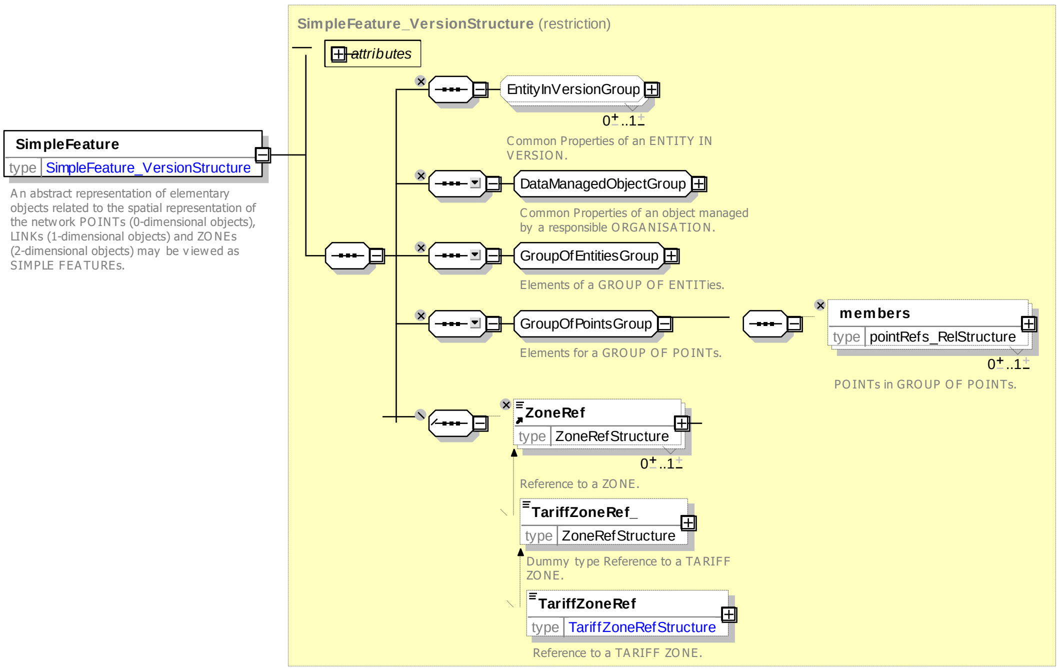


Figure 181 — *SimpleFeature – XSD*

7.6.9 Generic Projection

7.6.9.1 Generic PROJECTION – Conceptual MODEL

The ENTITIES used for describing a transport network can be grouped into different LAYERs. Each LAYER is associated with a certain LOCATING SYSTEM, and the entities belonging to the LAYER have a position within this LOCATING SYSTEM. Examples of layers include:

— Infrastructure layer: describes road or rail network.

— Route layer: describes route topology.

— Service layer: describes network of stops served by routes.

— Timing layer: describes location of timing points and times between them.

Different aspects of the network planning process deal with different layers of data. For instance, strategic planning does not have to deal with details of the infrastructure like signals, switch points etc., but tactical planning may very well have to do so.

Obviously there are many cases where information from different layers is needed to produce a result: e.g. a map showing routes and stops needs to be drawn, distances between passenger stops need to be calculated for statistics analysis etc. The mechanism for relating ENTITIES of one LAYER to ENTITIES of another LAYER is called projection. Projection can happen implicitly by transforming the entity position from the source LOCATION SYSTEM to the destination LOCATION SYSTEM. However, there are cases where such automatic transformation is not possible or practical, e.g. if a route needs to be displayed on a schematic map, there is no way of calculating the positions from the spatial coordinates. TRANSMODEL and NeTEx therefore contain a mechanism for explicitly projecting (spatial) entities of one layer to another layer.

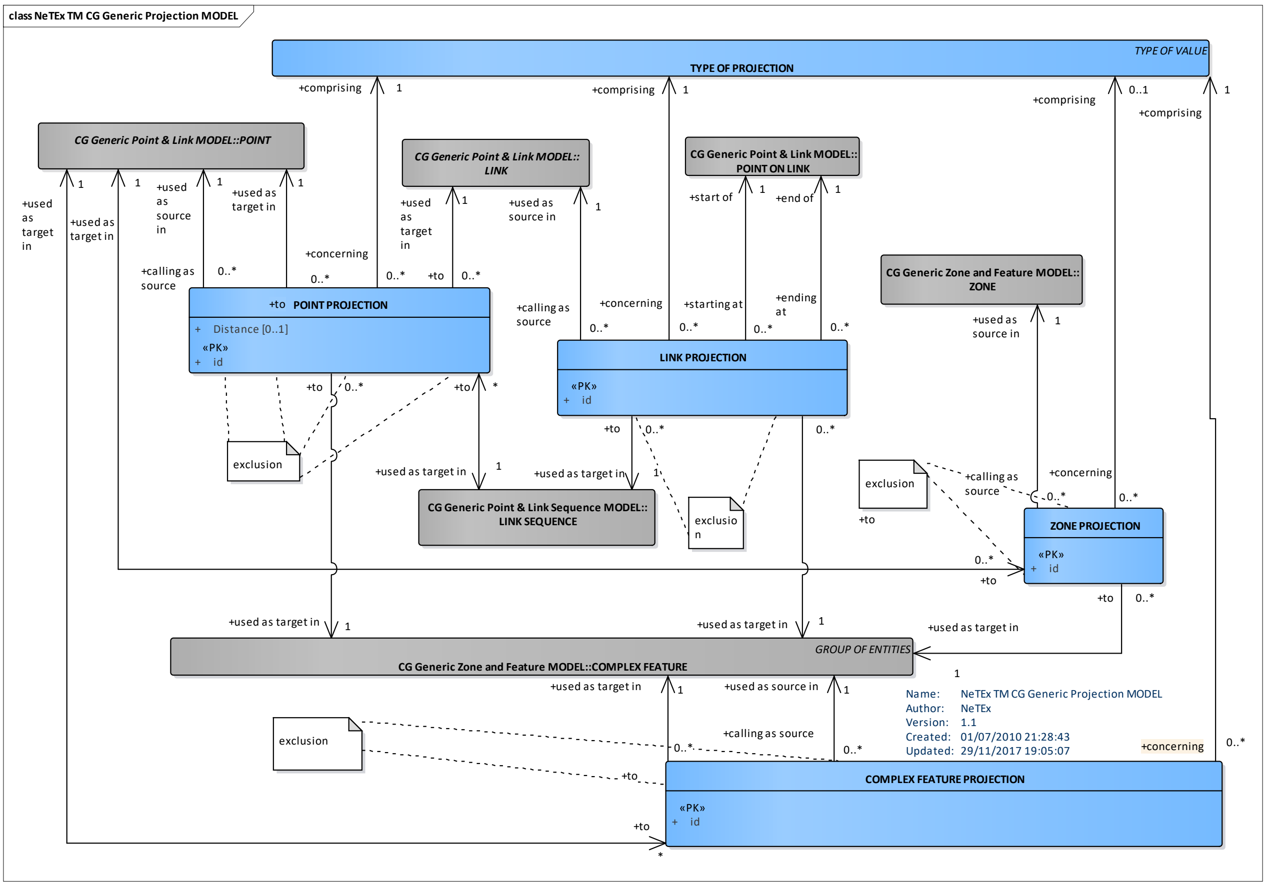


Figure 182 — Generic Projection – Conceptual MODEL (UML)

7.6.9.1.1 Point Projection – Conceptual MODEL

Explicit projection is possible between POINTs, LINKs and ZONEs. The TYPE OF PROJECTION element identifies the source and target layer of the projection and describes its purpose. A POINT may be related to a POINT on a different layer, e.g. a timing point may be projected on a stop point.

The POINT PROJECTION is used to project a point of a source layer to an ENTITY of the target layer. The target ENTITY can be POINT or LINK, but not ZONE. If the target of POINT PROJECTION is a link, the distance from the start of the link is set in POINT PROJECTION.

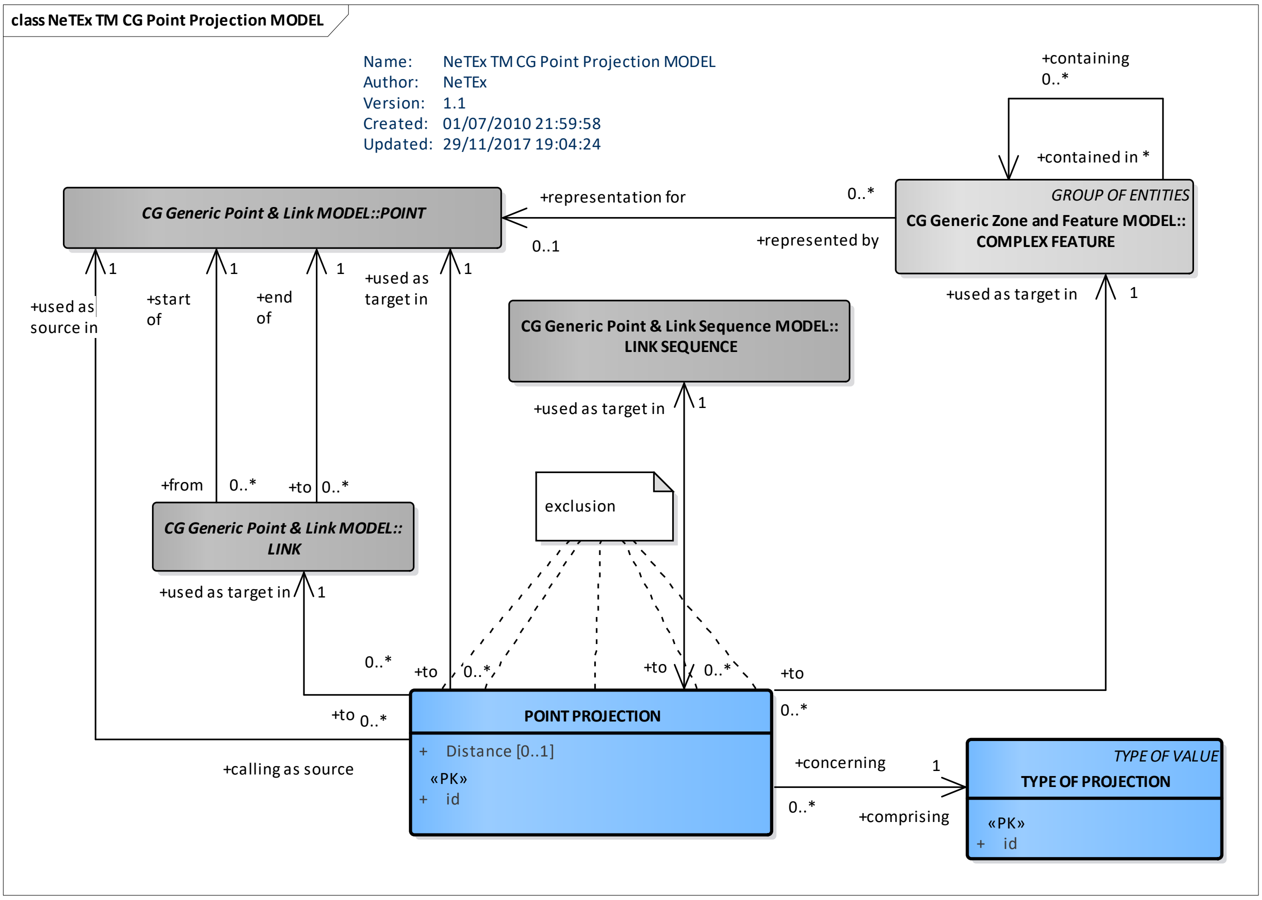


Figure 183 — Point Projection – Conceptual MODEL (UML)

7.6.9.1.2 Link Projection – Conceptual MODEL

The LINK PROJECTION is used to project a LINK on one or more LINKs of another layer. As a precondition, the destination link must have one or more POINT ON LINK entities associated to it. The start and end point of the source link are projected on POINT ON LINK of the destination LINKs. An example of LINK projection might be the projection of a SERVICE PATTERN LINK between two stops to the LINKs of the (road or rail).

The ZONE projection relates a point to a ZONE, e.g. a stop to a fare zone.

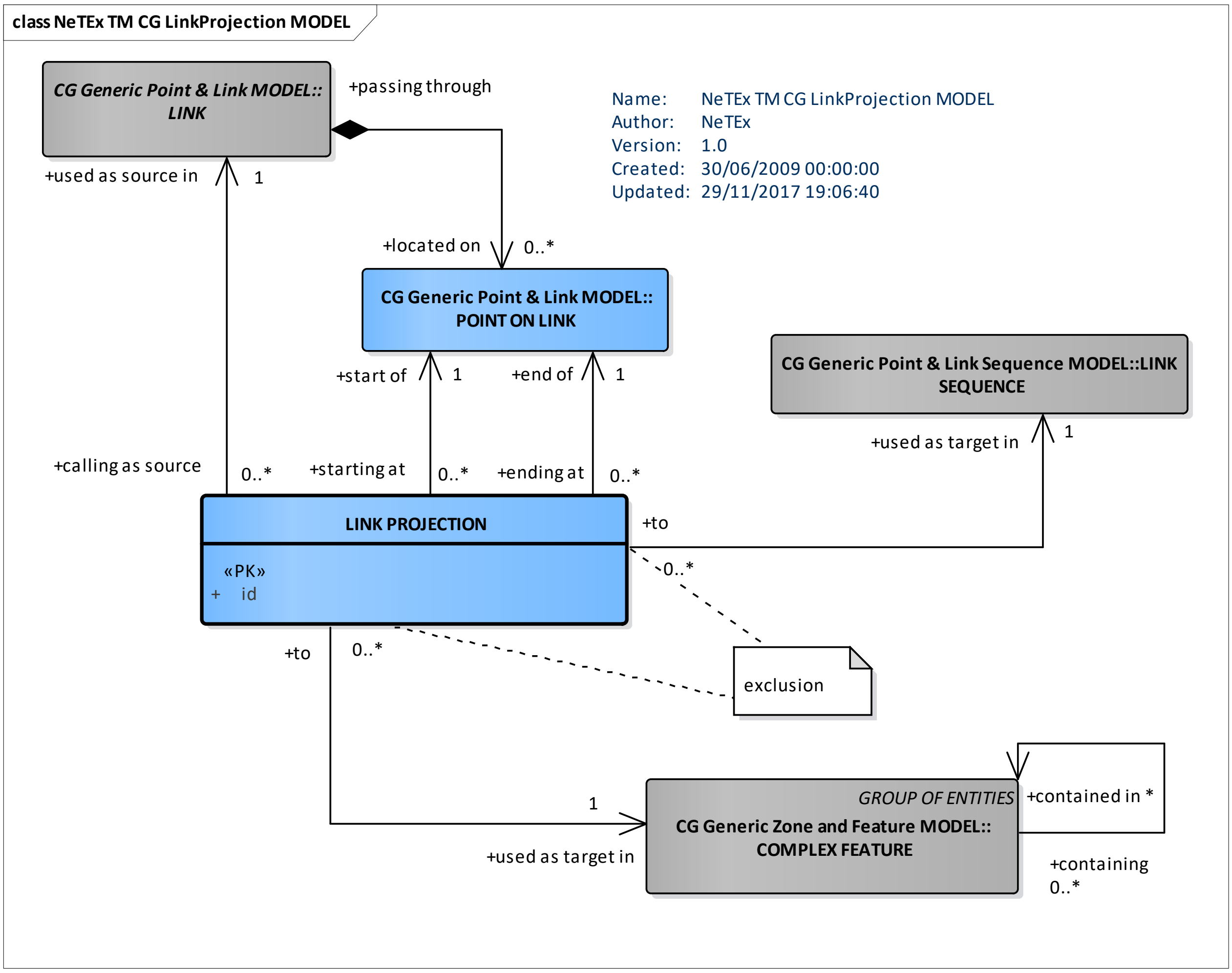


Figure 184 — Link Projection – Conceptual MODEL (UML)

7.6.9.1.3 Shape of Linear Objects – Conceptual MODEL

It is also possible to associate an arbitrary LINE SHAPE as a sequence of points with a LINK or LINK SEQUENCE. This can be used to plot ROUTEs and other objects onto a map.

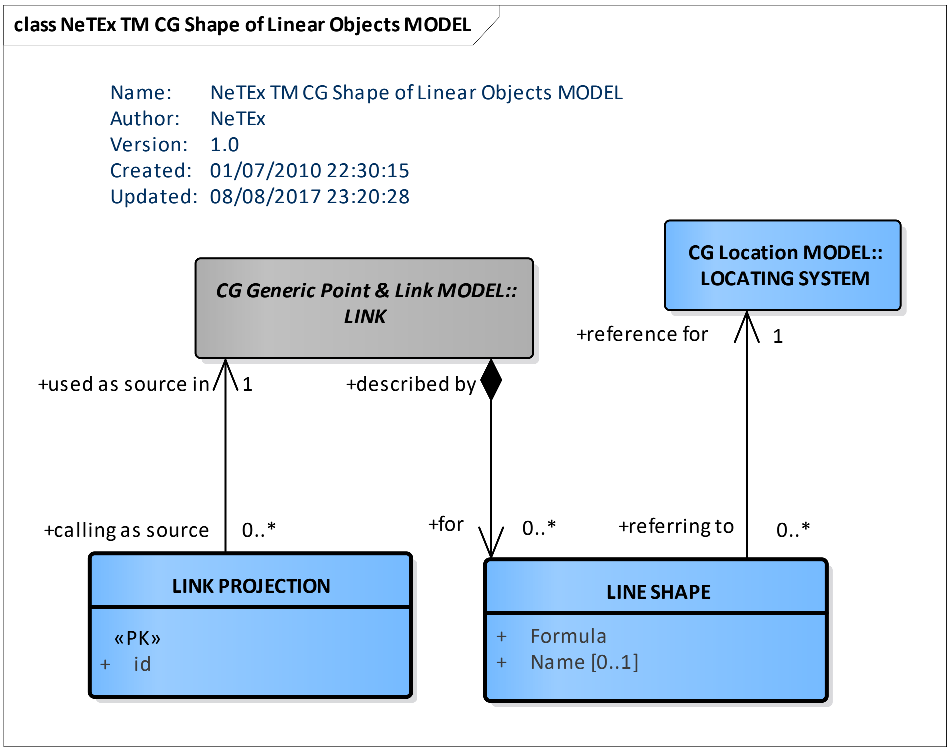


Figure 185 — Line Shape – Conceptual MODEL (UML)

7.6.9.1.4 Projection Example

In this section we show some examples of projection. The following screenshot shows a service pattern in a scheduling system :

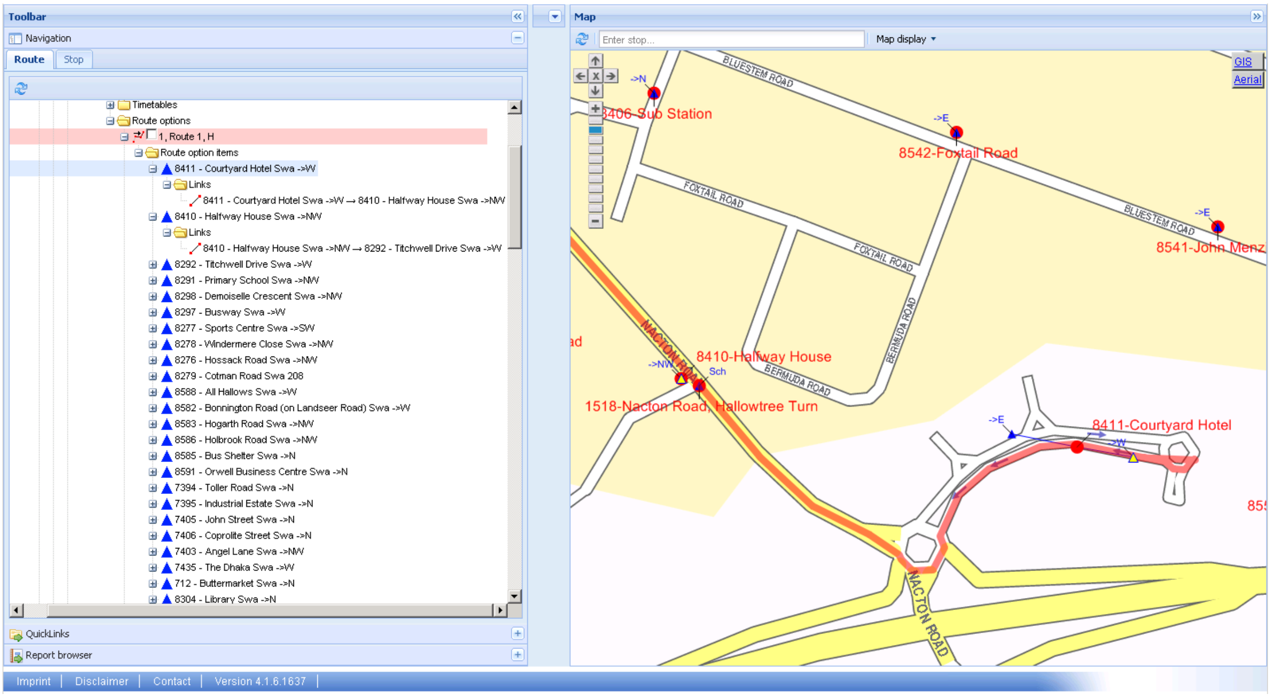


Figure 186 — Service Pattern on screen of scheduling system

The left window shows a service pattern (= route option) as a sequence of stops. The stops are connected by service pattern links, which carry certain information (e.g. distance). The right screen shows the selected service pattern on a map. However, the service pattern links do not hold any shape or route type information. This information is stored in the underlying GIS system. The following screenshot shows the GIS data of the map section visible above:

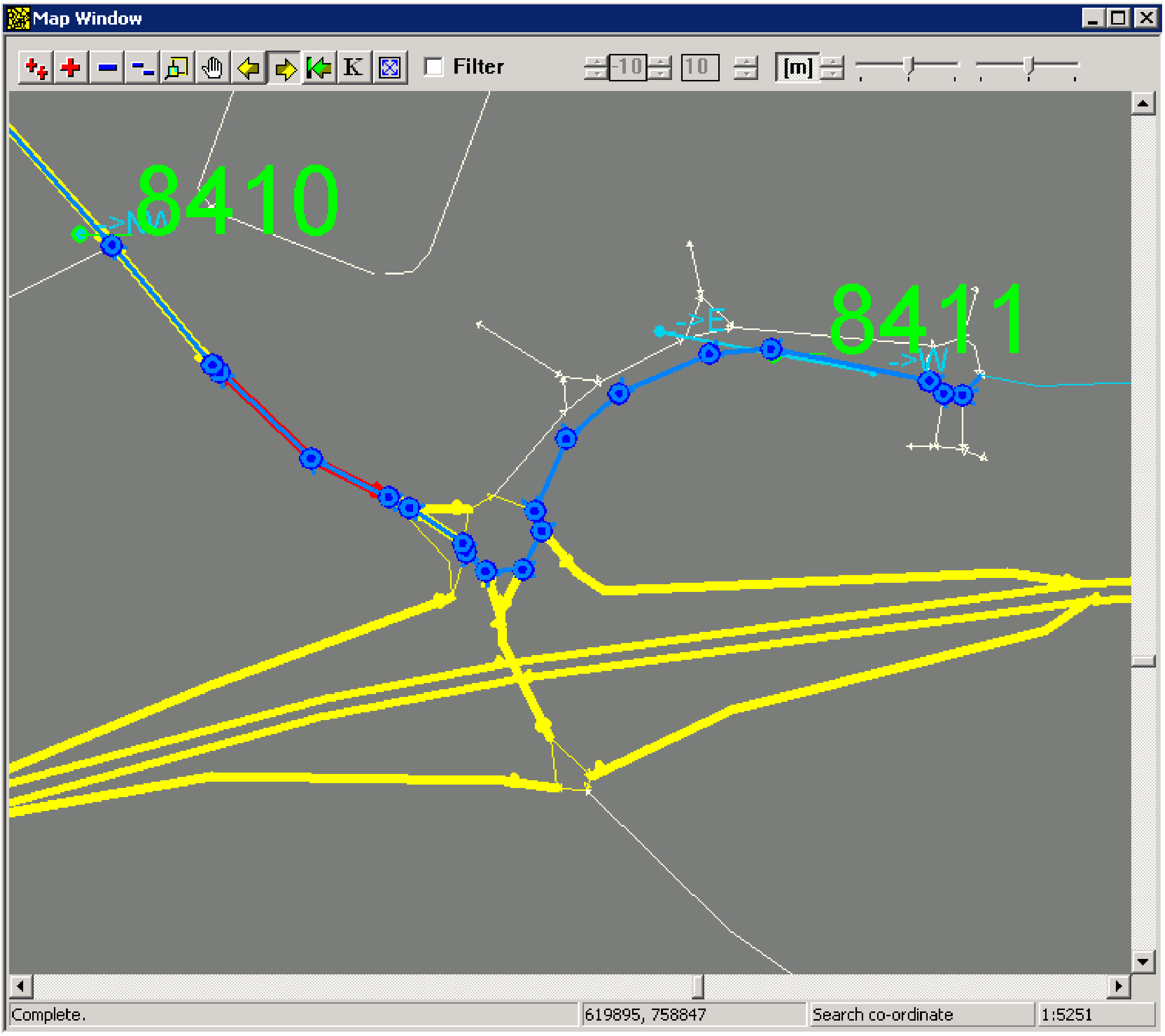


Figure 187 — GIS view of map screen

It can be seen that one service pattern link (here: from stop 8411 to 8410) covers many GIS links. Each GIS link contains a rich set of attributes: road type and name, speed category, allowed vehicle types, allowed direction of travel etc. The next screen shows the details of the GIS link highlighted in red:

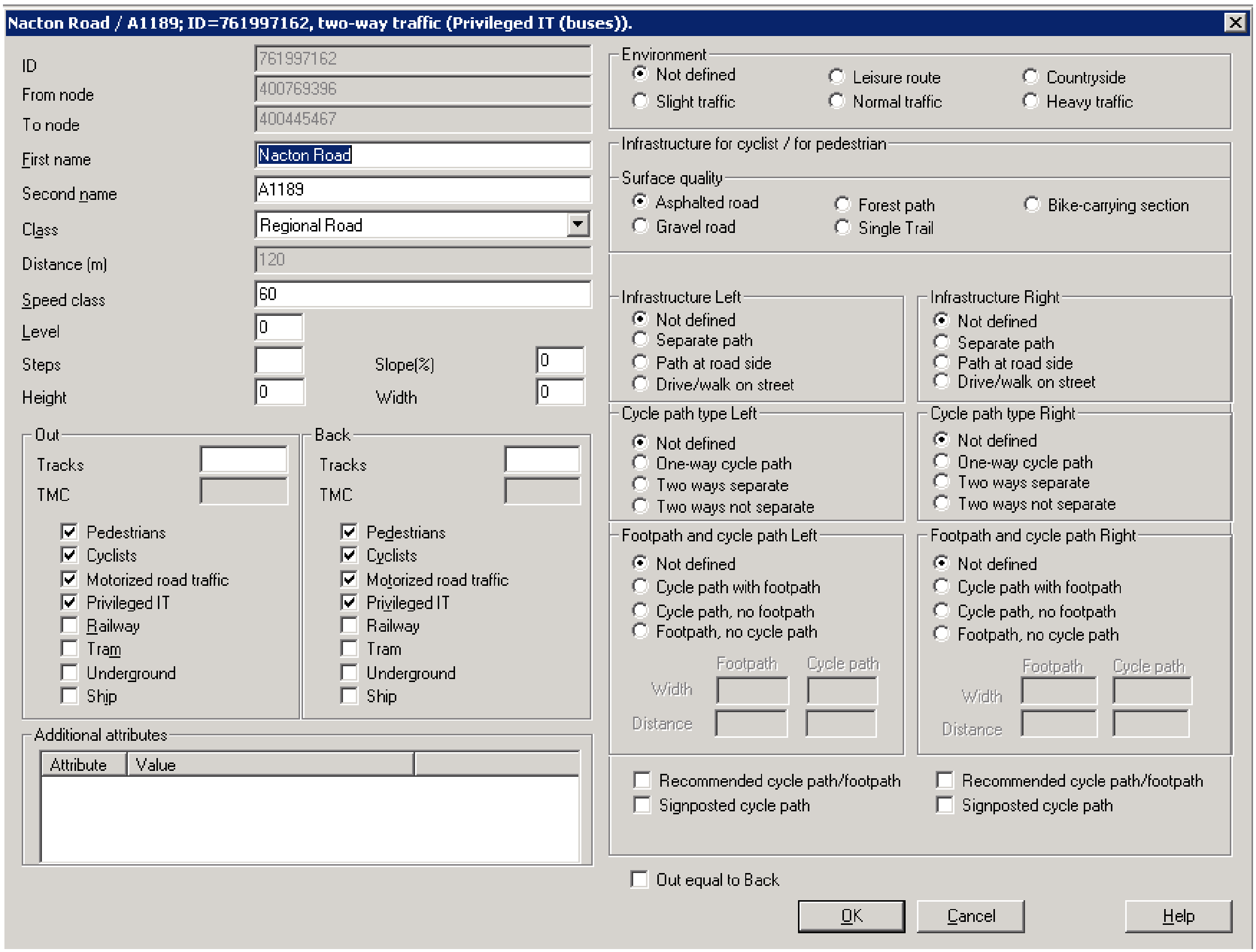


Figure 188 — Properties of an Infrastructure Link

It is now obvious that storing this information at service pattern link level is neither efficient nor even possible, since the attributes shown above could vary within the service pattern link. Instead, the service pattern link is projected on the infrastructure link, as shown in the screenshot below:

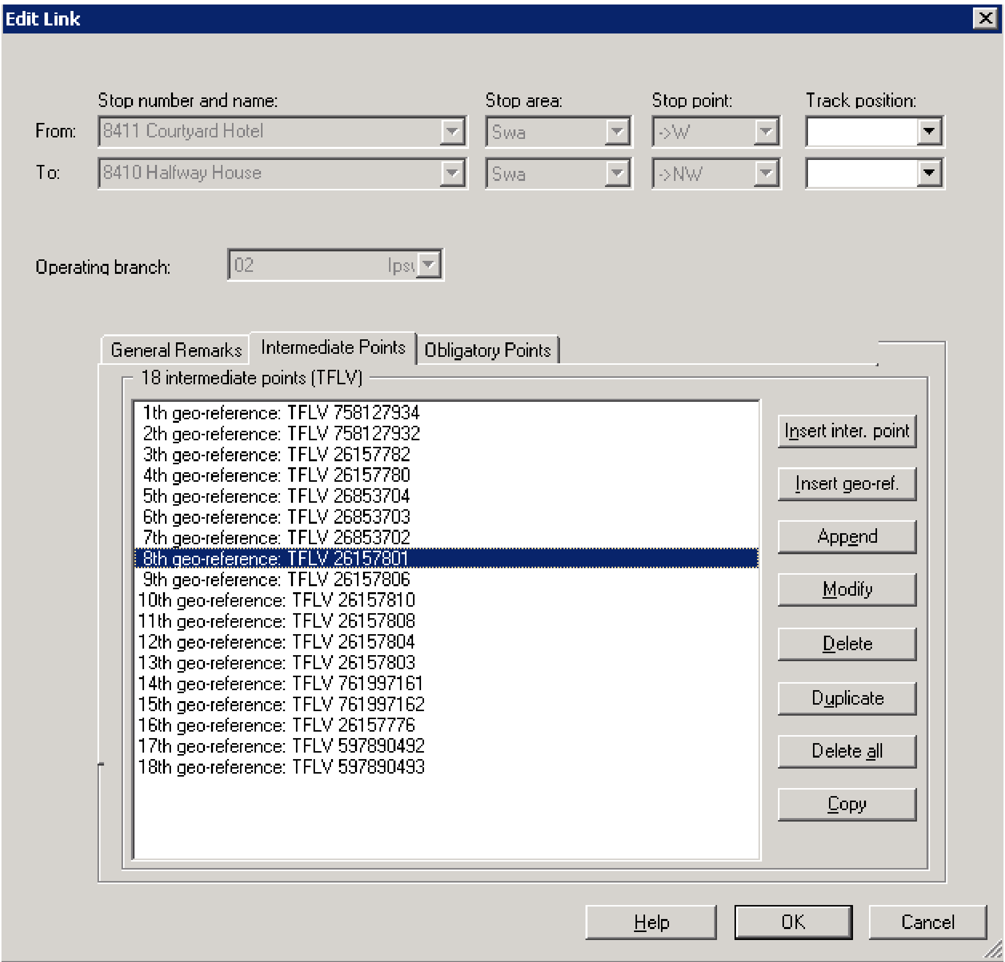


Figure 189 — Projection of SERVICE PATTERN LINK to INFRASTRUCTURE LINK

The start and end (stop) points of the service pattern link are also projected on the INFRASTRUCTURE LINKs:

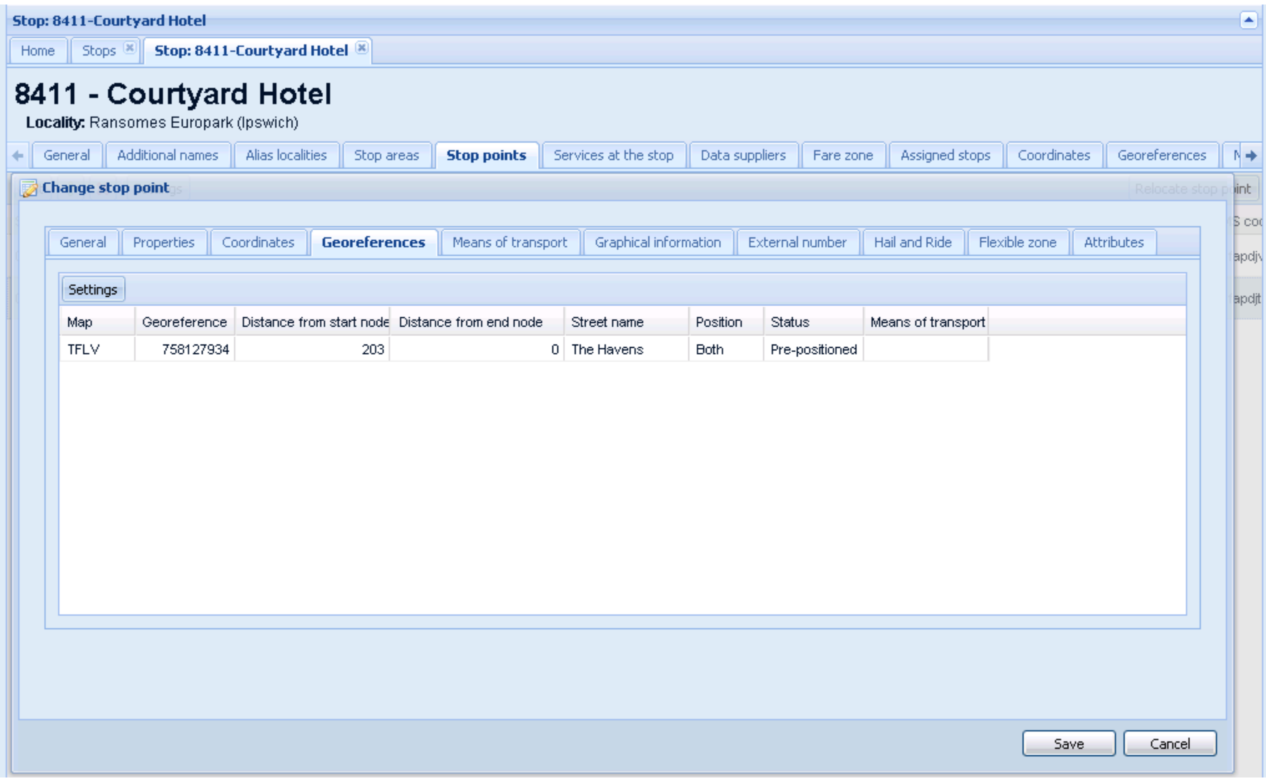


Figure 190 — Stop Point projected on a GIS segment

For NeTEx, SERVICE PATTERNs are projected on the infrastructure layer using POINT projections and LINK projections. The start and end points of the service pattern links are projected on an INFRASTRUCTURE LINK using POINT PROJECTION with TYPE OF PROJECTION « Point to Link ». The SERVICE PATTERN LINKs are projected using LINK PROJECTION.

7.6.9.2 Generic Projection – Physical Model

The Physical Model is similar to the Conceptual MODEL.

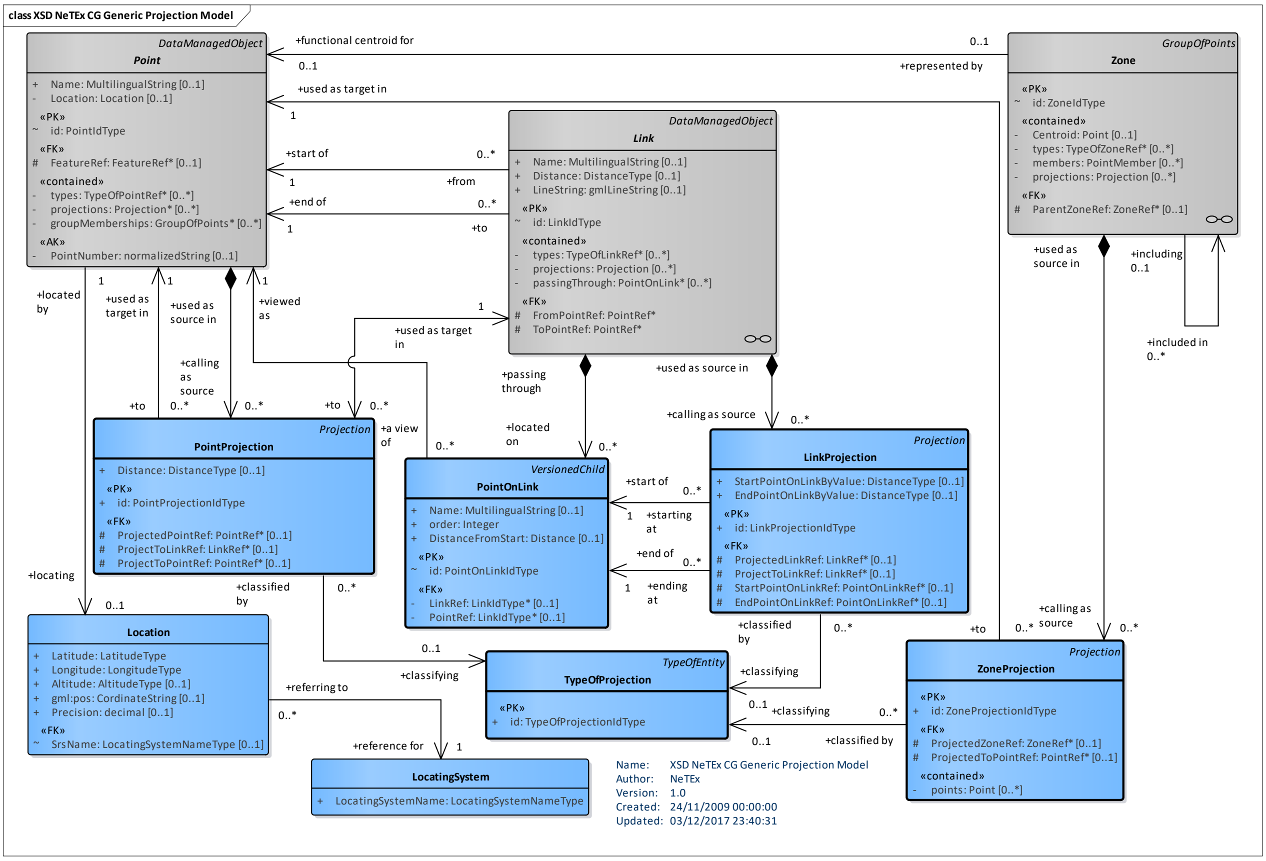


Figure 191 — Generic Projection – Physical Model (UML)

7.6.9.3 Projection – Attributes and XSD

7.6.9.3.1 Projection – Model Element

An oriented correspondence - of the shape of an ENTITY on a source layer - onto an entity in a target layer, e.g. POINT, LINK, LINK SEQUENCE, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

Table 123 — *Projection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	PROJECTION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ProjectionIdType*	1:1	Identifier of PROJECTION.
«FK»	***TypeOfProjection-Ref***	*TypeOfProjection-Ref*	0:1	Type of PROJECTION.
	Name	*xsd:normalizedString*	0:1	Description of PROJECTION.
«FK»	***SpatialFeatureRef***	*SpatialFeatureRef*		Spatial Feature projected by this PROJECTION.

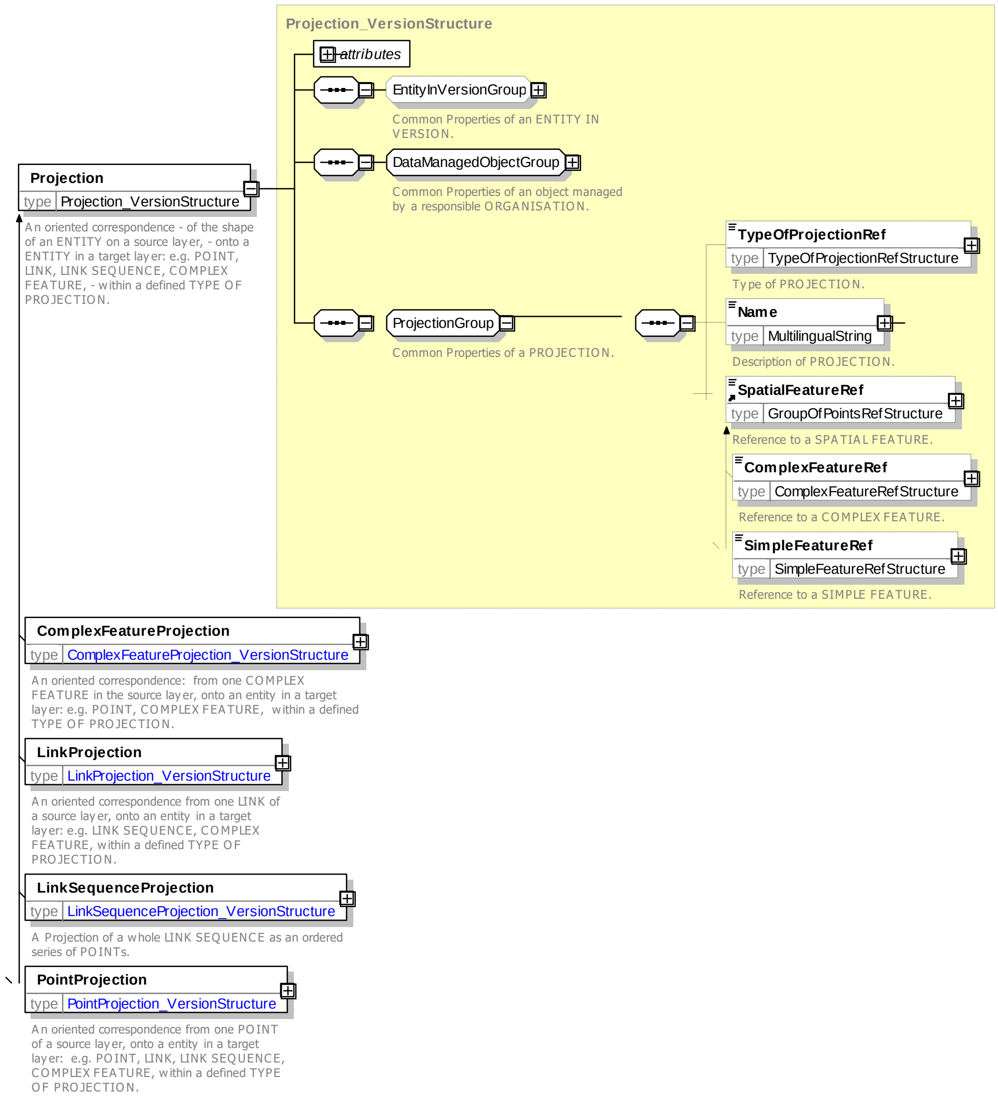


Figure 192 — *Projection – XSD*

7.6.9.3.2 PointProjection – Model Element

An oriented correspondence - from one POINT of a source layer - onto an entity in a target layer: e.g. POINT, LINK, LINK SEQUENCE, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

Table 124 — *PointProjection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Projection*</u>	::>	POINT PROJECTION inherits from PROJECTION.
«PK»	***id***	*PointProjectionIdType*	1:1	Identifier of POINT PROJECTION.
«FK»	***ProjectedPointRef***	(*PointRef*)	0:1	POINT that is being projected.
«FK»	***ProjectToPointRef***	(*PointRef*)	0:1	POINT onto which point is being projected.
«FK»	***ProjectToLinkRef***	(*LinkRef*)	0:1	LINK on which point is being projected.

	Distance	*LengthType*	1:1	Distance along projection.

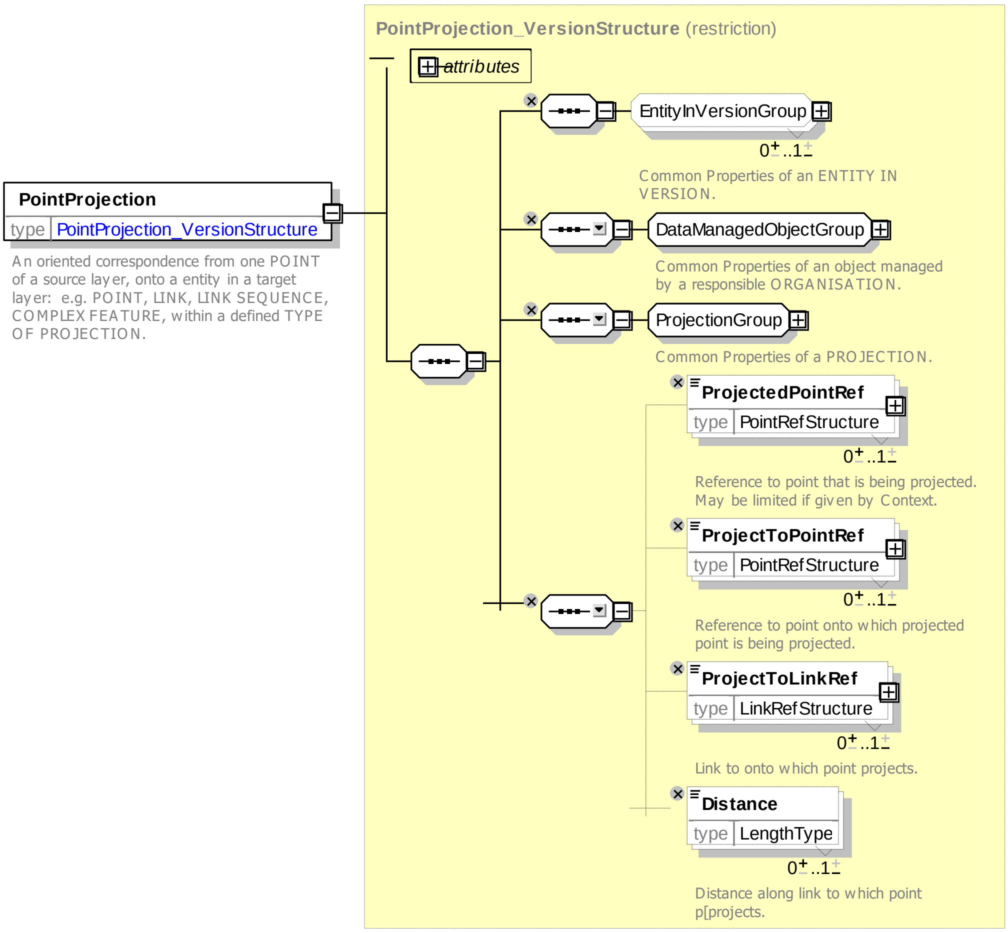


Figure 193 — *PointProjection – XSD*

7.6.9.3.3 LinkProjection – Model Element

An oriented correspondence - from one LINK of a source layer - onto an entity in a target layer: e.g. LINK SEQUENCE, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

Table 125 — *LinkProjection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Projection*	::>	LINK PROJECTION inherits from PROJECTION.
«PK»	***id***	*LinkProjectionIdType*	1:1	Identifier of LINK PROJECTION.
«FK»	***ProjectedLinkRef***	*(LinkRef)*	0:1	LINK that is being projected.
«FK»	***ProjectToLinkRef***	*(LinkRef)*	0:1	LINK on which LINK is being projected.
«FK»	***StartPoinOnLink***	*PointOnLink*	1:1	Start POINT ON LINK for PROJECTION.

«FK»	***EndPointOnLink***	*PointOnLink*	1:1	End POINT ON LINK for PROJECTION.

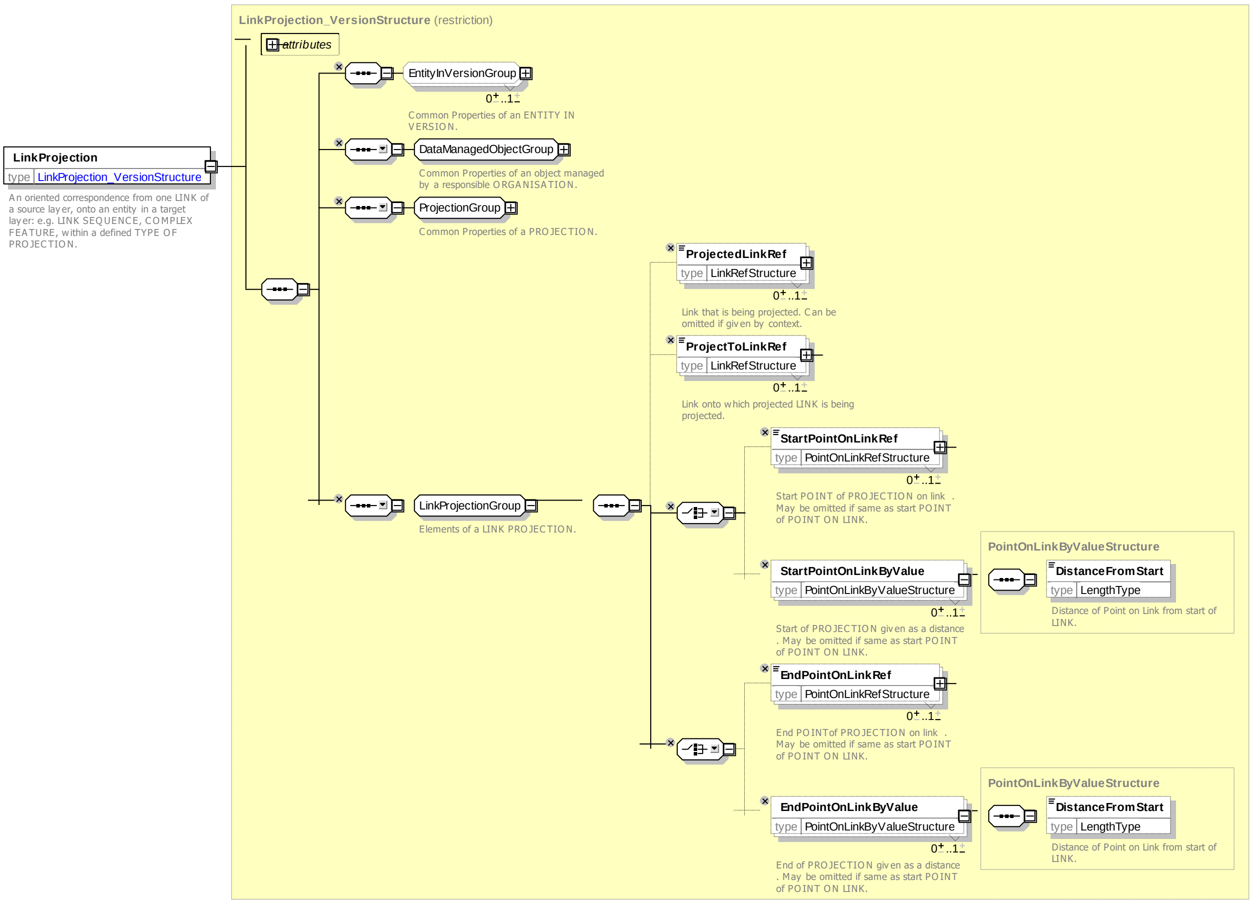


Figure 194 — *LinkProjection – XSD*

7.6.9.3.4 LinkSequenceProjection – Model Element

An oriented correspondence - from one LINK of a source layer - onto an entity in a target layer: e.g. LINK SEQUENCE, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

Table 126 — *LinkSequenceProjection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Projection*</u>	::>	LINK SEQUENCE PROJECTION inherits from PROJECTION.
«PK»	***id***	*LinkSequenceProjectionRef*	1:1	Identifier of LINK SEQUENCE PROJECTION.
«FK»	***ProjectedLink-SequenceRef***	(*LinkSequenceRef*)	0:1	LINK SEQUENCE that is being projected.
	Distance	*DistanceType*	1:1	Length of line along LINKs in LINK SEQUENCE.
		CHOICE		

«cntd»	a	***points***	*(PointRef)*	0:*	Points on PROJECTION.
«cntd»	b	***LineString***	*gml:LineString*	0:1	Coordinates of PROJECTION.

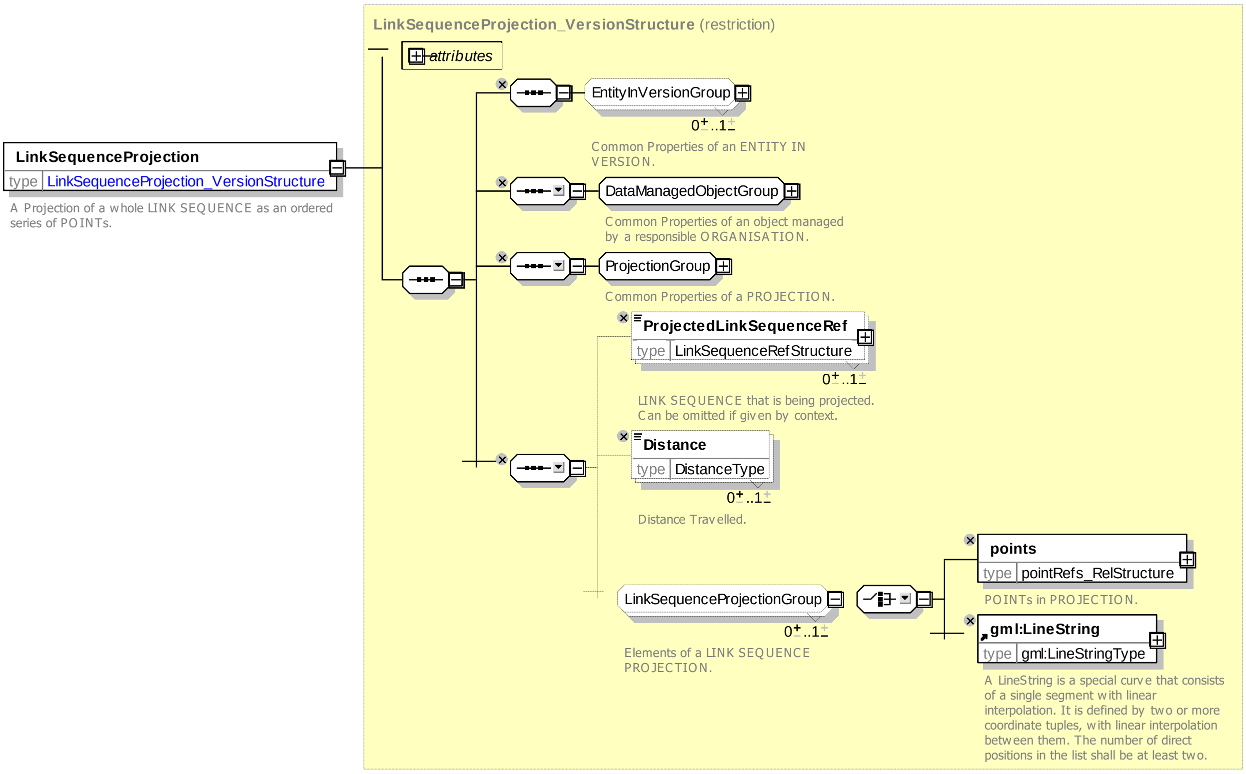


Figure 195 — *LinkSequenceProjection – XSD*

7.6.9.3.5 ZoneProjection – Model Element

An oriented correspondence: - from one ZONE in a source layer - onto a target entity : e.g. POINT, COMPLEX FEATURE, within a defined TYPE OF PROJECTION.

Table 127 — *ZoneProjection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Projection*	::>	ZONE PROJECTION inherits from PROJECTION.
«PK»	***id***	*ZoneProjection-IdType*	1:1	Identifier of ZONE PROJECTION.
«FK»	***ProjectedZone-Ref***	(*ZoneRef*)	0:1	ZONE that is being projected.
«FK»	***ProjectToPoint-Ref***	(*LinkRef*)	0:1	Centre Point of Zone onto which ZONE is being projected.
«cntd»	***points***	*(PointRef)*	0:*	Points making up PROJECTION.

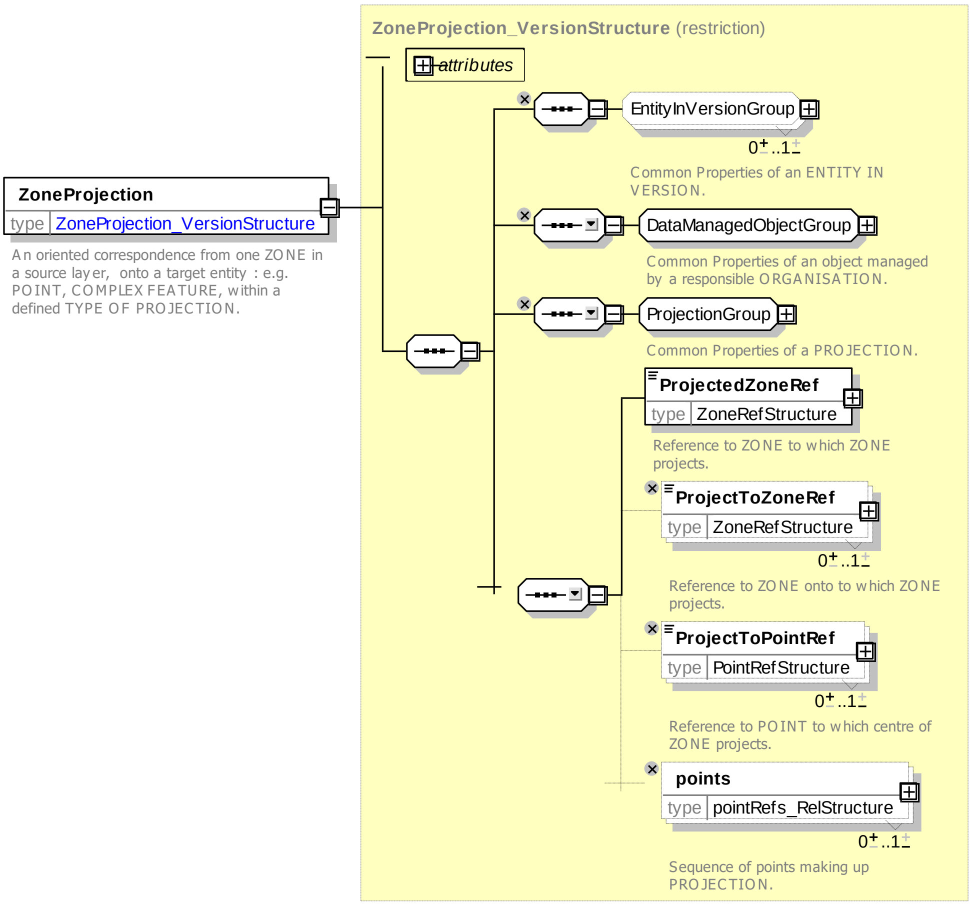


Figure 196 — *ZoneProjection – XSD*

7.6.9.3.6 Complex Feature Projection – Model Element

An oriented correspondence: - from one COMPLEX FEATURE in the source layer, - onto an entity in a target layer: e.g. POINT, COMPLEX FEATURE, - within a defined TYPE OF PROJECTION.

Table 128 — *ComplexFeatureProjection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	Projection	::>	COMPLEX FEATURE PROJECTION inherits from PROJECTION.
«PK»	***id***	*ComplexFeature-ProjectionIdType*	1:1	Identifier of COMPLEX FEATURE PROJECTION.
«FK»	***ProjectedFeatureRef***	*ComplexFeatureRef*	0:1	Reference to a COMPLEX FEATURE which is being projected.
«FK»	***OntoFeatureRef***	*ComplexFeatureRef*	0:1	Reference to a COMPLEX FEATURE onto which projection is being made.
«FK»	***PointRef***	*(PointRef)*	0:1	Reference to a POINT onto which feature is projected.

«cntd»	***features***	*FeatureRef*	0:*	Reference to FEATUREs onto which feature is projected.

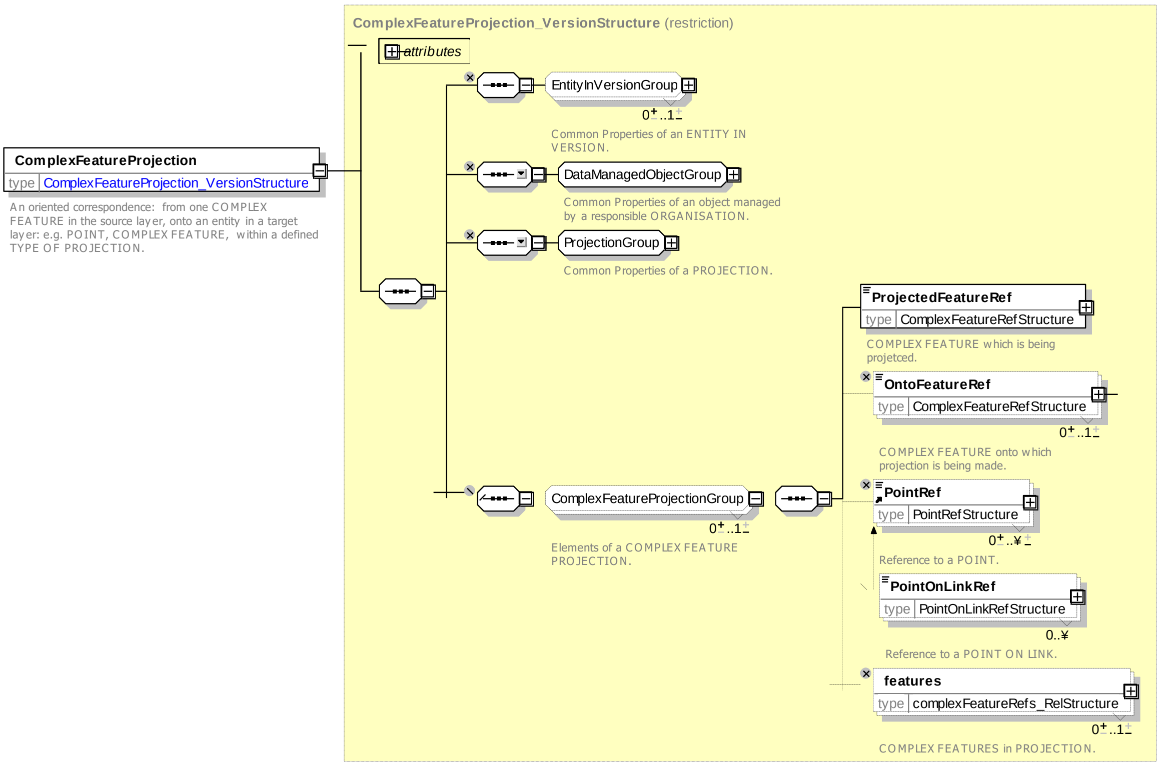


Figure 197 — *ComplexFeatureProjection – XSD*

7.6.9.3.7 LineShape – Model Element

The graphical shape of a LINK obtained from a formula or other means, using the LOCATION of its limiting POINTs and depending on the LOCATING SYSTEM used for the graphical representation.

Table 129 — *LineShape – Element*

Classification	Name	Type	Cardinality	Description
	Formula	*Name*	1:1	Formula for calculating line e.g. Bezier.
	Name	*normalizedString*	0:1	Name of LINE SHAPE.
«FK»	***LocatingSystem Ref***	*SrsNameType*	1:1	LOCATING SYSTEM for coordinates of LINE SHAPE.
«FK»	***LinkRef***	*(LinkRef)*	1:1	Link associated with LINE SHAPE.

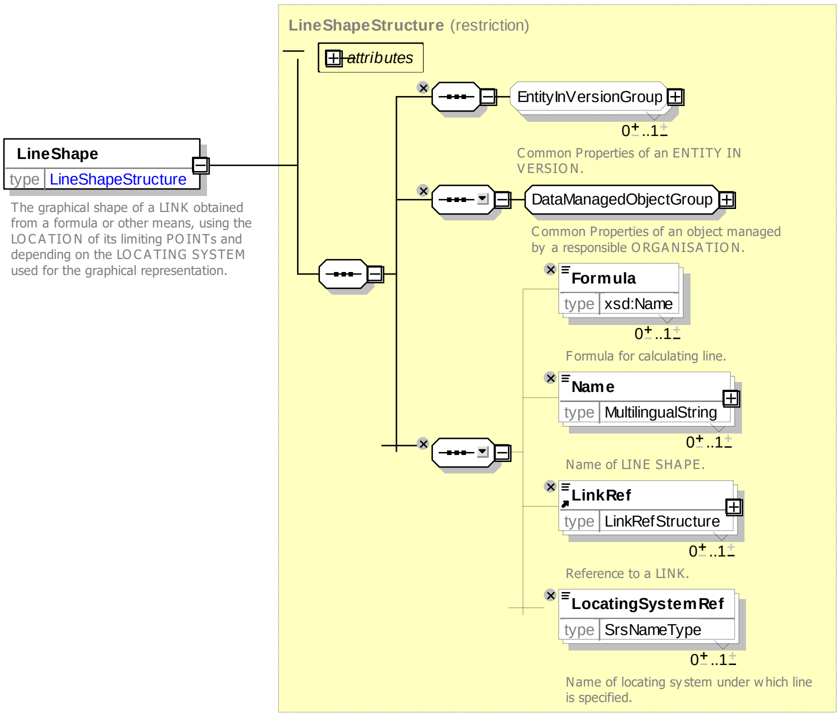


Figure 198 — *LineShape – XSD*

7.6.9.3.8 TypeOfProjection – Model Element

A classification of a PROJECTION according to its functional purpose, e.g. the source and target layers.

Table 130 — *TypeOfProjection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF PROJECTION inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfProjectionIdType*	1:1	Identifier of TYPE OF PROJECTION.

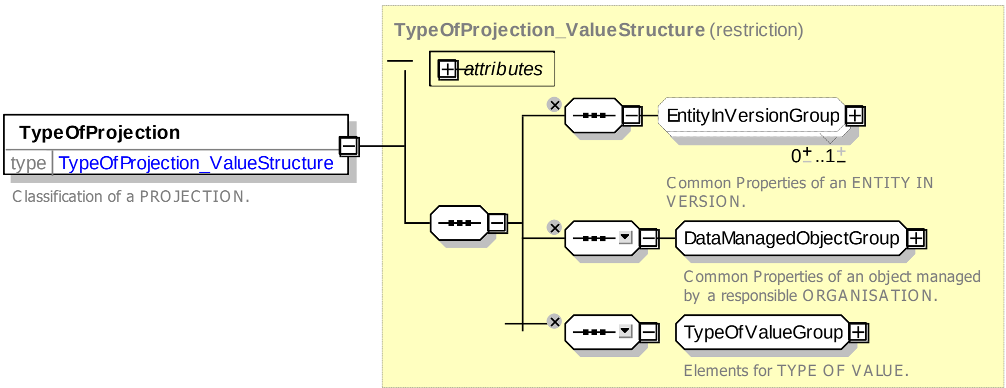


Figure 199 — *TypeOfProjection – XSD*

7.6.9.4 XML Examples of Projection

7.6.9.4.1 Point Projection – XML Fragment Example

The following XML fragment shows a POINT PROJECTION of a POINT onto a distance along a link

EXAMPLE

```xml
<ScheduledStopPoint version="001" created="2001-06-17T09:30:47.0Z" id="ssp:SSP4">
        <Name>Stop 4</Name>
        <Location>
            <Longitude>-180</Longitude>
            <Latitude>+53</Latitude>
        </Location>
        <projections>
            <PointProjection version="any" id="ssp:LPJ_SSP4_pj01">
                <ProjectToLinkRef version="any" ref="spq:RL_Rt001o_05to06"/>
                <Distance>25</Distance>
            </PointProjection>
    </projections>
    <Label>Towards SSSP9</Label>
</ScheduledStopPoint>
```

7.6.9.4.2 Link Projection – XML Fragment Example

The following XML fragment shows a LINK PROJECTION of a SERVICE LINK ontothree other links.

EXAMPLE

```xml
    <ServiceLink version="001" created="2010-12-17T09:30:47.0Z"
id="ssp:ServiceLink:SL_SvP001o_2to3">
        <Name>2 to 3 </Name>
        <projections>
            <LinkProjection version="any" id="ssp::LP_SL_SvP001o_2+3_PJ01" order="1">
                <Name>Projection mid way along link SL 2-3::>RL2-3 x </Name>
                <ProjectToLinkRef version="any" ref="spq::RL_Rt001o_02++03"/>
                <StartPointOnLink>
                    <DistanceFromStart>40</DistanceFromStart>
                </StartPointOnLink>
            </LinkProjection>
            <LinkProjection version="any" id="ssp::LP_SL_SvP001o_2+3_PJ02" order="2">
                <Name>Projection whole link SL 2-3::>RL 3-4 x </Name>
                <ProjectToLinkRef version="any" ref="spq::RL_Rt001o_03+04"/>
            </LinkProjection>
            <LinkProjection version="any" id="ssp::LP_SL_SvP001o_2+3_PJ03" order="3">
                <Name>Projection part link SL 2-3::>RL 4-5 x </Name>
                <ProjectToLinkRef version="any" ref="spq::RL_Rt001o_04+05"/>
                <EndPointOnLink>
                    <DistanceFromStart>35</DistanceFromStart>
                </EndPointOnLink>
            </LinkProjection>
        </projections>
        <FromPointRef version="001" ref="ssp:SP2"/>
        <ToPointRef version="001" ref="ssp:SSP3"/>
    </ServiceLink>
```

7.6.9.4.3 Zone Projection – XML Fragment Example

The following XML fragment shows a simple ZONE PROJECTION of a QUAY onto another ZONE.

EXAMPLE

```xml
<Quay version="any" id="noptis:9025001000002221">
    <Name>1</Name>
    <Centroid>
        <Location srsName="WGS84">
            <Longitude>18.0717945909683</Longitude>
            <Latitude>59.3454833520777</Latitude>
        </Location>
    </Centroid>
```

```
        <projections>
            <ZoneProjection version="any" id="noptis:9025001000002221-01">
                <ProjectedZoneRef ref="noptis:9081001110000017"/>
            </ZoneProjection>
        </projections>
    </Quay>
```

7.6.10 Generic Place

7.6.10.1 Generic PLACE – Conceptual MODEL

The PLACE model defines topographically significant places that a transport model may wish to describe. It also allows the description of the possibility of connecting between them. A PLACE may be of dimension 0 (a POINT), 1 (a road section) or 2 (a ZONE).

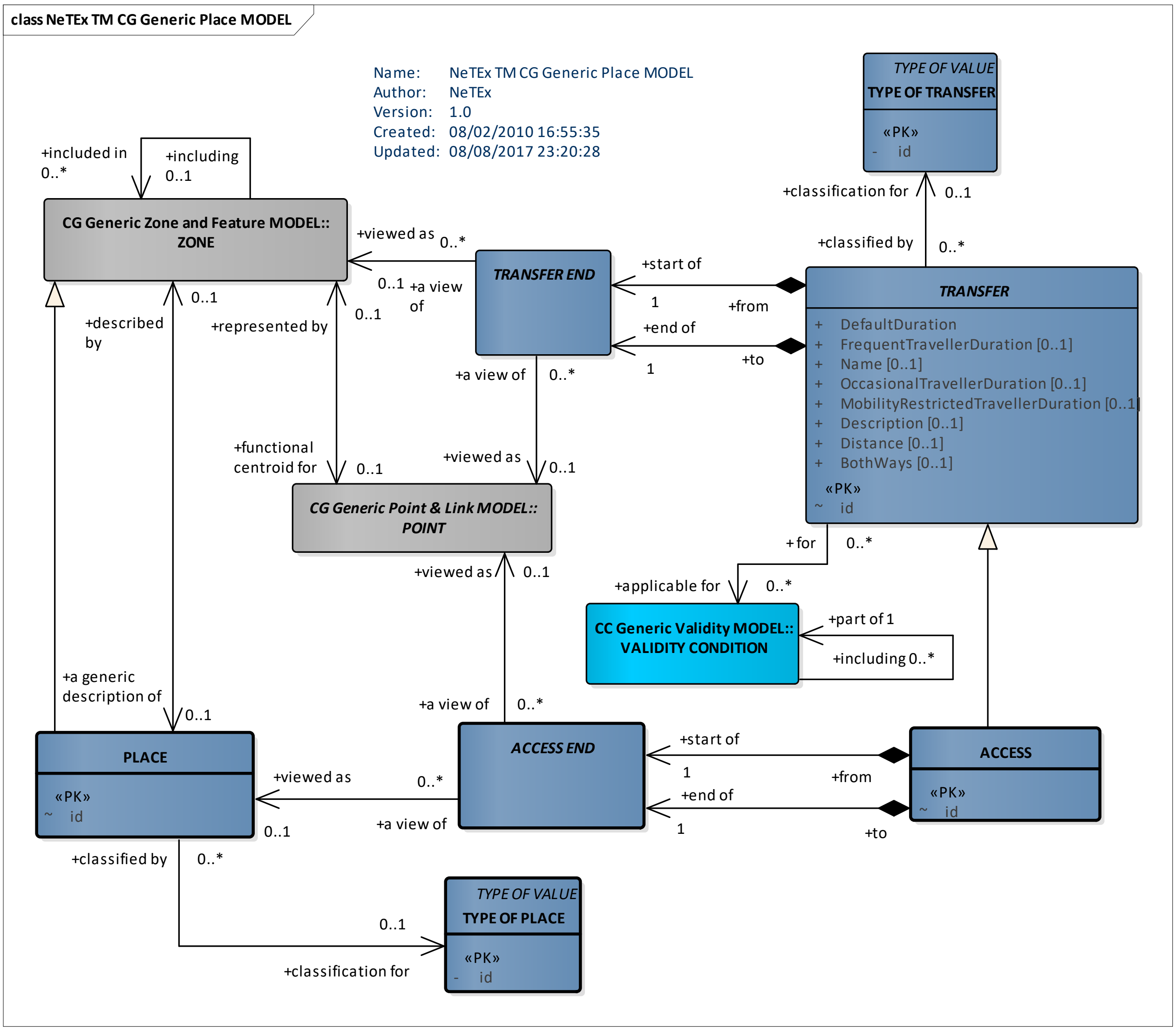


Figure 200 — Generic Place – Conceptual MODEL (UML)

A PLACE can be specialised in a ADDRESSABLE PLACE whenever it refers to a place having an address either a ROAD ADDRESS, or a POSTAL ADDRESS (see 7.7.7 - Topographic Place and specially).

7.6.10.2 Generic Place – Physical Model

The PLACE Physical Model provides abstract classes for PLACE and TRANSFER and a concrete specialisation of TRANSFER: the ACCESS link. To simplify the implementation, PLACE is made a type of ZONE, even though in the conceptual model they are separate concerns. (i.e. a multiple inheritance model in the conceptual model is coerced to a single inheritance hierarchy in the implementation.)

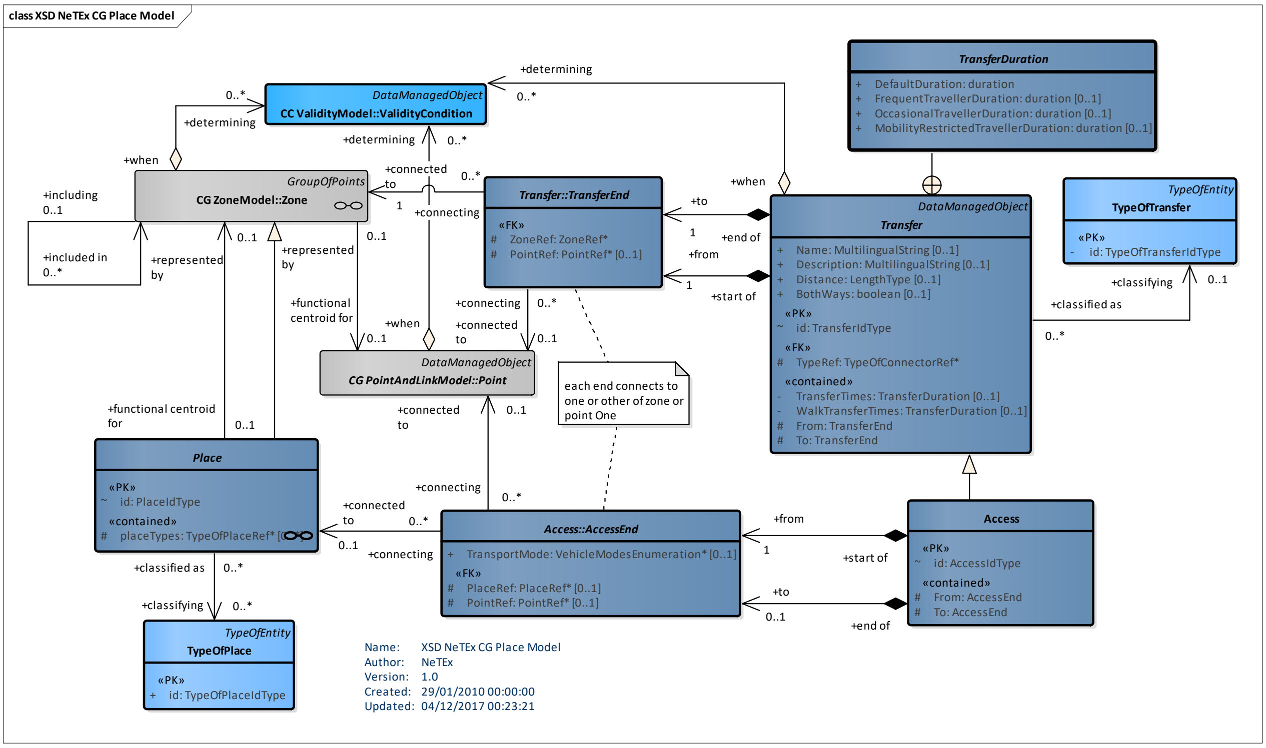


Figure 201 — Generic Place – Physical Model (UML)

7.6.10.2.1 Explicit Places – Physical Model

The following figure summarises the explicit specialisations of PLACE used in NeTEx.

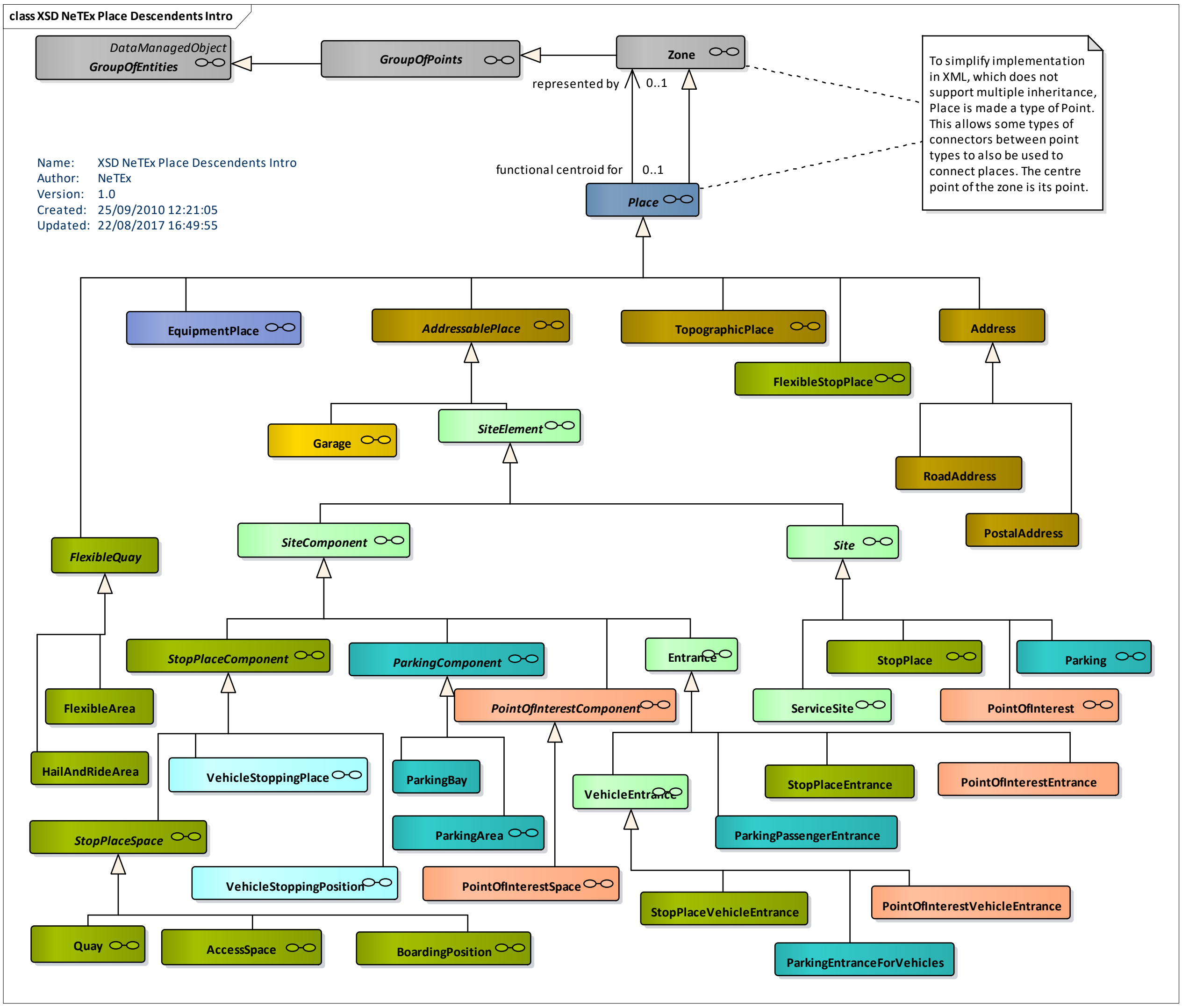


Figure 202 — Generic place – Physical Model (UML)

7.6.10.3 Generic Place – Attributes and XSD

7.6.10.3.1 Place – Model Element

A geographic place of any type which may be specified as the origin or destination of a trip. A PLACE may be of dimension 0 (a POINT), 1 (a road section) or 2 (a ZONE).

Table 131 — *Place – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Zone*	::>	PLACE inherits from ZONE.
«PK»	***id***	*PlaceIdType*	1:1	Identifier of PLACE.
«cntd»	***placeTypes***	*TypeOfPlaceRef*	0:*	Types of PLACE.

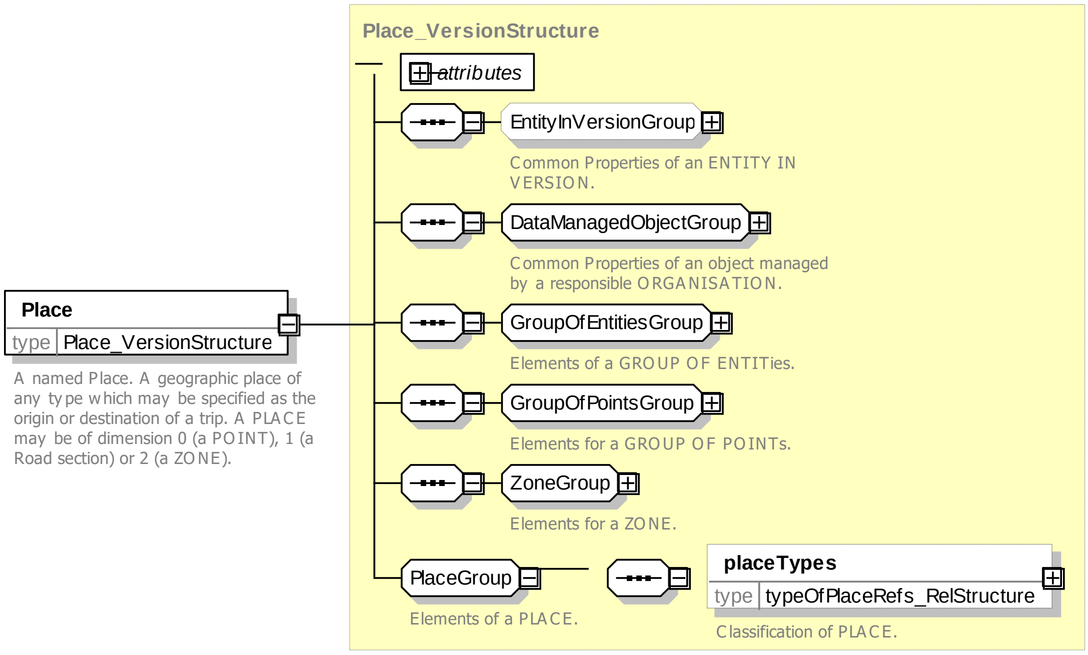


Figure 203 — *Place – XSD*

7.6.10.3.2 Transfer – Model Element

A couple of POINTs located sufficiently near together that they offer a passenger the possibility to reach one of these POINTs when starting at the other one in a timescale which is realistic when carrying out a trip. A TRANSFER may be specialised as CONNECTION (a transfer within a STOP AREA) or an ACCESS LINK (a longer TRANSFER between PLACEs).

Table 132 — *Transfer – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	TRANSFER inherits from DATA MANAGED OBJECT.
«PK»	***id***	*TransferIdType*	1:1	Identifier of TRANSFER.
	Name	*MultilingualString*	0:1	Name of TRANSFER.
«FK»	***TypeOfTransfer-Ref***	*TypeOfTransferRef*	1:1	Reference to identifier of a TYPE OF TRANSFER.
	Description	*MultilingualString*	0:1	Description of TRANSFER.
	Distance	*DistanceType*	0:1	Distance of TRANSFER link.
«cntd»	***TransferDuration***	<u>*TransferDuration*</u>	0:1	Scheduling TRANSFER TIMEs that apply to TRANSFER.
«cntd»	***WalkTransferDuration***	<u>*TransferDuration*</u>	0:1	Walking TRANSFER TIMEs that apply to TRANSFER.

	BothWays	*xsd:boolean*	0:1	Whether TRANSFER can be traversed in both directions.
«cntd»	(***From***)	*TransferEnd*	1:1	Origin end of TRANSFER. Specialised in subclasse.
«cntd»	(***To***)	*TransferEnd*	1:1	Destination end of TRANSFER. Specialised in subclass.

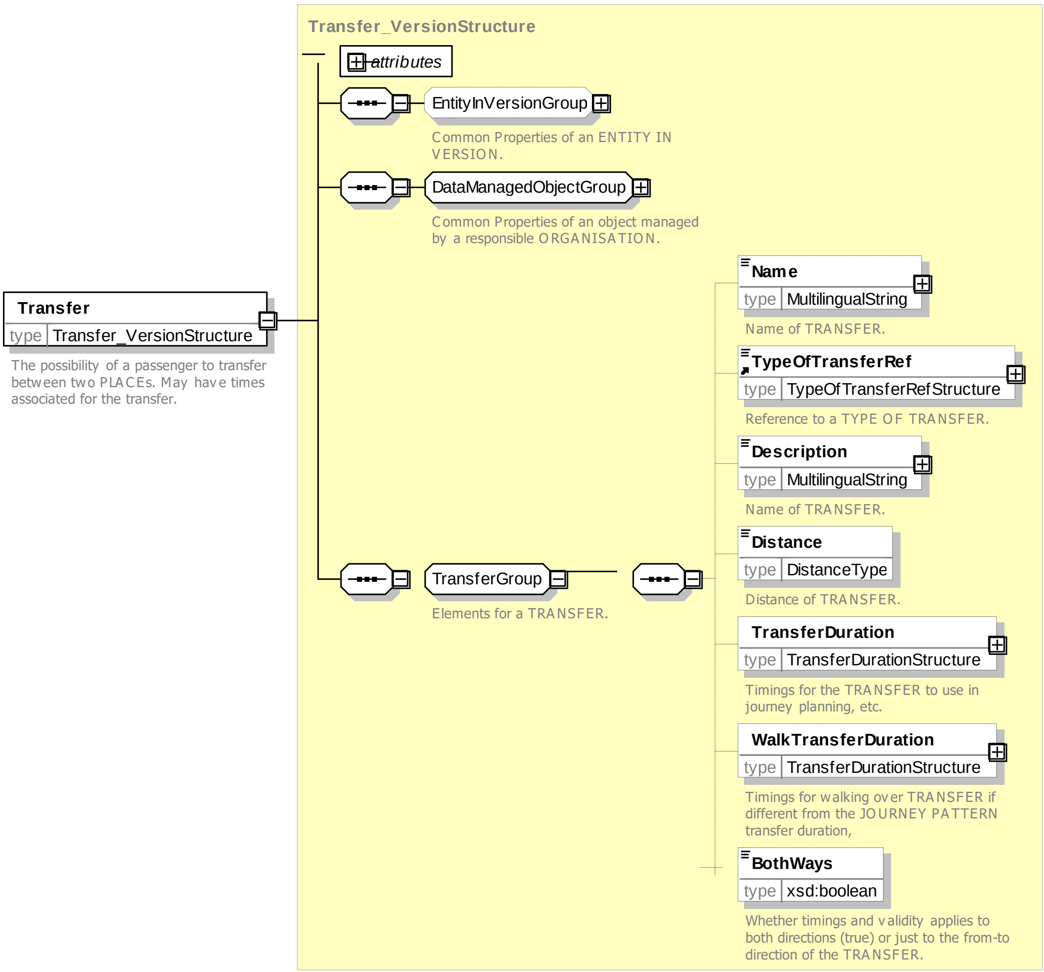


Figure 204 — *Transfer – XSD*

7.6.10.3.3 TransferEnd – Model Element

End point of a TRANSFER.

Table 133 — *TransferEnd – Element*

Classification	Name	Type	Cardinality	Description
«FK»	***ZoneRef***	*(ZoneRef)*	1:1	Reference to ZONE of end point of TRANSFER.
«FK»	***PointRef***	*(PointRef)*	0:1	Reference to origin or destination point of TRANSFER.

Transfer is abstract – Transfer end is implemented as a concrete specialisation, e.g. *AccessEnd*

7.6.10.3.4 TransferDuration – Model Element

Times taken to make a TRANSFER.

Table 134 — *TransferDuration – Structure*

Classification	Name	Type	Cardinality	Description
	DefaultDuration	*xsd:duration*	1:1	Default time needed to make transfer.
	FrequentTraveller-Duration	*xsd:duration*	0:1	Time for a traveller familiar with the journey to make transfer.
	OccasionalTraveller-Duration	*xsd:duration*	0:1	Time for a traveller unfamiliar with the journey to make transfer.
	MobilityRestricted-TravellerDuration	*xsd:duration*	0:1	Time for a mobility impaired traveller familiar to make transfer.

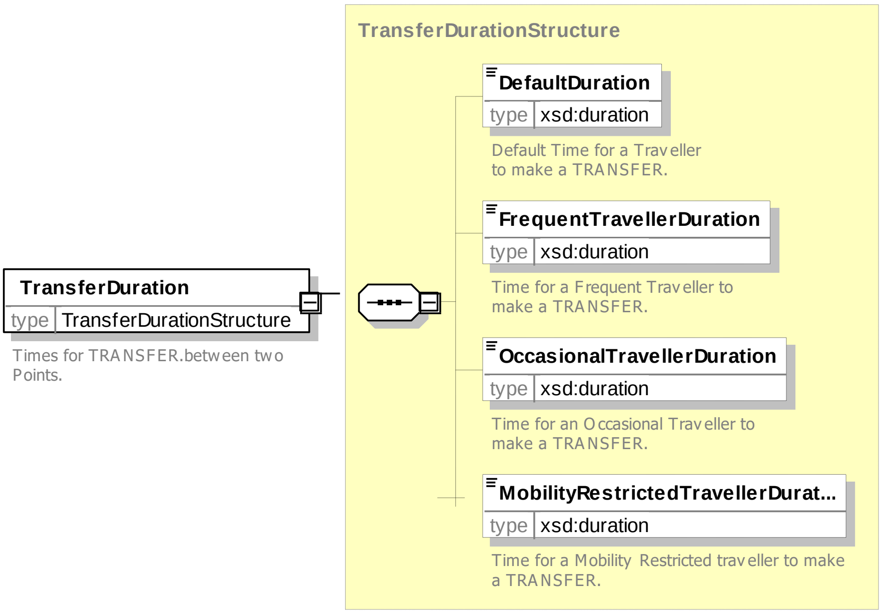


Figure 205 — Transfert Duration XSD

7.6.10.3.5 TypeOfPlace – Model Element

A Classification for PLACEs.

Table 135 — *TypeOfPlace – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*TypeOfEntity*</u>	::>	TYPE OF PLACE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfPlaceIdType*	1:1	Identifier of TYPE OF PLACE.

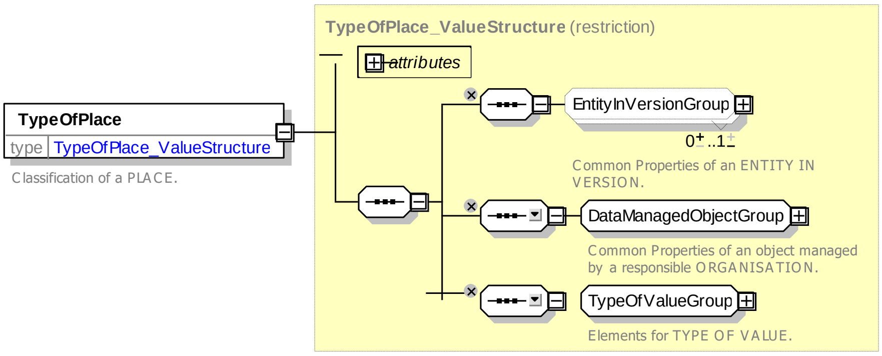


Figure 206 — *TypeOfPlace – XSD*

7.6.10.3.6 TypeOfTransfer – Model Element

Classification of a TRANSFER.

Table 136 — *TypeOfTransfer – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF TRANSFER inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfTransferIdType*	1:1	Identifier of TYPE OF TRANSFER.

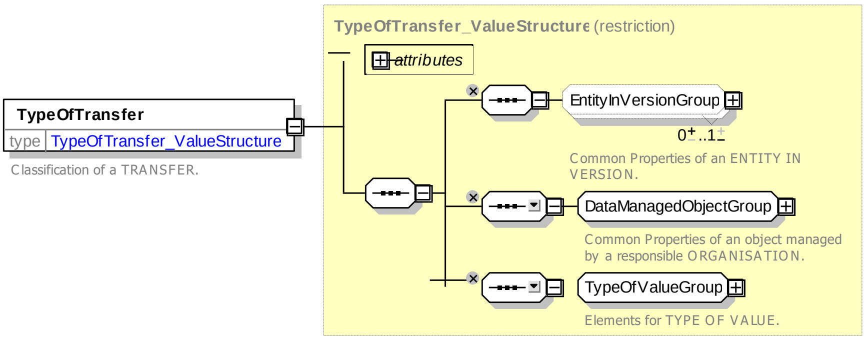


Figure 207 — *TypeOfTransfer – XSD*

7.6.10.3.7 Access – Model Element

The physical (spatial) possibility for a passenger to access or leave the public transport system. This link may be used during a trip for:

- the walking movement of a passenger from a PLACE (origin of the trip) to a STOP POINT (origin of the PT TRIP),

- the walking movement from a SCHEDULED STOP POINT (destination of the PT TRIP) to a PLACE (destination of the trip).

Table 137 — *Access – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	*Link*	::>	ACCESS link inherits from LINK.
«PK»	***id***	*AccessIdType*	1:1	Identifier of ACCESS link.
«cntd»	***From***	*AccessEnd*	1:1	Reference to a point at which ACCESS link starts.
«cntd»	***To***	*AccessEnd*	1:1	Reference to a point at which ACCESS link ends.

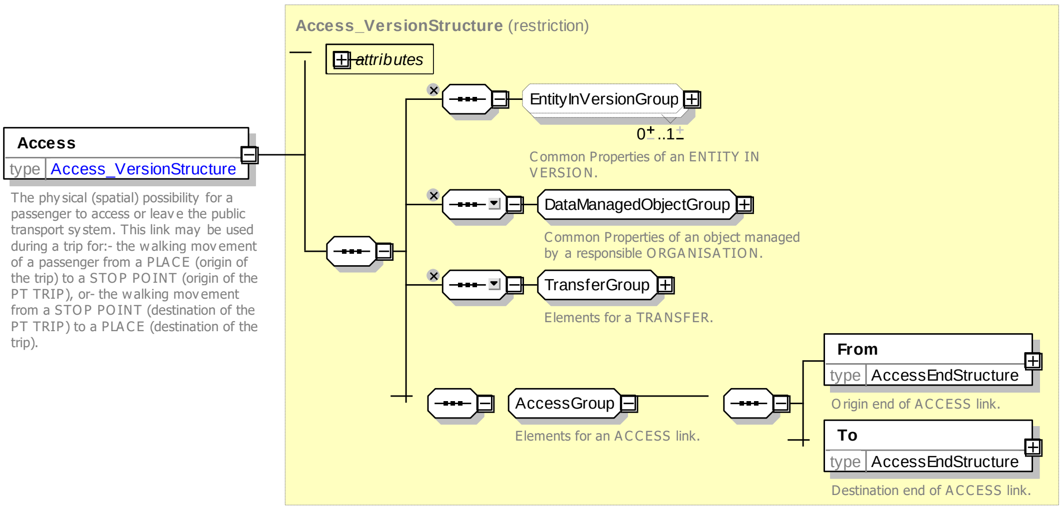


Figure 208 — *Access – XSD*

7.6.10.3.8 AccessEnd – Model Element

Origin or destination end of an ACCESS link. It may indicate a MODE, POINT and PLACE.

Table 138 — *AccessEnd – Structure*

Classification	Name	Type	Cardinality	Description
«FK»	***TransportMode***	*PlaceIdType*	0:1	Reference to Transport Mode at one end of TRANSFER.
«FK»	***PlaceRef***	(*PlaceRef*)	0:1	Reference to destination point of TRANSFER.
«FK»	***PointRef***	(*PointRef*)	0:1	Reference to origin point of TRANSFER.

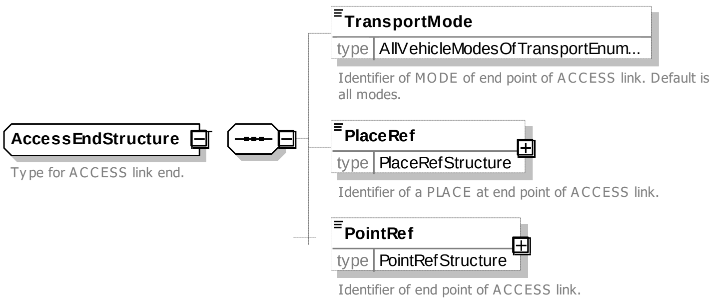


Figure 209 — *AccessEnd – XSD*

7.6.10.4 XML Examples of Place & Access

7.6.10.4.1 Access Link – XML Fragment Example

The following XML fragment shows an ACCESS link defined between a STOP PLACE (‘*napt::470012345*’) and a POINT OF INTEREST (‘*tbd::POI_23*’).

EXAMPLE

```
        <accesses>
            <Access version="any" id="tbd::POI_23_N1">
                <Name>Access to stadium from Framton metro</Name>
                <Distance>0.0</Distance>
                <TransferDuration>
                    <DefaultDuration>PT5M</DefaultDuration>
                    <FrequentTravellerDuration>PT3M</FrequentTravellerDuration>
                    <OccasionalTravellerDuration>PT5M</OccasionalTravellerDuration>

    <MobilityRestrictedTravellerDuration>PT20M</MobilityRestrictedTravellerDuration>
                </TransferDuration>
                <BothWays>true</BothWays>
                <From>
                    <TransportMode>metro</TransportMode>
                    <PlaceRef xsi:type="StopPlaceRefStructure" ref="napt::470012345"/>
                </From>
                <To>
                    <PlaceRef         version="any"         xsi:type="PointOfInterestRefStructure"
ref="tbd::POI_23"/>
                </To>
            </Access>
        </accesses>
```

NOTE Duration is encoded following XSD rules: the time interval is specified in the following form "PnYnMnDTnHnMnS" where:

- — P indicates the period (required)
- — nY indicates the number of years
- — nM indicates the number of months
- — nD indicates the number of days
- — T indicates the start of a time section (required if you are going to specify hours, minutes or seconds)
- — nH indicates the number of hours
- — nM indicates the number of minutes
- — nS indicates the number of seconds

7.6.11 Generic Assignment

7.6.11.1 Generic ASSIGNMENT – Conceptual MODEL

A common design pattern found in NeTEx is the assignment of a named set of properties to another element; there may often be more than one assignment for a given element, either with different validities, or with a specified precedence. This pattern is made explicit in Part3 element as an ASSIGNMENT; an assignment may have a name and an order.

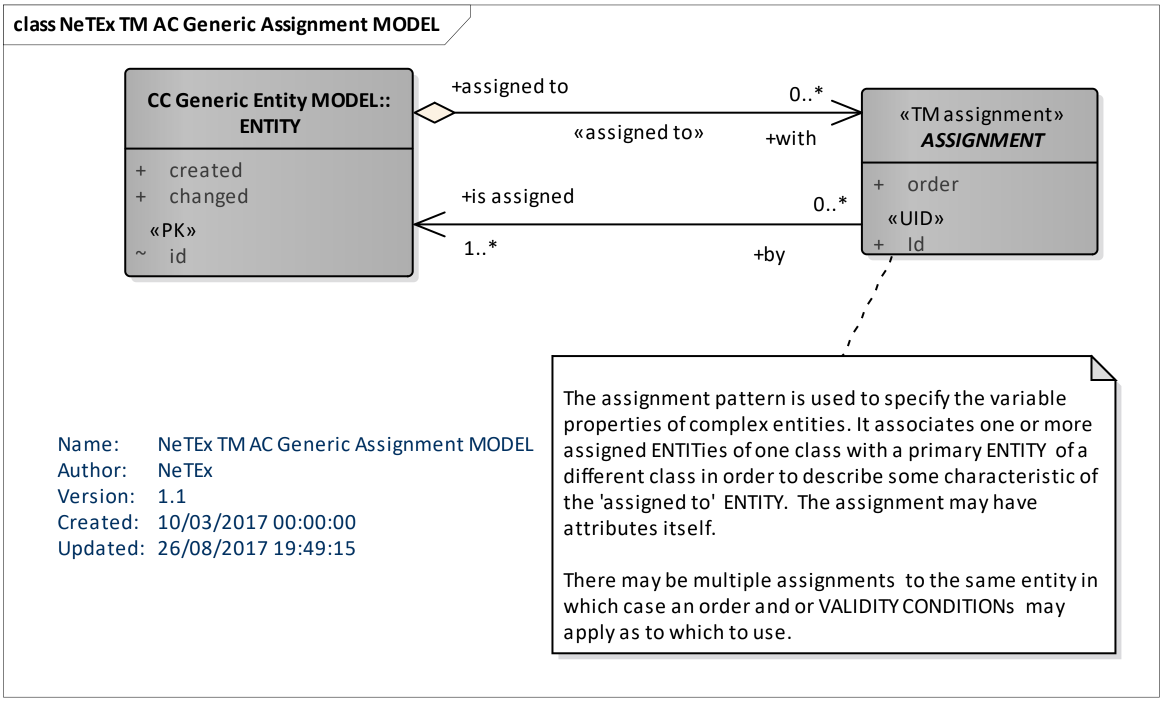


Figure 210 — Generic Assignment – Conceptual MODEL (UML)

7.6.11.2 Generic Assignment – Physical model

The following diagram shows detailed attributes of the ASSIGNMENT model. An assignment may have ***order***, ***Name*** and ***Description*** attributes. If an element is a specialisation of ASSIGNMENT it indicates that it is an ancillary element specifying dependent properties of a fundamental NeTEx element.

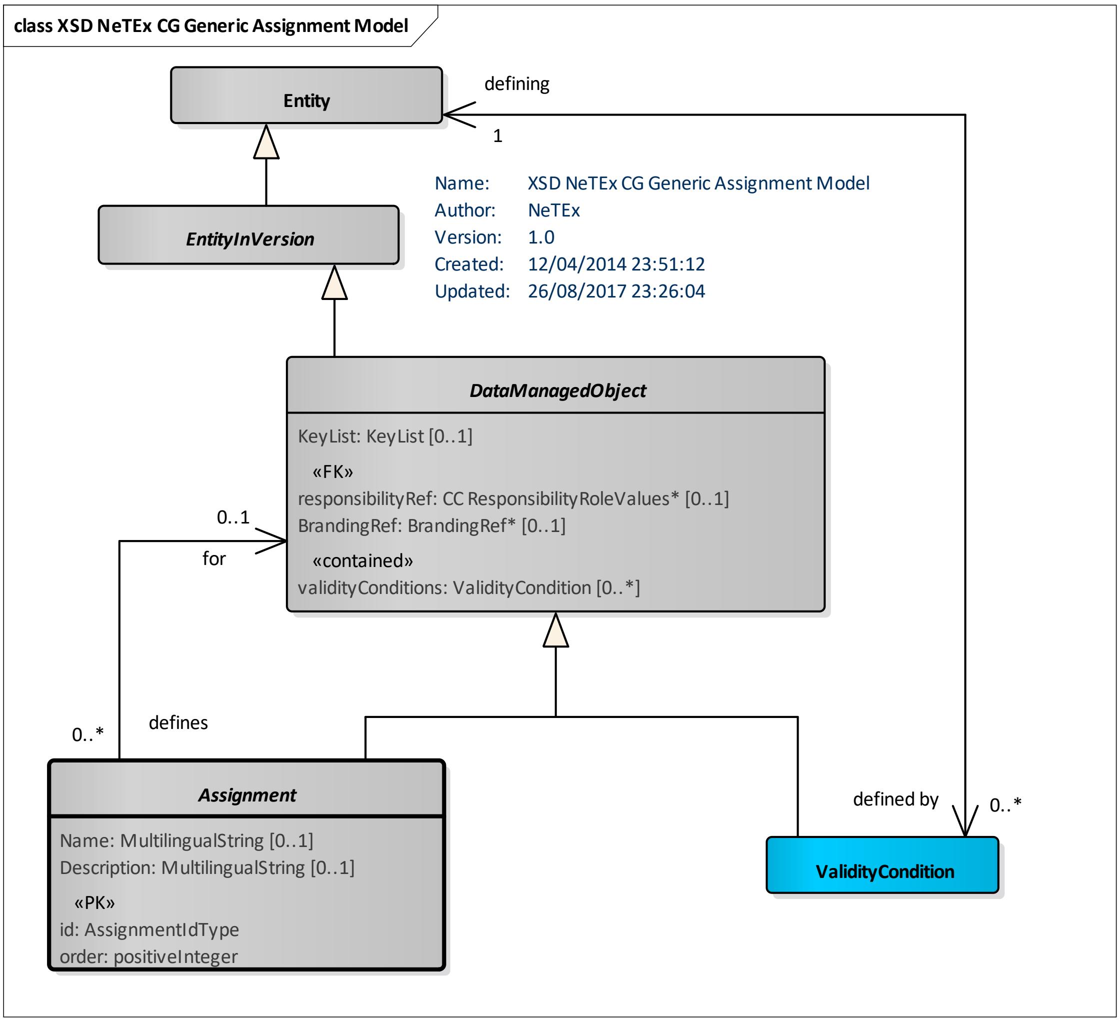


Figure 211 — Generic Assignment– Physical Model (UML)

7.6.11.3 Generic Assignment – Attributes and XSD

7.6.11.3.1 Assignment — Element

An ASSIGNMENT specifies a set of properties to be associated with another element on a temporary or permanent basis.

Table 139 – *Assignment* – Name

Classifi-cation	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	ASSIGNMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*AssignmentIdType*	1:1	Identifier of ASSIGNMENT.
«atr»	***order***	*xsd:positiveInteger*	0:1	Relative precedence of ASSIGNMENT.
	Name	*MultilingualString*	0:1	Name of ASSIGNMENT.

	Description	*MultilingualString*	0:1	Description of ASSIGNMENT.

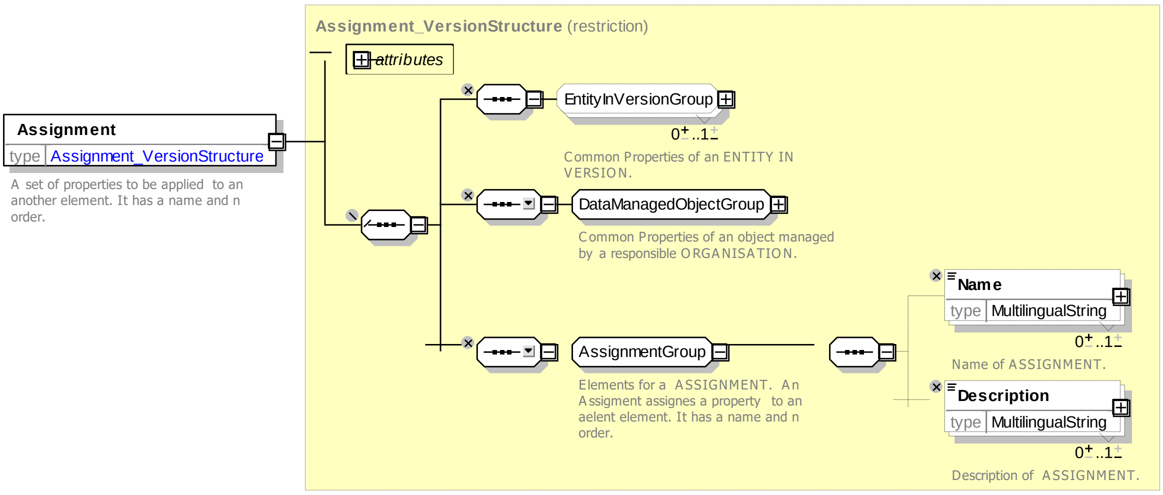


Figure 212 — *Assignment* — XSD

7.6.12 Generic Layer

Topological entities used for describing a transport network can be grouped into different LAYERs. Such sets are actually VERSION FRAMEs with a specific property, namely each LAYER is associated with a one and only one LOCATING SYSTEM, and the entities belonging to the LAYER have a position within this LOCATING SYSTEM. Examples of layers include:

- Infrastructure layer: describes road or rail network.
- Route layer: describes route topology.
- Service layer: describes network of stops served by routes.
- Timing layer: describes location of timing points and times between them.

In other words, the functional views of the network are described as layers. A projection is a mechanism enabling the description of the correspondence between the different layers. This mapping between the layers is particularly useful when spatial data from different environments (sources, functional domains) have to be combined. An example of such a situation is the mapping of the public transport network on the road network.

7.6.12.1 Generic Layer – Conceptual MODEL

The reference conceptual model for LAYERs is the following:

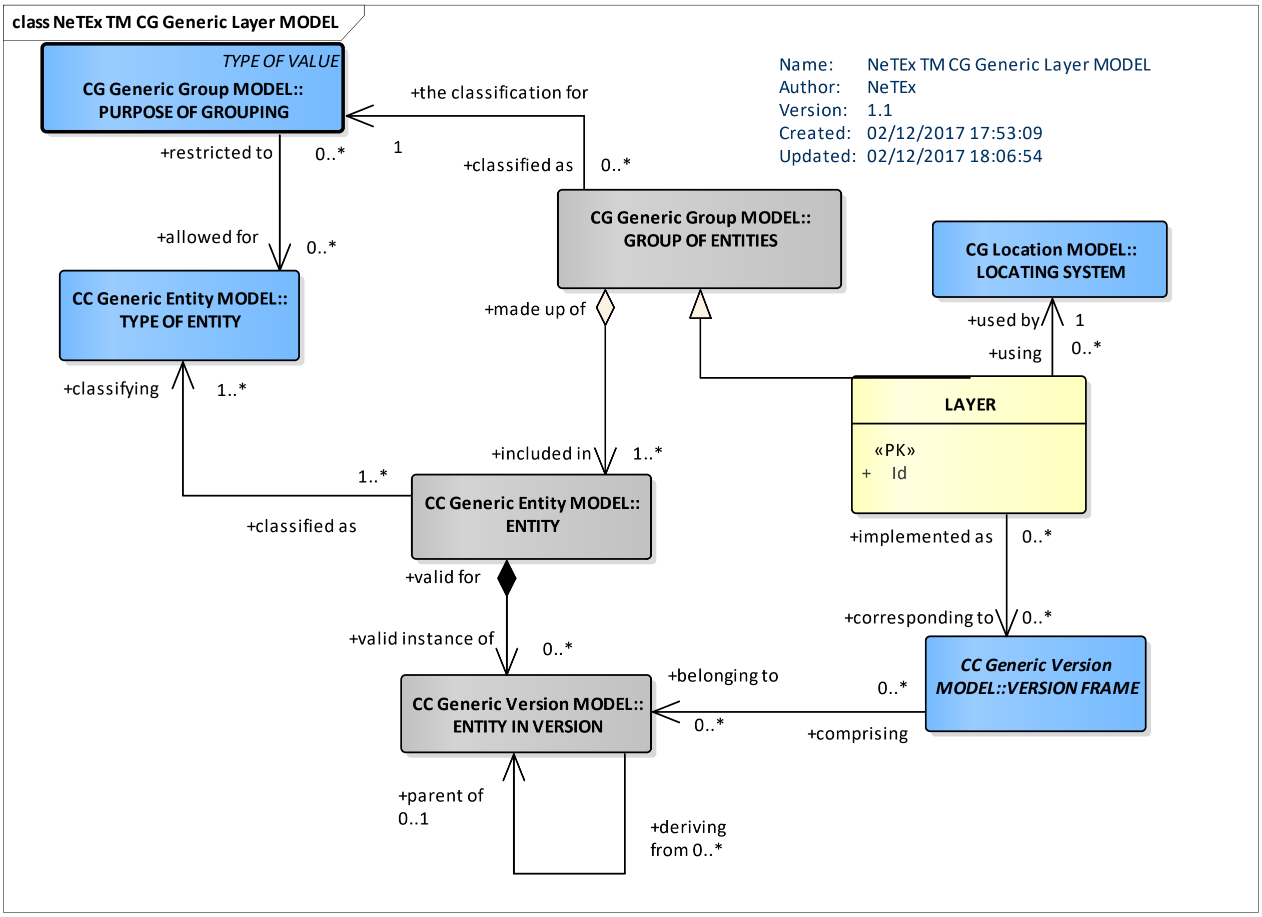


Figure 213 — Layer – Conceptual MODEL (UML)

7.6.12.2 Generic Layer – Physical model

The following diagram used by NeTEx shows detailed attributes of the LAYER model.

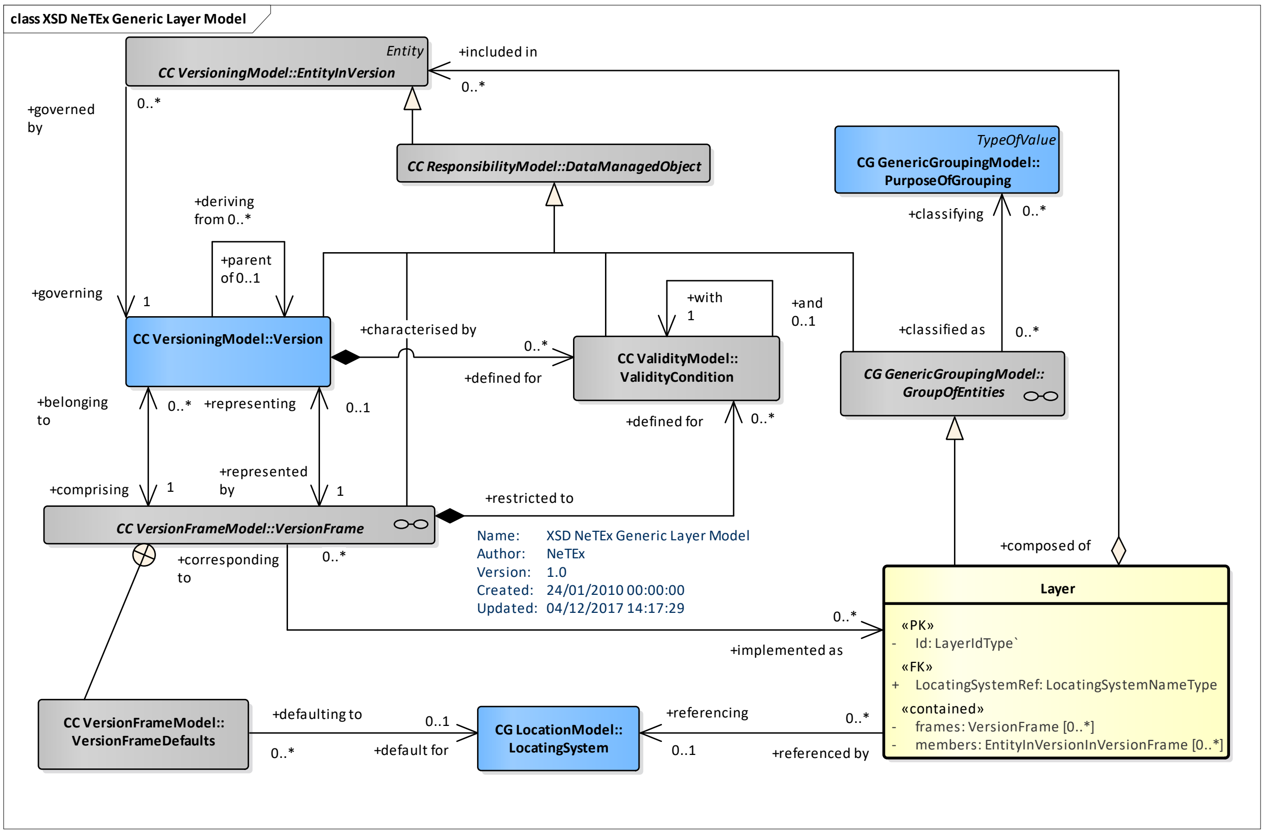


Figure 214 — Layer – Physical Model (UML)

7.6.12.3 Generic Layer – Attributes and XSD

7.6.12.3.1 Layer — Element

A LAYER is a user-defined GROUP OF ENTITies, specified for a particular functional purpose, associating data referring to a particular LOCATING SYSTEM.

Table 140 – *Layer* – Name

Classifi-cation	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	LAYER inherits from DATA MANAGED OBJECT.
	LocationSystem	*xsd:positiveInteger*	1:1	Name of location system that applies to LAYER.
“cntd»	***versionFrames***	*(VersionFrameRef)*	0:*	VERSION FRAMEs to which LAYER applies. of
“cntd»	***members***	*(ObjectRef)*	0:*	ELEMENTs belonging to LAYER.

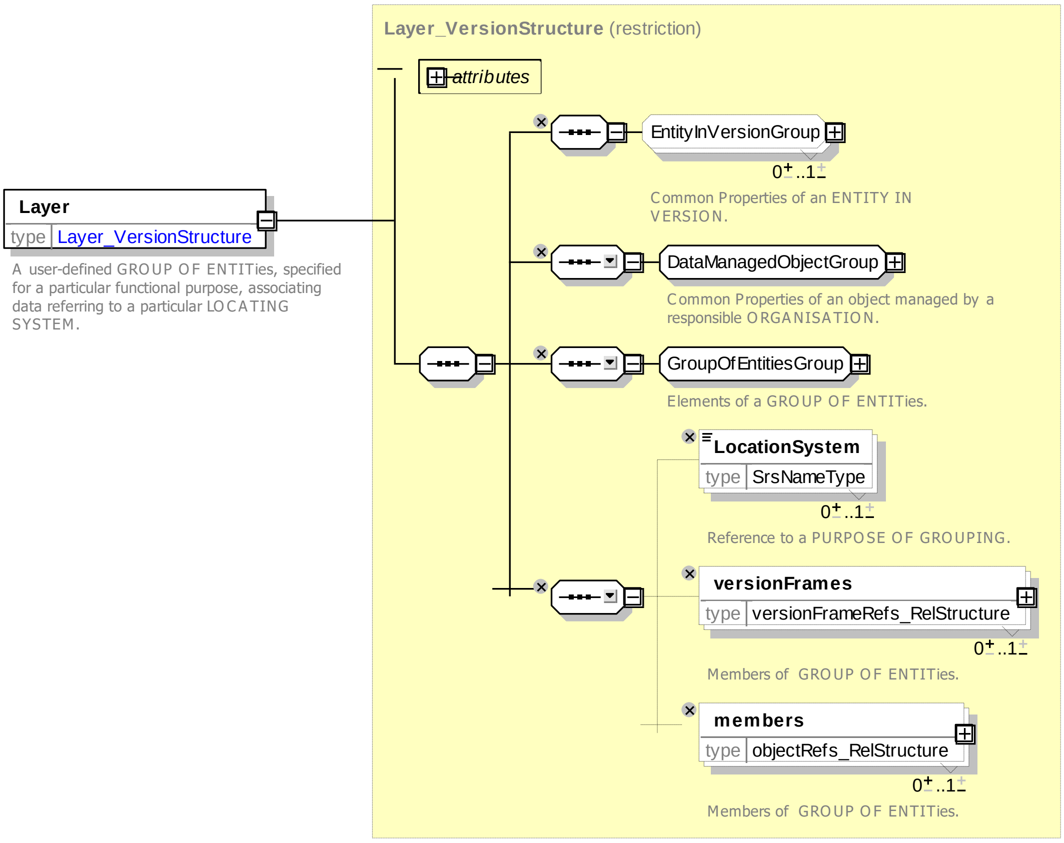


Figure 215 — *Layer* — XSD

7.6.13 Accessibility

NeTEx supports a detailed description of the accessibility of a SITE. This can be used in applications in various ways:

— For **computable** use the data can be used by a journey planner when calculating a journey that meets a given set of user criteria, for example, to choose stations or paths that are wheelchair accessible when planning a point-to-point journey and to direct a user to the entrances and exits most suitable according to their needs.

— For **browsing/navigation** use the data can be used to show the exact properties of a given interchange so that users may rehearse a trip ahead of making it and make their own judgement as to the best path through an interchange.

In order to journey plan across data from different data systems, a uniform set of summary assessment criteria is needed that can be used to establish possible routes of an equivalent level of accessibility. For example, can a path be followed in a wheelchair, without using steps, etc.

7.6.13.1 ACCESSIBILITY – Conceptual MODEL

The accessibility of SITE components is described using an ACCESSIBILITY ASSESSMENT: this allows any SITE COMPONENT to be described either in terms of suitability for specific USER NEEDs (using a SUITABILITY element) or in terms of LIMITATIONs of the SITE, or both.

To describe accessibility, *NeTEx* models has separate and distinct aspects: (a) the description of the USER'S NEEDs – for example *wheelchair, hearing impaired, vision impaired, lift-averse,* etc.; and (b) the ACCESSIBILITY LIMITATION, i.e. description of the limitations of a SITE ELEMENT to support a specific need, for example *Wheelchair, Step free, Escalator free, Lift free* – the last two also corresponding to some cognitive aversions (e.g. sufferers of claustrophobia may dislike lifts). These aspects can be grouped together as an ACCESSIBILITY ASSESSMENT and associated with various NeTEx ENTITies.

In addition, further information relevant for detailed accessibility is contained on many of the different EQUIPMENT elements – See later below. For example, lift dimensions and controls, step heights, handrails and the number of steps in a staircase, ramp gradients, whether barriers are wheelchair passable, etc.

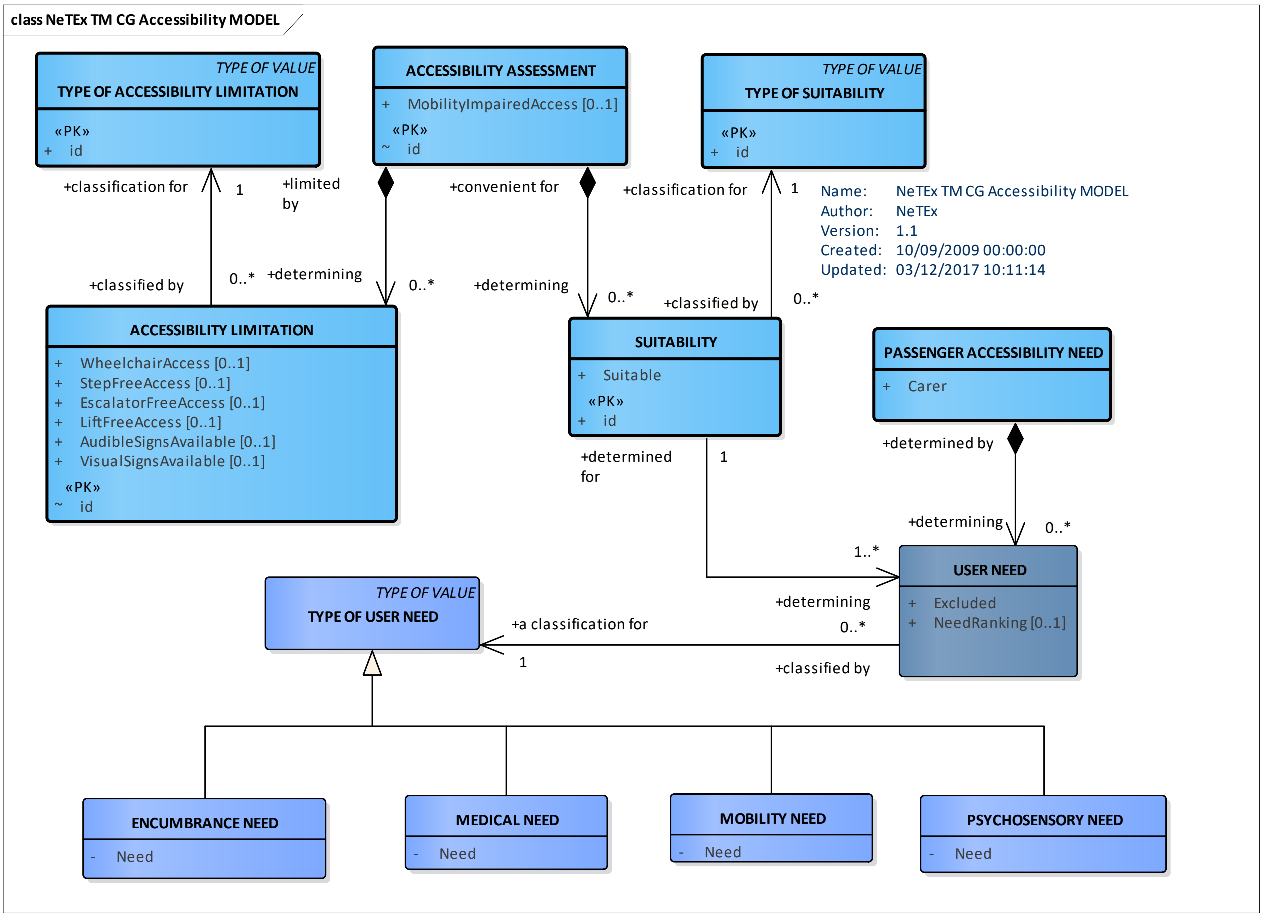


Figure 216 — Accessibility – Conceptual MODEL (UML)

7.6.13.2 Accessibility – Physical Model

The physical model provides specific values for the limitations and user needs.

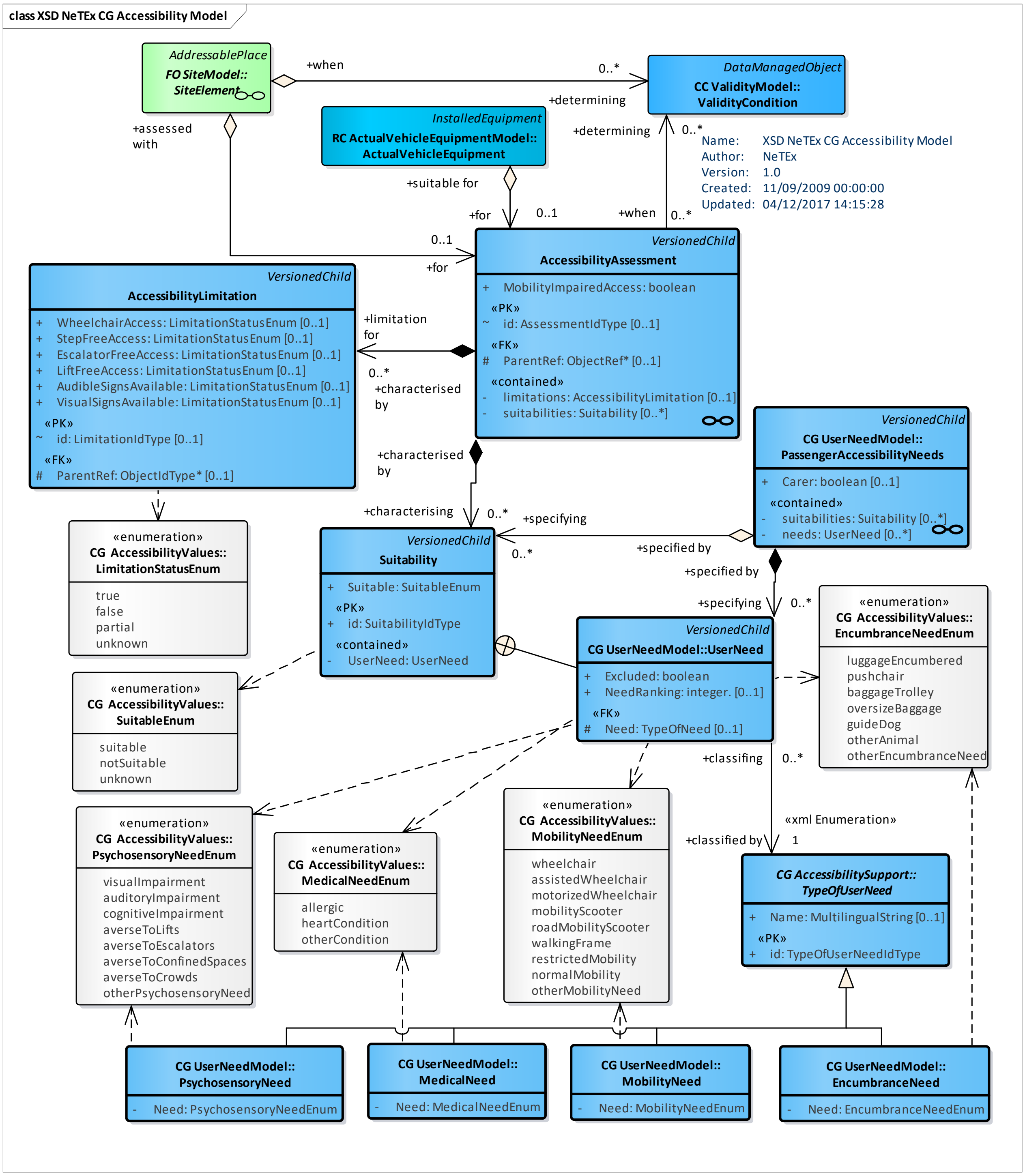


Figure 217 — Accessibility – Physical Model (UML)

7.6.13.2.1 Associating Accessibility Assessments with Site Components

NeTEx allows accessibility criteria to be associated with both SITE COMPONENTS, PATH LINKs and NAVIGATION PATHs.

In order to be able to search for the optimum path for a given set of user accessibility needs it is desirable to specify accessibility data at the most detailed level – on every PATH LINK and QUAY and ACCESS SPACE within a SITE. However in order to provide efficient journey planning it is helpful to summarise at various levels. Thus for example, if all the QUAYs of a STOP PLACE are wheelchair accessible, the STOP PLACE may be marked as wheelchair accessible; if all the PATH LINKs of a NAVIGATION PATH are accessible, then the whole NAVIGATION PATH can be marked as accessible; or if there is at least one wheelchair accessible NAVIGATION PATH between two QUAYs, then the CONNECTION can be marked as accessible.

Summarisation can in principle be derived automatically from the bottom up by looking at the EQUIPMENT and other properties of PATH LINKs and SITE COMPONENTs. For example if a PATH LINK in a NAVIGATION PATH involves the use of a *Lift,* then the PATH should be flagged as ***not*** *LiftFree*.

Nested QUAYs and ACCESS PLACEs must always be on the same level as their parent and can be assumed to have the same accessibility assessment.

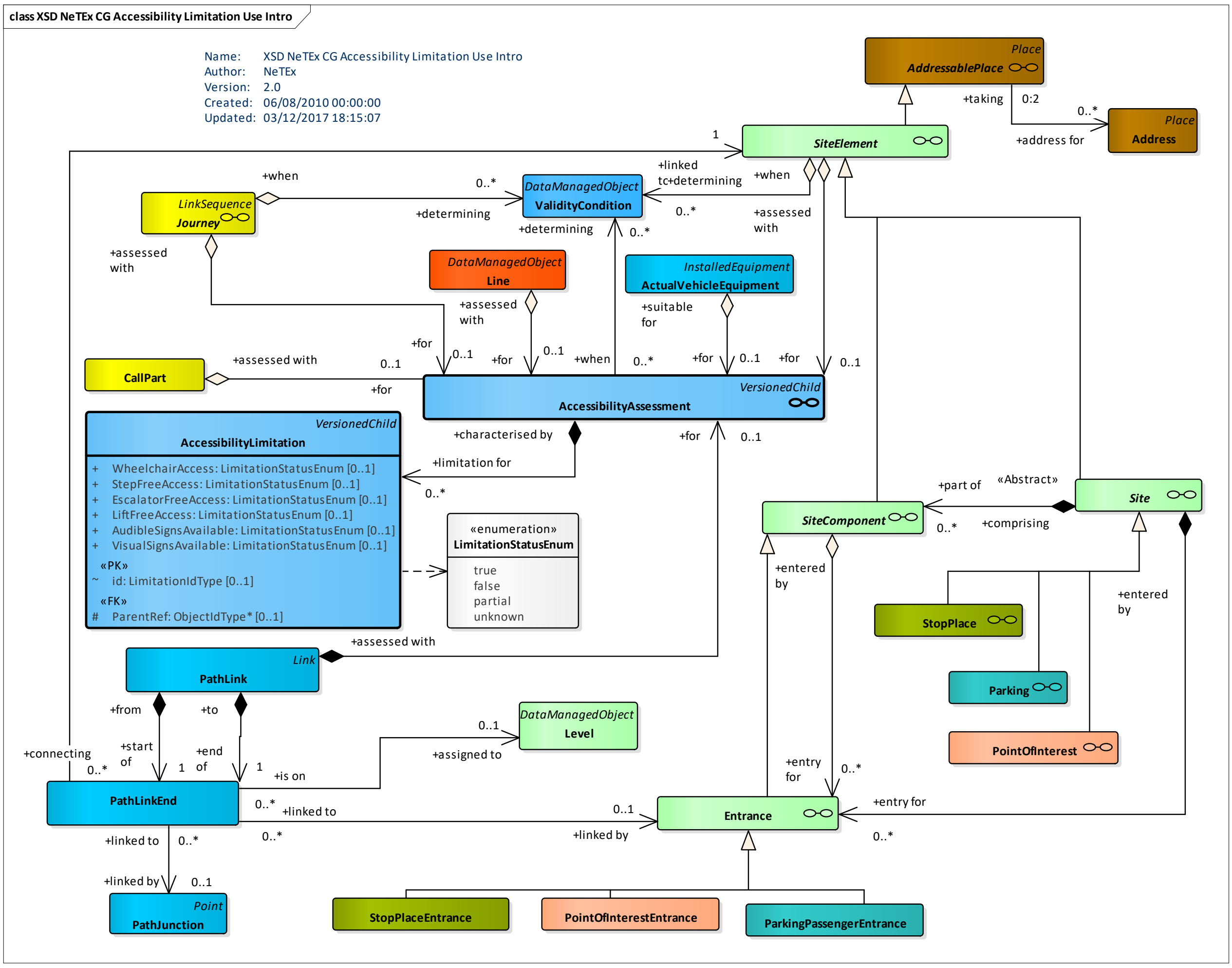


Figure 218 — Accessibility Assessment –Physical Model (UML)

7.6.13.2.2 Use of Accessibility Limitations on Site Components

The limitations should be assessed for individual SITE COMPONENTs as shown in the following figure.

Table 141 – Accessibility Limitations for Site Components

LIMITATION	SITE COMPONENT: e.g. ACCESS SPACE, QUAY	NAVIGATION PATH, PATH LINK, SITE: STOP PLACE, POINT OF INTEREST, etc.
WheelchairAccess	May be reached by someone in a wheelchair (possibly using a Lift).	May be traversed by someone in a wheelchair.
LiftFreeAccess	Does not require the use of a Lifts to reach it.	Does not require the use of a Lift to traverse it.
StepFreeAccess	Does not require the use of Steps to reach it.	Does not require the use of Steps to traverse it.

EscalatorFreeAccess	Does not require the use of Escalators to reach it.	Does not require the use of Escalators to traverse it.
TravelatorFreeAccess	Does not require the use of Travelators to reach it.	Does not require the use of Travelators to traverse it.

7.6.13.2.3 Accessibility Limitation constraints

Certain of the Accessibility LIMITATIONs are mutually exclusive – See the following figure.

Table 142 – Accessibility Attribute Constraints

	LiftFree	StepFree	EscalatorFree	TravelatorFree	Criterion
Wheelchair	***Wheelchair*** access may involve the use of lifts	***Wheelchair*** access must be step free	***Wheelchair*** access must be escalator free	***Wheelchair*** access must be travelator free	*To be able to drive a wheelchair unassisted*
LiftFree	--	***LiftFree*** access may involve the use of steps	***LiftFree*** access may involve the use of escalators	***LiftFree*** access may involve the use of travelators	*To avoid being enclosed in a lift*
StepFree	***StepFree*** access may involve the use of lifts	--	***StepFree*** access must be escalator free too	***StepFree*** access may still involve the use of travelators	*To avoid routes that demand high mobility*
EscalatorFree	***EscalatorFree*** access may involve the use of lifts	***EscalatorFree*** access may involve the use of steps	--	**EscalatorFree** *access may still* involve the use of *travelators*	*To avoid routes that demand high mobility*
TravelatorFree	***TravelatorFree*** access may involve the use of lifts	***TravelatorFree*** access may involve the use of steps	***TravelatorFree*** access must be escalator free	--	*To avoid routes that demand high mobility*

7.6.13.2.4 Using Accessibility Data in Journey Planning

The USER NEEDs can be used to describe inputs to a journey planner to identify the special requirements the user has that may need to be taken into account when seeking journeys. For example, an assisted wheelchair could use a route that involved a limited number of steps and the use of doors. An unassisted wheelchair would not.

Matching a given set of USER NEEDs to the limitations of a SITE is potentially complex and may involve relative weightings of different aspects. It is up to the journey planner to decide how to make these tradeoffs. It may be that there is no perfect match so a least bad or "satisficing" match may be needed. The following table illustrates general relationships between user needs and limitations.

Table 143 – User needs versus Limitations

Limitation	**Wheelchair**	**Lift Free**	**Step Free**	**Escalator Free**	**Travelator Free**	**Visual Signs**	**Auditory Signs**
User Need							
Wheelchair	Strongly prefer	(Neutral)	Strongly prefer	Essential	Prefer?	(Neutral)	(Neutral)
Assisted Wheelchair	Strongly prefer	(Neutral)	Prefer	Essential	(Neutral)	(Neutral)	(Neutral)
Motorized Wheelchair	Strongly prefer	(Neutral)	Strongly prefer	Essential	Prefer?	(Neutral)	(Neutral)
Averse to Escalators	(Neutral)	(Neutral)	(Neutral)	Strongly prefer	Prefer?	(Neutral)	(Neutral)

Averse to Lifts	(Neutral)	Strongly prefer	(Neutral)	(Neutral)	(Neutral)	(Neutral)	(Neutral)
Averse to confined spaces	(Neutral)	Strongly prefer	(Neutral)	low	(Neutral)	(Neutral)	(Neutral)
Restricted mobility	Prefer	(Neutral)	Prefer	Prefer	Prefer	(Neutral)	(Neutral)
Walking frame	Prefer	(Neutral)	(Neutral)	(Neutral)	(Neutral)	(Neutral)	(Neutral)
Guide Dog	(Neutral)	(Neutral)	(Neutral)	Prefer	Prefer	(Neutral)	(Neutral)
Pushchair	Prefer	(Neutral)	Prefer	Prefer	(Neutral)	(Neutral)	(Neutral)
Baggage Encumbered	(Neutral)	(Neutral)	Prefer	Prefer	(Neutral)	(Neutral)	(Neutral)
Baggage Trolley	Prefer	(Neutral)	(Neutral)	Essential	(Neutral)	(Neutral)	(Neutral)
Visual Impairment	(Neutral)	Prefer	Prefer?	Prefer	Prefer	(Neutral)	Strongly prefer
Auditory Impairment	(Neutral)	(Neutral)	(Neutral)	(Neutral)	(Neutral)	Strongly prefer	(Neutral)

NeTEx has a quite detailed user need model and it is possible for journey planners to apply sophisticated matching or partial matching algorithms – however for basic accessibility journey planning it suffices to choose a subset of values corresponding to the simple limitations, as used for example by TfL in London.

Certain of the limitations are mutually exclusive – as indicated by the following table:

Table 144 – Accessibility Attributes for Capability Level 1

Implies	**Criterion**	**LiftFree**	**StepFree**	**EscalatorFree**	**TravelatorFree**
Resulting Limitation					
Wheelchair	*To be able to drive a wheelchair unassisted*	***Wheelchair*** access can use lifts	***Wheelchair*** access must be step free	***Wheelchair*** access must be escalator free	***Wheelchair*** access must be travelator free
LiftFree	*To avoid being enclosed in a lift*	--	***LiftFree*** access may have steps	***LiftFree*** access may have escalators	LiftFree access may have travelators
StepFree	*To avoid routes that demand mobility*	***StepFree*** access may have lifts	--	***StepFree*** access must be escalator free too	***StepFree*** access may have travelators
EscalatorFree	*To avoid routes that demand mobility*	***EscalatorFree*** access may have lifts	***EscalatorFree*** access may have steps	--	**EscalatorFree** *access may have travelators*
TravelatorFree	*To avoid routes that demand mobility*	***TravelatorFree*** access may have lifts	***TravelatorFree*** access may have steps.	***TravelatorFree*** access must be escalator free	--

The ***MobilityImpairedAccess*** value provides an overall summary assessment of an element as accessible or not. The following table shows suggested derivation from the lower level values.

Table 145 – Rules for summarising Accessibility

	Value	***MobilityImpairedAccess***
Wheelchair	*false*	*false*

	true	*true*
	unknown	*false*
LiftFree	*false*	No effect
	true	No effect
	unknown	No effect
StepFree	*false*	*false*
	true	No effect
	unknown	*false*
EscalatorFree	*false*	*false*
	true	No effect
	unknown	*false*
TravelatorFree	*false*	No effect
	true	No effect
	unknown	No effect

7.6.13.3 Accessibility & Equipment

As well as ACCESSIBILITY ASSESSMENTs, a number of different other types of element are relevant for accessibility. These include various types of EQUIPMENT (see later below) such as stairs, lifts, etc. which describe detailed properties and CHECK CONSTRAINTs.

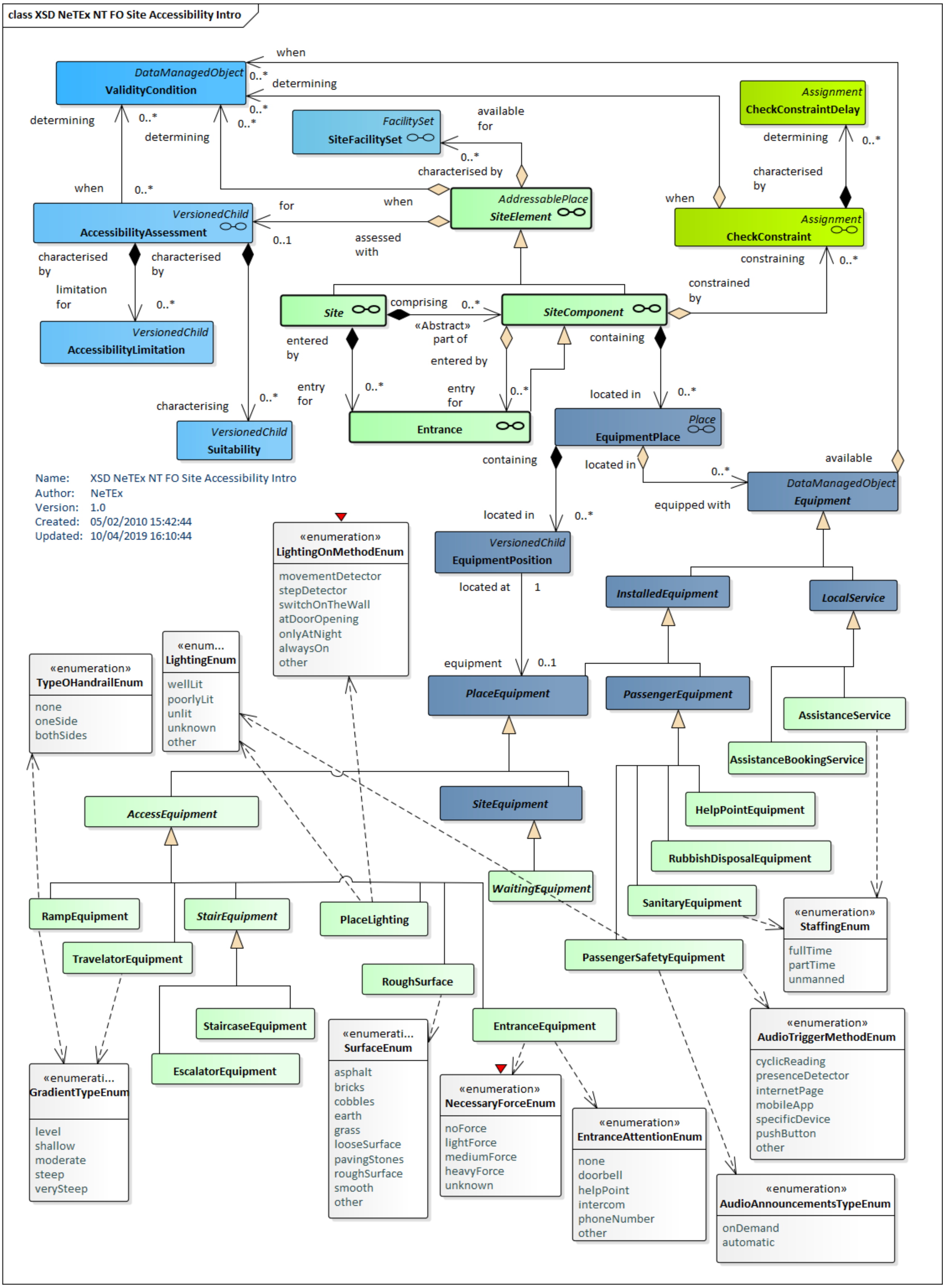


Figure 219 — Accessibility Related Elements – Physical Model (UML)

7.6.13.4 Accessibility – Attributes and XSD

7.6.13.4.1 AccessibilityAssessment – Model Element

Assessment for accessibility purposes of limitations of environment and/or SUITABILITies for one or more USER NEEDs of a component.

Table 146 — *AccessibilityAssessment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	ACCESSIBILITY ASSESSMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*AssessmentIdType*	0:1	Identifier of ACCESSIBILITY ASSESSENT.
«enum»	***MobilityImpaired-Access***	*AccessibilityEnum*	1:1	Overall assessment as to whether there is access for mobility impaired users. See allowed values below.
«cntd»	***limitations***	<u>*AccessibilityLimitation*</u>	0:1	The LIMITATIONs making up the ACCESSIBILITY ASSESSMENT.
«cntd»	***suitabilities***	<u>*Suitability*</u>	0:*	The SUITABILITies making up the ACCESSIBILITY ASSESSMENT.

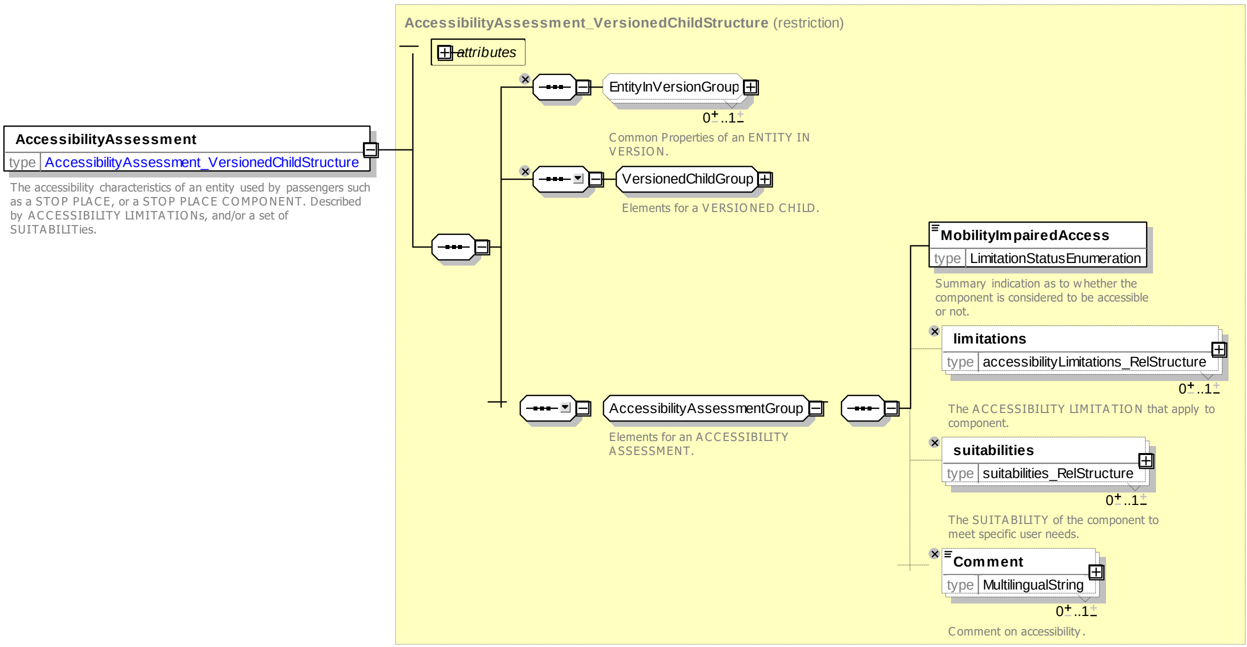


Figure 220 — *AccessibilityAssessment – XSD*

7.6.13.4.1.1 MobilityImpairedAccess – Allowed Values

Allowed values for ***MobilityImpairedAccess*** (*LimitationStatusEnumeration)*.

Table 147 — *LimitationStatus* – Allowed Values

Value	Description
true	Access possible.
false	Access not possible.
partial	Access possible to some areas.
unknown	Not known whether accesss is possible.

7.6.13.4.2 AccessibilityLimitation – Model Element

A categorisation of the ACCESSIBILITY characteristics of a STOP PLACE COMPONENT such as a STOP PATH LINK, STOP PLACE or ACCESS SPACE to indicate its usability by passengers with specific needs, for example, those needing wheelchair access, step-free access or wanting to avoid confined spaces such as lifts.

A small number of well-defined categories are used that are chosen to allow the consistent capture of data and the efficient computation of routes for different classes of user.

Table 148 — *AccessibilityLimitation – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*		ACCESSIBILITY LIMITATION inherits from VERSIONED CHILD.
«PK»	***id***	*LimitationIdType*	0:1	Identifier of ACCESSIBILITY LIMITATION.
«enum»	***WheelchairAccess***	*LimitationStatusEnum*	1:1	Whether there is access for wheelchair users. See allowed values above.
«enum»	***StepFreeAccess***	*LimitationStatusEnum*	0:1	Whether there is step free access. If absent assume '*unknown'*.
«enum»	***EscalatorFree-Access***	*LimitationStatusEnum*	0:1	Whether there is escalator free access. If absent assume '*unknown'*.
«enum»	***LiftFreeAccess***	*LimitationStatusEnum*	0:1	Whether there is lift free access. If absent assume '*unknown*'.
«enum»	***AudibleSigns-Available***	*LimitationStatusEnum*	0:1	Whether there are audible signs. If absent assume '*unknown*'.
«enum»	***VisualSigns-Available***	*LimitationStatusEnum*	0:1	Whether there are visual signs. If absent assume '*unknown*'.

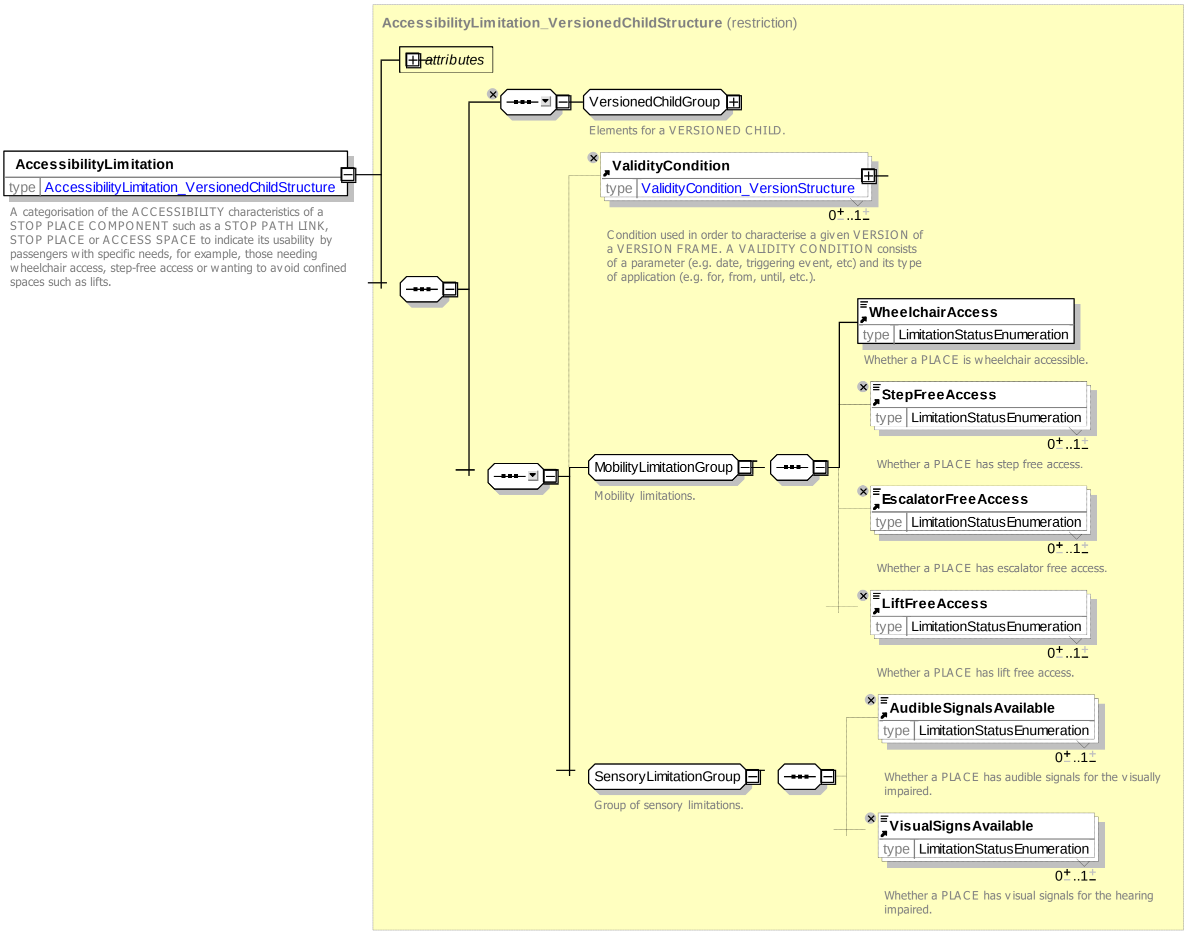


Figure 221 — *AccessibilityLimitation – XSD*

7.6.13.4.3 Suitability – Model Element

A statement of whether a particular USER NEED can be met. It can be used to state whether a SITE can be accessed by a passenger with a particular USER NEED.

Table 149 — *Suitability – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	SUITABILITY inherits from VERSIONED CHILD.
«PK»	***id***	*SuitabilityIdType*	1:1	Identifier of SUITABILITY.
«cntd»	***UserNeed***	<u>*UserNeed*</u>	1:1	A USER NEED for which the SUITABILITY is being specified.
	Excluded	*xsd:boolean*	0:1	Whether USER NEED is included or excluded.
	NeedRanking	*xsd:integer*	0:1	A relative ranking of the need that can be specified in some applications.
«enum»	***Suitable***	*SuitableEnum*	1:1	Whether the USER NEED is met. See allowed values below.

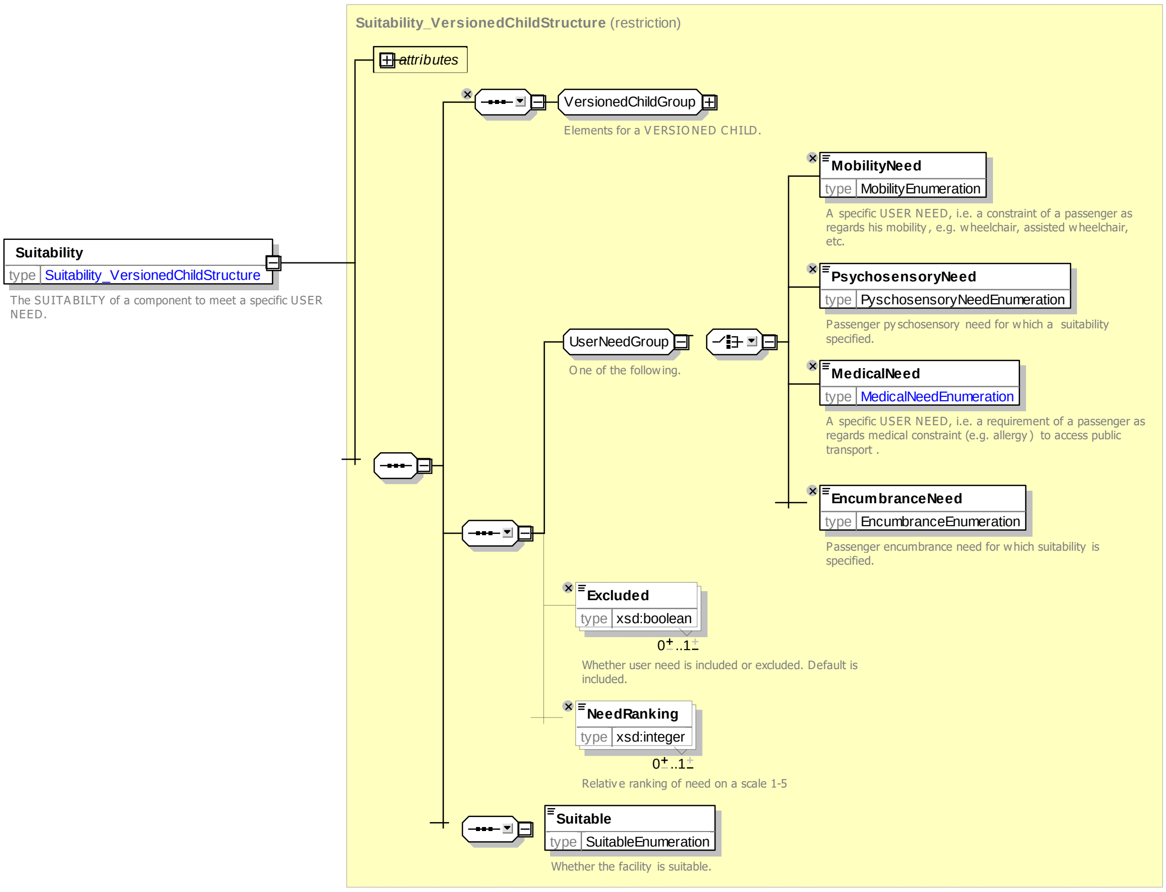


Figure 222 — *Suitability – XSD*

7.6.13.4.3.1 Suitable – Allowed Values

Allowed values for ***Suitable*** (*SuitableEnumeration)*.

Table -150 — SuitableEnum – Allowed Values

Value	Description
suitable	ENTITY is suitable according to criteria
notSuitable	ENTITY is not suitable according to criteria
unknown	Not known whether ENTITY is suitable according to criteria

7.6.13.4.4 UserNeed – Model Element

A User's needs for a particular SUITABILITY.

Table 151 — *UserNeed – Element*

Classification	Name		Type	Cardinality	Description
«PK»			CHOICE	1:1	Type of USER NEED:
«enum»	a	***MobilityNeed***	***MobilityNeed***	1:1	Mobility Need.

«enum»	b	***Psychosensory-Need***	***PsychosensoryNeed***	1:1	Psychosensory Need.
«enum»	c	***MedicalNeed***	***MedicalNeed***	1:1	Mobility Need
«enum»	d	***Encumbrance-Need***	***EncumbranceNeed***	1:1	Encumbrance Need.
	Excluded		*xsd:boolean*	0:1	Whether USER NEED is to be included or excluded.
	NeedRanking		*xsd:integer*	0:1	A relative ranking of the need that can be specified in some applications.

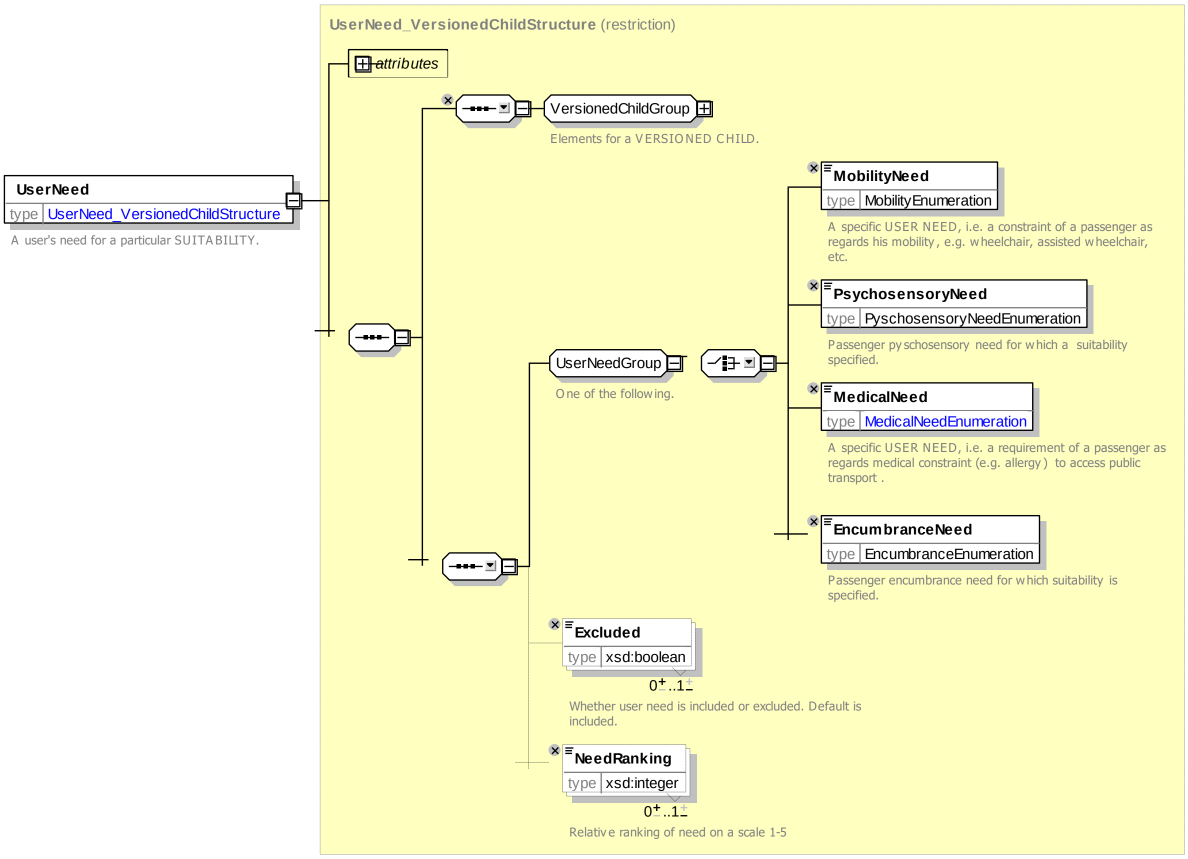


Figure 223 — *UserNeed – XSD*

7.6.13.4.5 MobilityNeed – Model Element

A specific USER NEED, i.e. a constraint of a passenger as regards his mobility, e.g. wheelchair, assisted wheelchair, etc.

Table 152 — *MobilityNeed – Element*

Classification	Name	Type	Cardinality	Description
«enum»	***MobilityNeed***	*MobilityNeedEnum*	1:1	Mobility need of a user. See allowed values

7.6.13.4.5.1 MobilityNeed – Allowed Values

Allowed values for ***MobilityNeed*** (*MobilityNeedEnumeration*).

Table 153 — MobilityNeed – Allowed Values

Value	Description
wheelchair	User has a wheelchair.
assistedWheelchair	User has a wheelchair pushed by a companion.
motorizedWheelchair	User has a motorized wheelchair.
walkingFrame	User has a walking frame.
restrictedMobility	User has restricted mobility.
otherMobilityNeed	User has some other mobility need.

7.6.13.4.6 PsychosensoryNeed – Model Element

Passenger psychosensory need for which a SUITABILITY is specified.

Table 154 — *PsychosensoryNeed – Element*

Classification	Name	Type	Cardinality	Description
«enum»	***PsychosensoryNeed***	*PsychosensoryNeedEnum*	1:1	Type of Psychosensory need. See allowed values

7.6.13.4.6.1 PsychosensoryNeed – Allowed Values

Allowed values for ***PsychosensoryNeed*** (*PsychosensoryNeedEnumeration*).

Table 155 — Psychosensory Need – Allowed Values

Value	Description
visualImpairment	User has visual impairment
auditoryImpairment	User has hearing impairment
cognitiveImpairment	User has cognitive impairment
averseToLifts	User is averse to using lifts
averseToEscalators	User is averse to using escalators
averseToConfinedSpaces	User is averse to confined spaces
averseToCrowds	User is averse to crowds
otherPscyhosensoryNeed	User has other aversion

7.6.13.4.7 MedicalNeed – Model Element

A specific USER NEED, i.e. a requirement of a passenger as regards medical constraint (e.g. allergy) to access public transport.

Table 156 — *MedicalNeed – Element*

Classification	Name	Type	Cardinality	Description
«enum»	***MedicalNeed***	*MedicalNeedEnum*	1:1	Type of Medical need. See allowed values

7.6.13.4.7.1 MedicalNeed – Allowed Values

Allowed values for ***MedicalNeed*** (*MedicalNeedEnumeration*).

Table 157 — *MedicalNeed– AllowedValues*

Value	Description
allergic	User is allergic to environment factors
heartCondition	User has heart condition
otherCondition	user has other condition

7.6.13.4.8 EncumbranceNeed – Model Element

A specific USER NEED, i.e. a requirement of a passenger travelling with luggage, animal or any other object requiring special arrangements to access public transport.

Table 158 — *EncumbranceNeed – Element*

Classification	Name	Type	Cardinality	Description
«enum»	***EncumbranceNeed***	*EncumbranceNeedEnum*	1:1	Type of Encumbrance need.

7.6.13.4.8.1 EncumbranceNeed – Allowed Values

Allowed values for ***EncumbranceNeed*** (*EncumbranceNeedEnumeration*).

Table 159 — *EncumbranceNeed – AllowedValues*

Value	Description
luggageEncumbered	User is encumbered with luggage.
pushchair	User is encumbered with a push chair.
baggageTrolley	User is encumbered with a baggage trolley.
oversizeBaggage	User is encumbered with oversize luggage.
guideDog	User has a guide dog.
otherAnimal	User has another animal.
otherEncumbranceNeed	.

7.6.13.4.9 PassengerAccessibilityNeed – Model Element

A passenger's requirement for accessibility, comprising one or more USER NEEDs. For example, that he is unable to navigate stairs, or lifts, or has visual or auditory impairments. PASSENGER ACCESSIBILITY NEEDS can be used to derive an accessibility constraint for the passenger, allowing the computation of paths for passengers with specifically constrained mobility. Example: Wheelchair, No Lifts, No Stairs.

Table 160 — *PassengerAccessibilityNeeds – Element*

Classification	Name	Type	Cardinality	Description
	Carer	*boolean*	1:1	Whether user has carer.
«cntd»	***suitabilities***	*Suitability*	0:*	List of suitabilities required by user.
«cntd»	***needs***	*UserNeed*	0:*	List of suitabilities required by user.

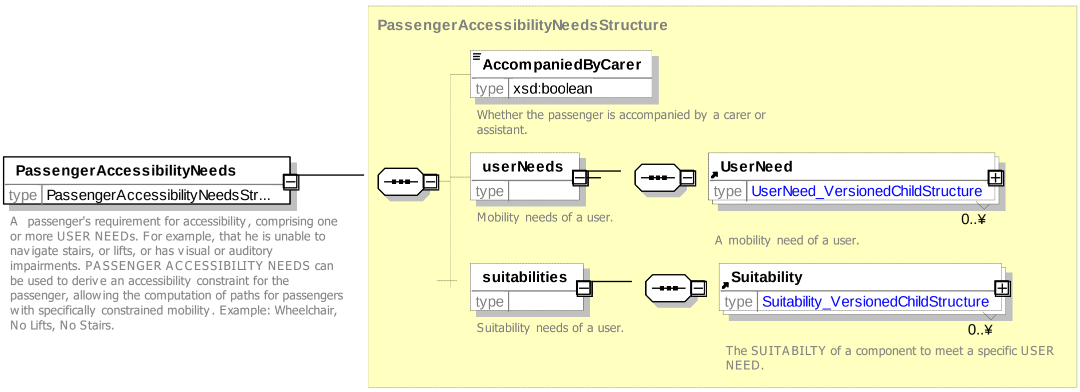


Figure 224 — *PassengerAccessibilityNeeds – XSD*

7.6.13.5 XML Examples of Accessibility

7.6.13.5.1 Assigning an AccessibilityAssesment to an element – XML Example fragment

The following example shows the use of a ACCESSIBILITY ASSESSMENT to define the accessibility properties of a STOP PLACE.

EXAMPLE

```
<StopPlace version="any" responsibilitySetRef="napt:RS_147" id="napt:940GZZLUWHM">
    <Name>West Ham Underground Station </Name>
        <ShortName>West Ham</ShortName>
        <types>
            <TypeOfZoneRef version="any" ref="napt:TypeOfZone:GTMU"/>
        </types>
        <AccessibilityAssessment version="any" id="napt:940GZZLUWHM">
            <MobilityImpairedAccess>true</MobilityImpairedAccess>
            <limitations>
                <AccessibilityLimitation created="2010-05-17T09:30:47Z" modification="new">
                    <WheelchairAccess>true</WheelchairAccess>
                    <StepFreeAccess>true</StepFreeAccess>
                    <EscalatorFreeAccess>true</EscalatorFreeAccess>
                    <LiftFreeAccess>true</LiftFreeAccess>
                    <AudibleSignalsAvailable>false</AudibleSignalsAvailable>
                    <VisualSignsAvailable>true</VisualSignsAvailable>
                </AccessibilityLimitation>
            </limitations>
            <suitabilities>
                <Suitability id="napt:910GWIMBLDN_Ac1_S01">
                    <MobilityNeed> wheelchair</MobilityNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S02">
                    <MobilityNeed> assistedWheelchair</MobilityNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S03">
                    <MobilityNeed> motorizedWheelchair</MobilityNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S04">
                    <EncumbranceNeed>pushchair</EncumbranceNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S05">
                    <EncumbranceNeed>oversizeBaggage</EncumbranceNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S06">
                    <EncumbranceNeed>guideDog</EncumbranceNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S07">
                    <EncumbranceNeed>baggageTrolley</EncumbranceNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S08">
                    <MedicalNeed>heartCondition</MedicalNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S09">
                    <MedicalNeed>allergic</MedicalNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S10">
                    <PsychosensoryNeed>visualImpairment</PsychosensoryNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S11">
                    <PsychosensoryNeed>auditoryImpairment</PsychosensoryNeed>
                    <Suitable>suitable</Suitable>
                </Suitability>
                <Suitability id="napt:910GWIMBLDN_Ac1_S12">
                    <PsychosensoryNeed>averseToCrowds</PsychosensoryNeed>
```

```
                <Suitable>suitable</Suitable>
            </Suitability>
            <Suitability id="napt:910GWIMBLDN_Ac1_S13">
                <PsychosensoryNeed>averseToConfinedSpaces</PsychosensoryNeed>
                <Suitable>suitable</Suitable>
            </Suitability>
            <Suitability id="napt:910GWIMBLDN_Ac1_S14">
                <PsychosensoryNeed>averseToLifts</PsychosensoryNeed>
                <Suitable>suitable</Suitable>
            </Suitability>
            <Suitability id="napt:910GWIMBLDN_Ac1_S15">
                <PsychosensoryNeed>averseToEscalators</PsychosensoryNeed>
                <Suitable>suitable</Suitable>
            </Suitability>
        </suitabilities>

    </AccessibilityAssessment>
    <TransportMode>metro</TransportMode>
    <StopPlaceType>metroStation</StopPlaceType>
```

7.7 Reusable Components

The Reusable Components model defines common Public Transport related objects that can be used in any other NeTEx package. The reusable components are not related to a specific PT topic, but are of general relevance to a number of different topcis.

7.7.1 Reusable Components – Model Dependencies

The reusable components are modularised into separate submodels, built on top of the common framework. The main modules are:

- TRANSPORT MODE Model – Defines standard Transport modes & submodes.
- REUSABLE AVAILABILITY Model – Defines standardised temporal VALIDITY CONDITIONs.
- TOPOGRAPHIC Model – Defines named TOPOGRAPHIC PLACES that relate to places that transport visits.
- TRANSPORT ORGANISATION Model – Defines OPERATORS, AUTHORITIES and other Transport ORGANISATIONs.
- OTHER ORGANISATION Model – Defines SERVICED ORGANISATIONs and other non-Transport ORGANISATIONs.
- GENERIC EQUIPMENT Model – Defines general EQUIPMENT for VEHICLEs and SITEs.
- ACTUAL VEHICLE EQUIPMENT Model – Defines EQUIPMENT USAGE on a VEHICLE.
- VEHICLE TYPE Model – Defines VEHICLE TYPES, VEHICLE MODELs and VEHICLEs.
- VEHICLE PASSENGER EQUIPMENT Model – Defines VEHICLE TYPES, VEHICLE MODELs and VEHICLEs.
- FACILITY Model – Defines simple service and facility categories.
- SERVICE REQUIREMENTS Model – Defines requirements for VEHICLE TYPEs.
- SCHEMATIC MAP Model – Defines general purpose SCHEMATIC MAP contents.
- NOTICE Model – Defines footnotes and other NOTICEs.

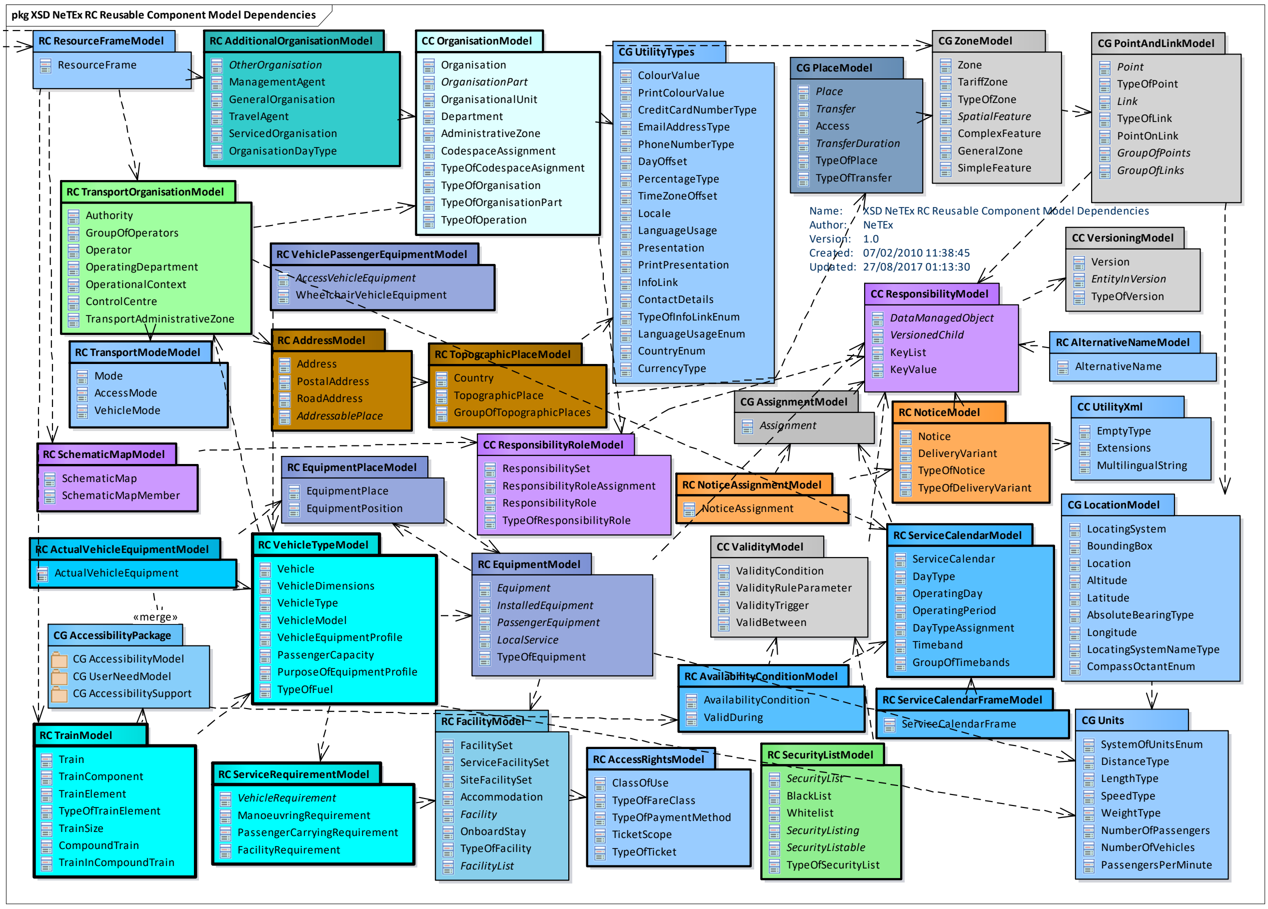


Figure 225 — Reusable components dependencies Components – Model Dependencies

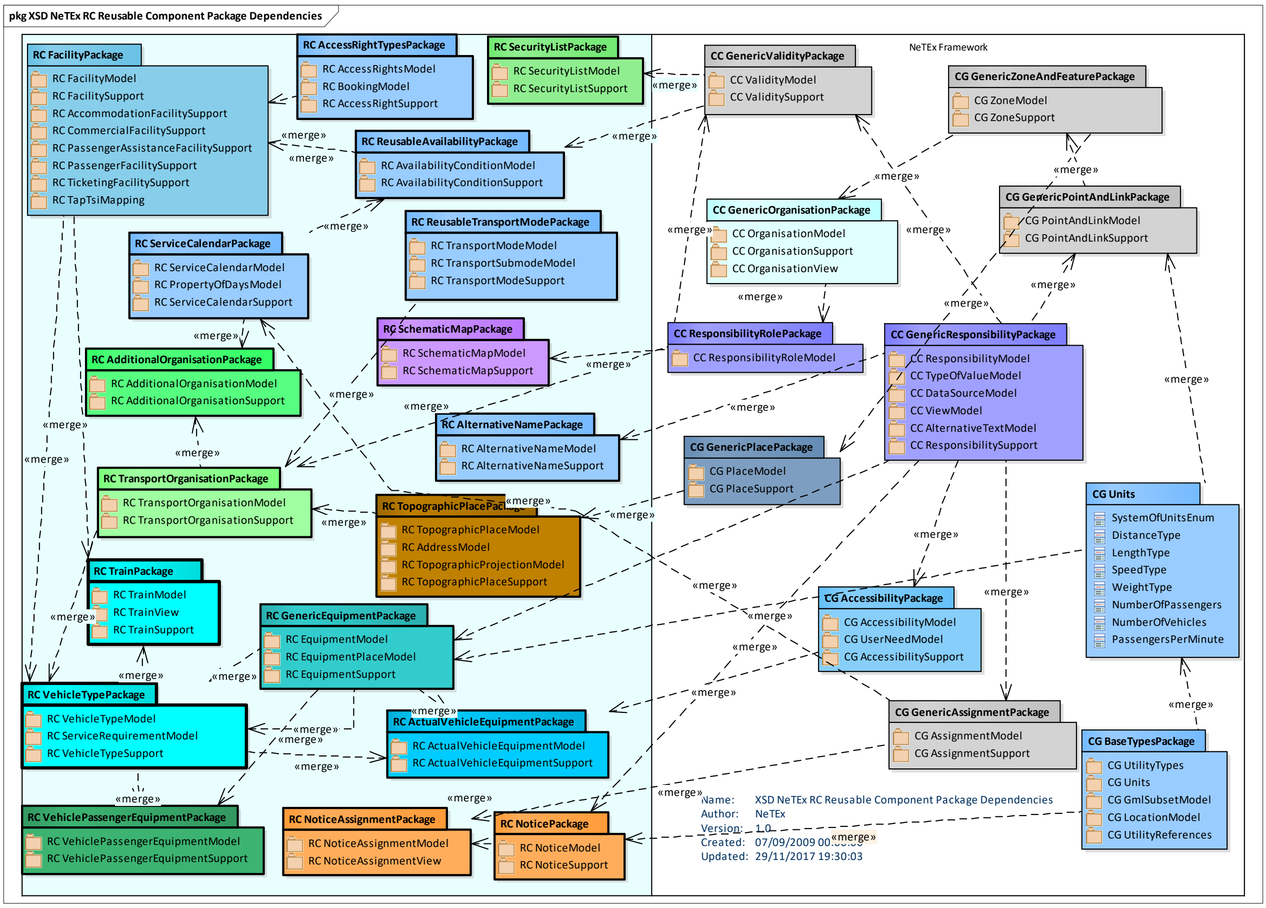


Figure 226 — Reusable components package dependencies

7.7.2 Resource Frame

7.7.2.1 Resource Frame – Physical Model

The RESOURCE FRAME is used to group Reusable Components for exchange, for example to declare the local code values used in a given data set, or entities common to many frames such as organisations. A RESOURCE FRAME can be grouped with other frames using a COMPOSITE FRAME.

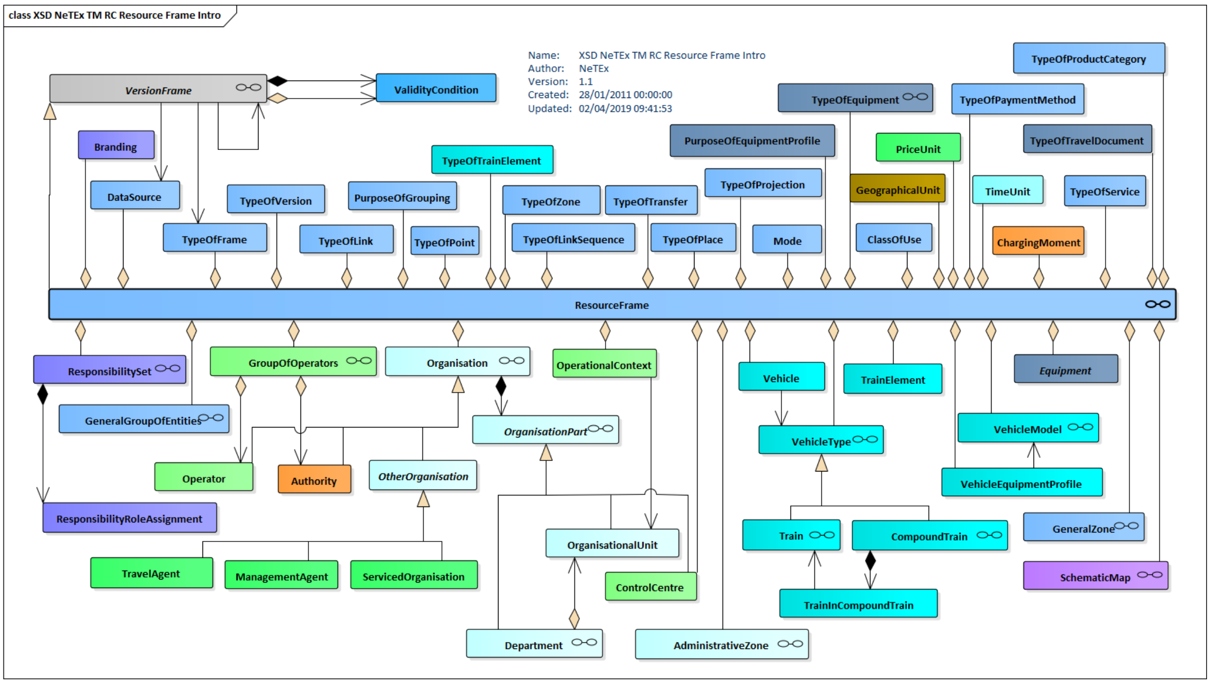


Figure 227 — Resource Frame – Physical MODEL (UML)

7.7.2.2 Resource Frame – Attributes and XSD

7.7.2.2.1 Resource Frame – Model Element

A coherent set of resource data to which the same VALIDITY CONDITIONs have been assigned. Used to define common resources that will be referenced by other types of FRAME.

Table 161 — *ResourceFrame – Element*

Classification	Name	Type	Cardinality	Description
::>	***::>***	*VersionFrame*		RESOURCE FRAME inherits from VERSION FRAME.
«PK»	***id***	*ResourceFrameIdType*	1:1	Identifier of RESOURCE FRAME.
«cntd»	***dataSources***	*DataSource*	0:*	DATA SOURCEs contained in RESOURCE FRAME.
«cntd»	***responsibilitySets***	*ResponsibilitySet*	0:*	RESPONSIBILITY SETs contained in RESOURCE FRAME.
«cntd»	***versions***	*Version*	0:*	VERSIONs contained in RESOURCE FRAME.
«cntd»	***typesOfValue***	*ValueSet*	0:*	Sets of TYPE OF VALUE contained in the RESOURCE FRAME.
«cntd»	***organisations***	*Organisation*	0:*	ORGANISATIONs contained in RESOURCE FRAME.
“cntd»	***groupsOfOperators***	*GroupOfOperators*	0:*	GROUPs OF OPERATORs contained in RESOURCE FRAME.
«cntd»	***operational-Contexts***	*OperationalContext*	0:*	OPERATIONAL CONTEXTs contained in the RESOURCE FRAME.
“cntd»	***controlCentres***	*ControlCentre*	0:*	CONTROL CENTREs contained in RESOURCE FRAME.
«cntd»	***equipment***	*Equipment*	0:*	EQUIPMENT contained in the FRAME.
«cntd»	***vehicleTypes***	*VehicleType*	0:*	VEHICLE TYPEs contained in RESOURCE FRAME.
«cntd»	***vehicleModels***	*VehicleModel*	0:*	VEHICLE MODELs contained in RESOURCE FRAME.

«cntd»	***vehicleEquipment-Profiles***	*VehicleEquipmentProfile*	0:*	VEHICLE EQUIPMENT PROFILEs contained in RESOURCE FRAME.
«cntd»	***vehicles***	*Vehicle*	0:*	VEHICLEs contained in RESOURCE FRAME.
«cntd»	***schematicMaps***	*SchematicMap*	0:*	SCHEMATIC MAPs contained in the FRAME.
«cntd»	***groupsOfEntities***	*GeneralGroupOfEntities*	0:*	GENERAL GROUPs of ENTITies contained in the RESOURCE FRAME.
«cntd»	***zones***	*Zone*	0:*	ZONEs contained in the RESOURCE FRAME +v1.1.
«cntd»	***blacklists***	*Blacklist*	0:*	BLACKLISTs contained in the RESOURCE FRAME. +v1.1
«cntd»	***whitelists***	*Whitelist*	0:*	WHITELISTs contained in the RESOURCE FRAME. +v1.1

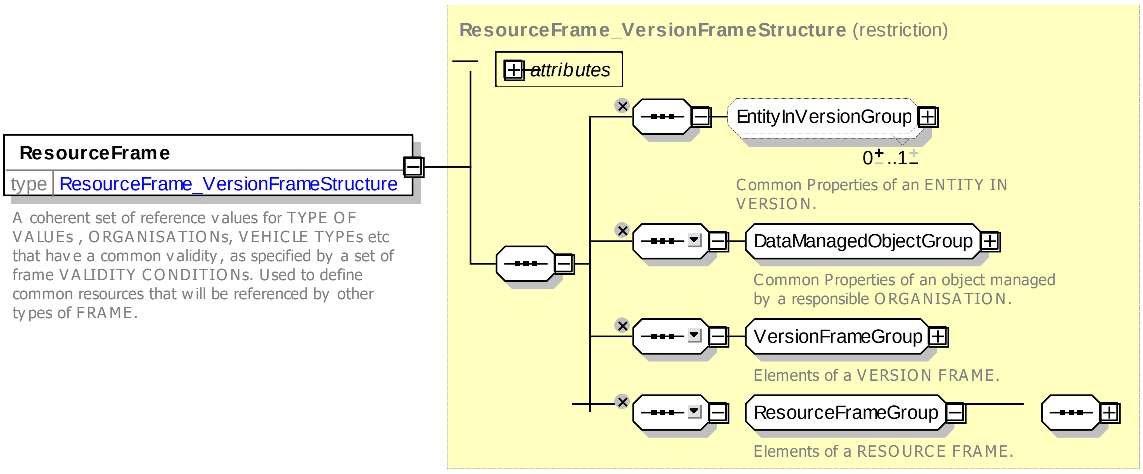


Figure 228 — *ResourceFrame – XSD*

7.7.2.2.1.1 ResourceFrameGroup – XML Group

Figure 229 — *ResourceFrameGroup – XSD*

7.7.3 Transport Mode

7.7.3.1 TRANSPORT MODE – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

The MODE defines the mean of transport used or available. NeTEx subdivides the MODE into TRANSPORT MODE, used inside public transport, and ACCESS MODE, used to join public transport (from the start point of a journey, to the end point, during connections, etc.).

The entity TRANSPORT MODE refers to the classification of transport systems present in large cities or on important transport corridors, for instance: bus, tramway, light rail, metro, long-distance rail, ferry.

A VEHICLE TYPE must belong to one TRANSPORT MODE. For instance, the “bus” TRANSPORT MODE will gather standard, articulated, minibus, double-deck buses.

A LINE is in general operated by only one TRANSPORT MODE: metro line, ferry line, etc.

The ACCESS MODE is any out of vehicle mode used to reach a TRANSPORT MODE.

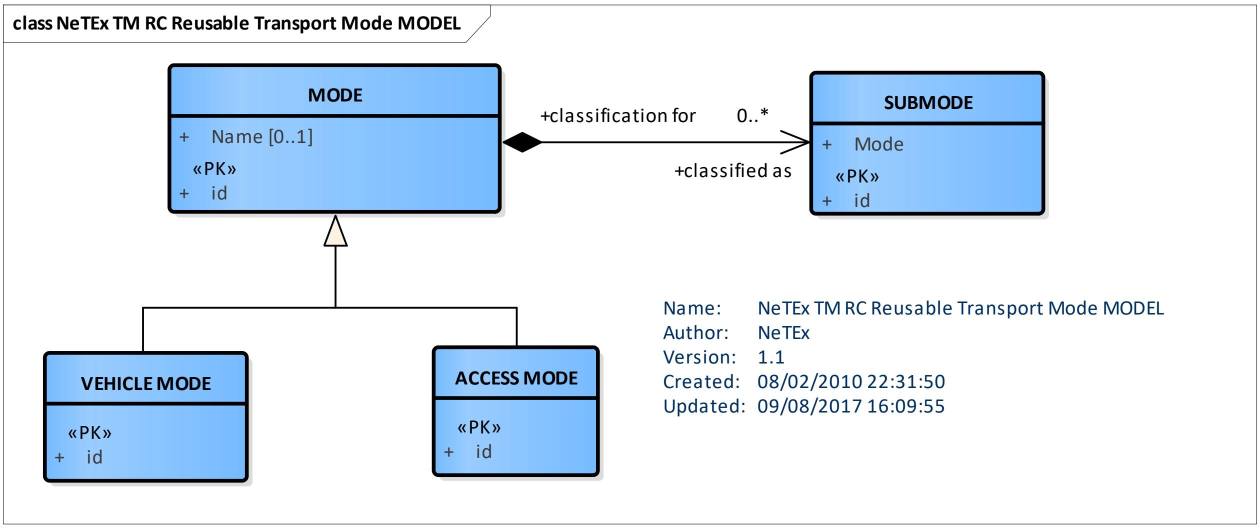


Figure 230 — Reusable Transport Mode – Conceptual MODEL (UML)

7.7.3.2 Transport Mode – Physical Model

The Physical Model for reusable transport mode normally implements the TRANSPORT MODE as an enumerated value so that the value used are standardised.

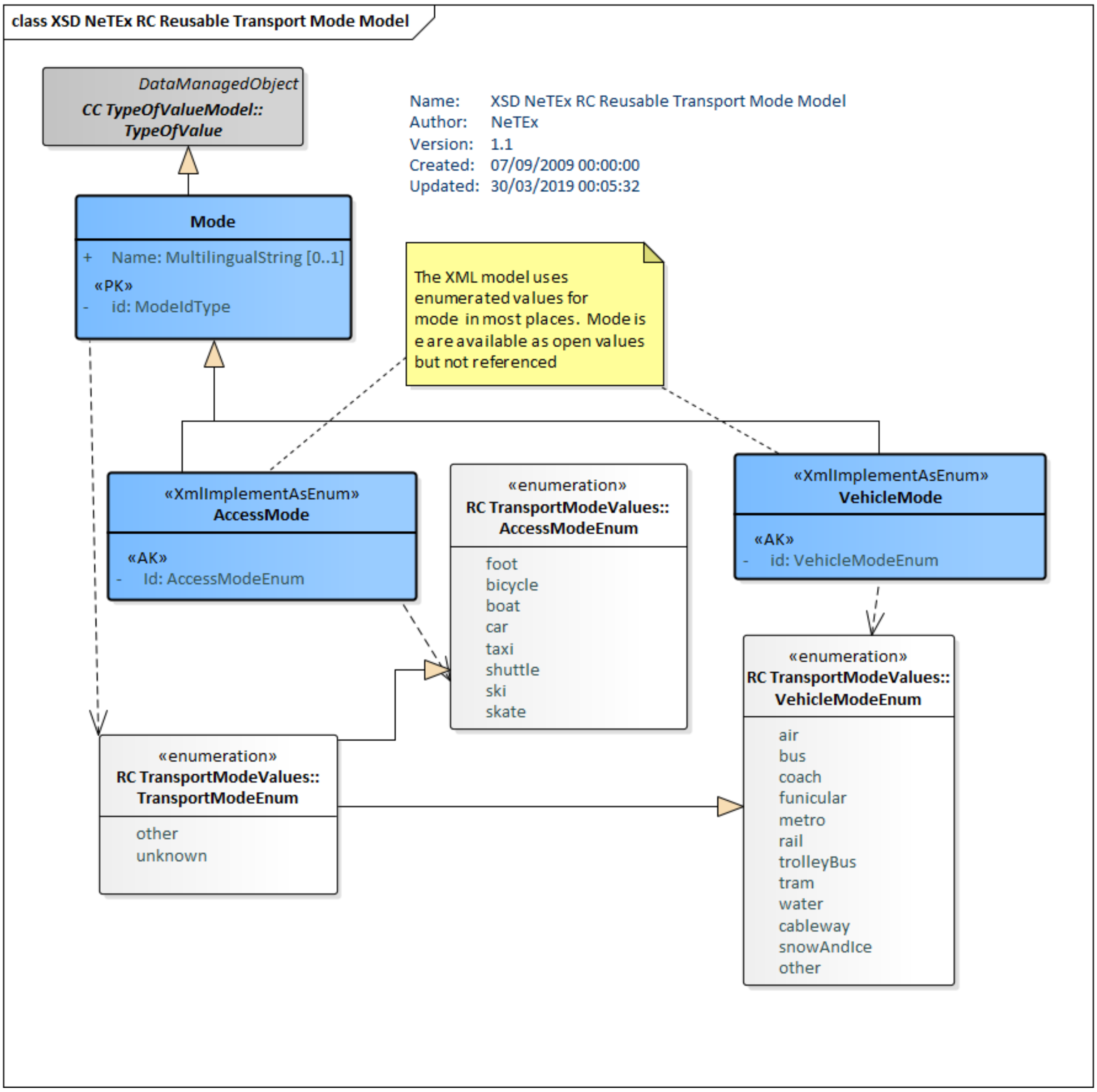


Figure 231 — Reusable Transport Mode – Physical Model (UML)

7.7.3.3 Transport Mode – Attributes and XSD

7.7.3.3.1 OpenTransportMode – Model Element

A characterisation of the operation according to the means of transport. Includes both vehicle (bus, tram, metro, train, ferry, ship) and out of vehicle access modes.

In most places in NeTEx TRANSPORT MODE is implemented as a fixed enumeration, ***TransportMode***. Additionally, an ***OpenTransportMode*** element is also provided to allow reflection by applications, for example, to determine the allowed submodes for a given mode. This is named ***OpenTransportMode*** to avoid confusion with the implementation of *TransportMode* as a fixed enumeration.

Table 162 — *OpenTransportMode – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	<u>*TypeOfEntity*</u>	::>	MODE inherits from TYPE OF ENTITY.
«PK»	***id***	*ModeIdType*	1:1	Identifier of TRANSPORT MODE.
	Name	*MultilingualString*	0:1	Name of MODE.

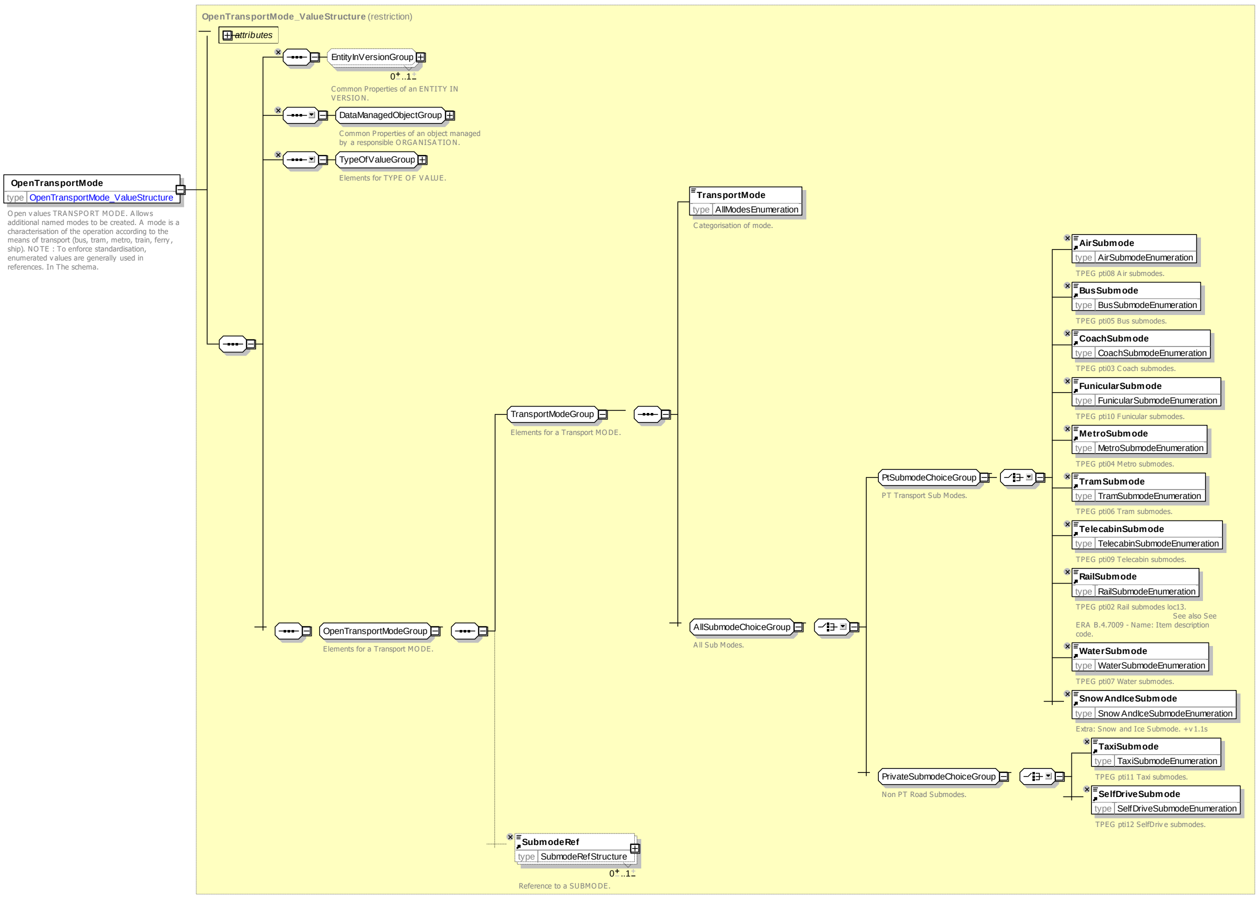


Figure 232 — *OpenTransportMode – XSD*

7.7.3.3.1.1 TransportMode – Allowed Values

Common allowed values for TRANSPORT MODE (*TransportModeEnumeration)*. These are inherited by the other transport modes: ***AccessMode, VehicleMode,*** etc***.***

Table 163 — *TransportMode – AllowedValues*

Value	Description
other	Other mode
unknown	Unknown Mode

7.7.3.3.2 AccessMode – Model Element

ACCESS MODE describes modes of transport that are used to access public transport (taxi, foot, cycle, etc).

Table 164 — *AccessMode* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TransportMode*	::>	ACCESS MODE inherits from TRANSPORT MODE
«AK»	***id***	*AccessModeEnum*	1:1	Identifier of ACCESS MODE. See allowed values below.

Implemented as an enumerated list of values.

7.7.3.3.2.1 AccessMode – Allowed Values

Allowed values for ACCESS MODE (*AccessModeEnum*).

Table 165 — *AccessModeEnum* – *AllowedValues*

Value	Description
foot	Pedestrian access
bicycle	Bicycle
boat	Boat
car	Car
taxi	Taxi
shuttle	Shuttle
ski	Ski. + v1.1
skate	Skate +v1.1

7.7.3.3.3 VehicleMode – Model Element

A characterisation of the operation according to the means of transport (bus, tram, metro, train, ferry, ship).

Implemented as an enumerated list of values.

Table 166 — *VehicleMode* –Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TransportMode*	::>	VEHICLE MODE inherits from TRANSPORT MODE.
«AK»	***id***	*VehicleModeEnum*	1:1	Identifier of VEHICLE MODE. See allowed values below.

7.7.3.3.3.1 VehicleMode – Allowed Values

Allowed values for VEHICLE MODE (*VehicleModeEnum*).

Table 167 — *VehicleModeEnum* – *AllowedValues*

Value	Description
air	Air
bus	Bus
coach	Coach
funicular	Funicular
metro	Metro
rail	Rail
trolleyBus	Trolley Bus
tram	Tram
water	Water
cableway	Cableway
snowAndIce	Snow and ice +v1.1

other	Other mode

7.7.3.4 XML Examples of Transport Mode

7.7.3.4.1 Use of Enumerated Modes – XML Example fragment

The following example shows the use of a TRANSPORT MODE enumeration to define the mode of a STOP PLACE and a line.

EXAMPLE

```
<StopPlace version="any" created="2006-09-11T15:42:00" id="mybus:SP001A">
    <Name>St George's Road (SW19)</Name>
    <ShortName>Place A </ShortName>
    <Centroid>
        <Location>
            <gml:pos srsName="UKOS">524811     170666 </gml:pos>
        </Location>
    </Centroid>
    <TransportMode>bus</TransportMode>
    <StopPlaceType>onstreetBus</StopPlaceType>

<Line   version="002"   created="2011-01-17T09:30:47.0Z"   changed="2011-02-08T09:30:47.0Z"
id="lul:CIR">
    <Name>Circle Line</Name>
    <ShortName>Circle</ShortName>
    <TransportMode>metro</TransportMode>

<ScheduledStopPoint created="2006-09-06T15:44:51" version="10" id="napt:4900ABWD">
    <Name>Abbey Wood Rail Station</Name>
    <PublicCode>tiploc:ABWD</PublicCode>
    <VehicleModes>rail</VehicleModes>
```

7.7.3.4.2 Use of Open Transport Modes – XML Example fragment

The following example shows the use of a OPEN TRANSPORT MODE value to document the modes in use in an application.

EXAMPLE

```
<ValueSet version="any" id="napt:ValueSet:StopTypes">
    <Name>Mode Names</Name>
    <values>
        <OpenTransportMode version="any" id="napt:OpenTransportMode:Metro">
            <Name>Metro</Name>
            <TransportMode>metro</TransportMode>
            <MetroSubmode>tube</MetroSubmode>
        </OpenTransportMode>
        <OpenTransportMode version="any" id="napt:OpenTransportMode:Bus">
            <Name>Bus</Name>
            <TransportMode>bus</TransportMode>
            <BusSubmode>localBus</BusSubmode>
        </OpenTransportMode>
        <OpenTransportMode version="any" id="napt:OpenTransportMode:NightBus">
            <Name>Bus</Name>
            <TransportMode>bus</TransportMode>
            <BusSubmode>nightBus</BusSubmode>
            </OpenTransportMode>
        <OpenTransportMode version="any" id="napt:OpenTransportMode:Tram">
            <Name>Tram</Name>
            <TransportMode>tram</TransportMode>
            <TramSubmode>cityTram</TramSubmode>
        </OpenTransportMode>
    </values>
```

7.7.4 Transport Submode

7.7.4.1 TRANSPORT SUBMODE – Conceptual MODEL

The SUBMODE model allows the TRANSPORT MODE to be further qualified by the specification of a SUBMODE (see TRANSPORT MODE figure).

7.7.4.2 Transport Submode – Physical Model

The Physical model allows the TRANSPORT MODE to be further qualified by the specification of a SUBMODE.

The enumerated values are chosen to be compliant with TPEG and SIRI (knowing that SIRI has already been aligned with TPEG regarding transport modes and submodes).

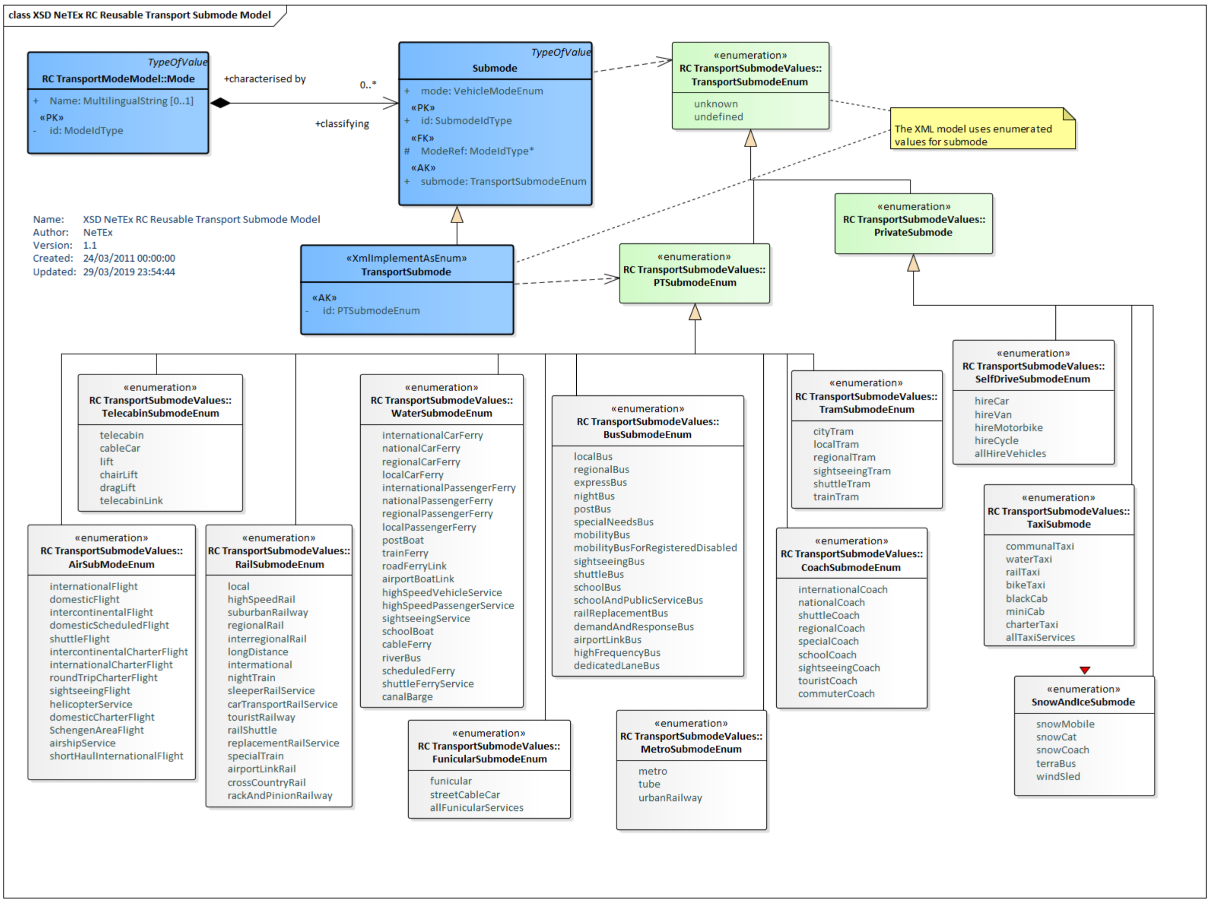


Figure 233 — Reusable Transport Submode – Physical Model (UML)

7.7.4.3 Transport Submode – Attributes and XSD

7.7.4.3.1 Submode – Model Element

A variant of a MODE, as for instance international or domestic rail (rail being the MODE).

Table 168 —*Submode – Element*

Classification	Name	Type	Cardinality	Description
«PK»	***id***	*SubmodeIdType*	1:1	Identifier of SUBMODE.

«FK»	***ModeRef***	*ModeRef*	1:1	Reference to a MODE of which this is a SUBMODE.
«enum»	***mode***	*VehicleModeEnum*	1:1	Vehicle Mode enumeration associated with SUBMODE. See allowed value above.
«enum»	***submode***	*TransportSubmodeEnum*	1:1	Submode enumeration associated with MODE. See allowed values

7.7.4.3.2 TransportSubmode

A characterisation of the operation according to the means of transport (bus, tram, metro, train, ferry, ship).

Table 169 — *TransportSubmode*

Classification	Name	Type	Cardinality	Description
«AK»	***id***	*PTSubmodeEnum*	1:1	Public Transport SUBMODE identifier.

7.7.4.3.2.1 TransportSubmode – Allowed Values

Allowed values for Transport SUBMODE - Common values.

Table 170 — *TransportSubmode – AllowedValues*

Value	Description
unknown	Unknown SUBMODE.
undefined	Undefined SUBMODE.

7.7.4.3.2.2 AirSubmode – Allowed Values

Allowed values for Air SUBMODE of transport (*AirSubModeEnumeration*) - Based on TPEG pti08 Air submodes

Table 171 — *AirSubmodeEnum – AllowedValues*

Value	Description
internationalFlight	International Flight
domesticFlight	Domestic Flight
intercontinentalFlight	Intercontinental Flight
domesticScheduledFlight	Scheduled domestic Flight
shuttleFlight	Shuttle Flight
intercontinentalCharterFlight	Intercontinental Flight
internationalCharterFlight	International Charter Flight
roundTripCharterFlight	Round trip flight
sightseeingFlight	Sightseeing flight
helicopterService	Helicopter service
domesticCharterFlight	Domestic Charter flight
SchengenAreaFlight	Schengen area flight
airshipService	Airship service
shortHaulInternationalFlight	Short haul international flight

7.7.4.3.2.3 **RailSubmode – Allowed Values**

Allowed values for Rail SUBMODE of transport (*RailSubmodeEnumeration*) - Based on TPEG pti02 Rail submodes loc13.

See ERA B.4.7009 - Name: Item description code.

Table 172 — *RailSubmodeEnum – AllowedValues*

Value	Description
local	Local rail
highSpeedRail	See ERA B.4.7009 - Name: Item description code: (8 high speed train). Long distance train formed by a unit capable for high speed running on high speed or normal lines. Most modern train unit.
suburbanRailway	See ERA B.4.7009 - Name: Item description code: (12 subUrban). Regional train organised by the regional government public transport in and around cities, running on its own freeways underground or overground, operational running with signals.
regionalRail	See ERA B.4.7009 - Name: Item description code. (11 Regional). Regional train organised by the regional government even if formed by a unit capable for high speed running on high speed lines.
interregionalRail	See ERA B.4.7009 - Name: Item description code: (10 Interregional). Regional train running in more than one region.
longDistance	See ERA B.4.7009 - Name: Item description code: (9 Intercity). Long distance train formed by a unit capable for high speed or not running on high speed or normal lines. Modern train unit high quality service restricted stopping pattern.
international	International rail
nightTrain	See ERA B.4.7009 - Name: Item description code: (13 Night train). Long distance train running overnight offering sleeping facilities (beds and or couchettes).
sleeperRailService	Sleep rail
carTransportRailService	See ERA B.4.7009 - Name: Item description code: (14 Motor rail). Service transporting passenger's motor vehicle. Passengers are admitted either with vehicle only or with or without vehicle. Service mode.
touristRailway	See ERA B.4.7009 - Name: Item description code: (16 Historic train). Local train providing historical characteristics.
AirportLinkRail	Airport link rail.
railShuttle	Rail shuttle.

replacementRailService	Replacement rail service.
specialTrain	Special train.
crossCountryRail	Cross country rail .
rackAndPinionRailway	See ERA B.4.7009 - Name: Item description code: (15 Mountain train)). Local train adapted for running in mountain railway lines.

7.7.4.3.2.4 **BusSubmode – Allowed Values**

Allowed Values for Bus SUBMODE of transport (*BusSubmodeEnumeration*) - Based on TPEG pti_table_05, loc_table_10.

Table 173 —*BusSubmodeEnum – AllowedValues*

Value	Description
localBus	Local bus
regionalBus	Regional bus
expressBus	Express bus
nightBus	Night bus
postBus	Post bus
specialNeedsBus	Special needs bus
mobilityBus	Mobility bus
mobilityBusFor-RegisteredDisabled	Mobility bus for registered disabled
sightseeingBus	Sightseeing bus
shuttleBus	Shuttle bus
schoolBus	School bus
schoolAndPublic-ServiceBus	School and public service bus
railReplacementBus	Rail replacement bus
demandAnd-ResponseBus	Demand and response bus
airportLinkBus	Airport link bus

7.7.4.3.2.5 **TramSubmode – Allowed Values**

Allowed Values for Tram SUBMODE of transport (*TramSubmodeEnumeration*) - Based on TPEG pti_table_06, loc_table_13.

Table 174 — *TramSubmodeEnum – AllowedValues*

Value	Description
cityTram	City Tram
localTram	Local Tram
regionalTram	Regional Tram
sightseeingTram	Sightseeing Tram
shuttleTram	Shuttle Tram
tramTrain	Tram being able to run from an urban tramway network to main-line railway lines which are shared with conventional trains.

7.7.4.3.2.6 **CoachSubmode – Allowed Values**

Allowed values for Coach SUBMODE of transport (*CoachSubmodeEnumeration*) - Based on TPEG pti03 Coach submodes

Table 175 — *CoachSubmodeEnum – AllowedValues*

Value	Description
internationalCoach	International coach
nationalCoach	National coach
shuttleCoach	Shuttle coach
regionalCoach	Regional coach
specialCoach	Special coach
sightseeingCoach	Sightseeing coach
touristCoach	Tourist coach
commuterCoach	Commuter coach

7.7.4.3.2.7 **MetroSubmode – Allowed Values**

Allowed values for Metro SUBMODE of transport (*MetroSubmodeEnum*) - Based on TPEG pti04 Metro submodes

Table 176 — *MetroSubmodeEnum – AllowedValues*

Value	Description
metro	Metro system
tube	Underground rail system
urbanRailway	Light urban railway system considered as a type of Metro

7.7.4.3.2.8 **WaterSubmode – Allowed Values**

Allowed values for Water SUBMODE of transport *(WaterSubmodeEnumeration)* - Based on TPEG pti_table_07.

Table 177 — *WaterSubmodeEnum – AllowedValues*

Value	Description
internationalCarFerry	International car ferry.
nationalCarFerry	National car ferry.
regionalCarFerry	Regional car ferry.
localCarFerry	Local car ferry.
internationalPassengerFerry	International passenger ferry.
nationalPassengerFerry	National passenger ferry.
regionalPassengerFerry	Regional passenger ferry.
localPassengerFerry	Local passenger ferry.
postBoat	Post boat.
trainFerry	Train ferry.
roadFerryLink	Road ferry link.
airportBoatLink	Airport Boat link.
highSpeedVehicleService	High speed Vehicle ferry.
highSpeedPassengerService	High speed Passenger ferry.
sightseeingService	Sightseeing ferry.
schoolBoat	School board.
cableFerry	Cable ferry.
riverBus	River bus.
canalBarge	Canal barge. + v1.1
scheduledFerry	Scheduled ferry.
shuttleFerryService	Shuttle ferry.

7.7.4.3.2.9 FunicularSubmode – Allowed Values

Allowed values for Funicular SUBMODE of transport (*FunicularSubmodeEnumeration*) - Based on TPEG pti_table_10

Table 178 — *FunicularSubmodeEnum – AllowedValues*

Value	Description
funicular	Funicular
allFunicularServices	All Funicular services

7.7.4.3.2.10 SnowAndIceSubmode – Allowed Values

Allowed values for Snow and Ice SUBMODE of transport (*SnowAndIceSubmodeEnumeration)* (Not in TPEG).*deEnum – AllowedValues*

Value	Description
snowMobile	Snow-mobile service
snowCat	Snow cat serviceat
snowCoach	Coach that runs on snow
terraBus	Terra bus Service
windSled	Wind sled service

7.7.4.3.2.11 TelecabinSubmode – Allowed Values

Allowed values for Telecabin SUBMODE of transport (*TelecabinSubmodeEnumeration)* - Based on TPEG pti_table_09, loc_table_14.

Table 179 — *TelecabinSubmodeEnum – AllowedValues*

Value	Description
telecabin	Telecabine lift.
cableCar	Aerial cable car.
lift	Any lift.
chairLift	Chair lift.
dragLift	Drag lift.
telecabinLink	Link between telecabines.

7.7.4.3.2.12 SelfDriveSubmode – Allowed Values

Allowed values for Self-Drive SUBMODE of transport (*PTSubmodeEnumeration*). - Based on TPEG pti_table_12

Table 180 — *SelfDriveSubmodeEnum – AllowedValues*

Value	Description
hireCar	Hire car.
hireVan	Hire Van.
hireMotorbike	Hire motorbike.
hireCycle	Hire cycle.
allHireVehicles	All hire vehicles

7.7.4.3.2.13 TaxiSubmode – Allowed Values

Allowed values for Taxi SUBMODE of transport. (*TaxiSubmodeEnumeration*) - Based on TPEG pti11 Taxi submodes

Table 181 — *TaxiSubmodeEnum – AllowedValues*

Value	Description
communalTaxi	Communal Taxi.
waterTaxi	Water Taxi.
railTaxi	Rail taxi.
bikeTaxi	Bike taxi (Rickshaw).
blackCab	Registered taxi.

miniCab	Minicab service.		*charterTaxi*	Chartered taxi +v1.1.		*allTaxiServices*	All taxi services.

7.7.5 Service Calendar

7.7.5.1 Introduction

The transport offering of a public transport company is tailored to accommodate different levels of demand. In order to simplify the supply planning almost all operators design their production plan using a classification by type of day, which summarises the level of demand or other characteristics: for example, workday, weekend, school holiday, market day, etc. Long-term planned schedules are designed through the so-called transportation calendar, in which calendar days are classified as specific DAY TYPEs.

OPERATING DAYs are in most cases similar to calendar days, with some possible differences (e.g. start and end times). An assignment process of DAY TYPEs to OPERATING DAY allows selection of the most appropriate schedules to meet the demand and face the traffic conditions. This leads to an operational plan for every OPERATING DAY. The plan is completed by the assignment of physical resources to the theoretical work and amended as necessary to deal with unexpected circumstances.

7.7.5.2 Service Calendar Frame – Physical MODEL

The Calendar elements are grouped in a SERVICE CALENDAR FRAME. This allows the same SERVICE CALENDAR to be shared with many other functional frames (especially TIMETABLE FRAMEs), and for a given functional frame to be used with different SERVICE CALENDARs just by changing the SERVICE CALENDAR FRAME associated with it.

The SERVICE CALENDAR can be seen as placeholder for calendar related objects with the same validity condition and/or applicable period (defined by FromDate-ToDate).

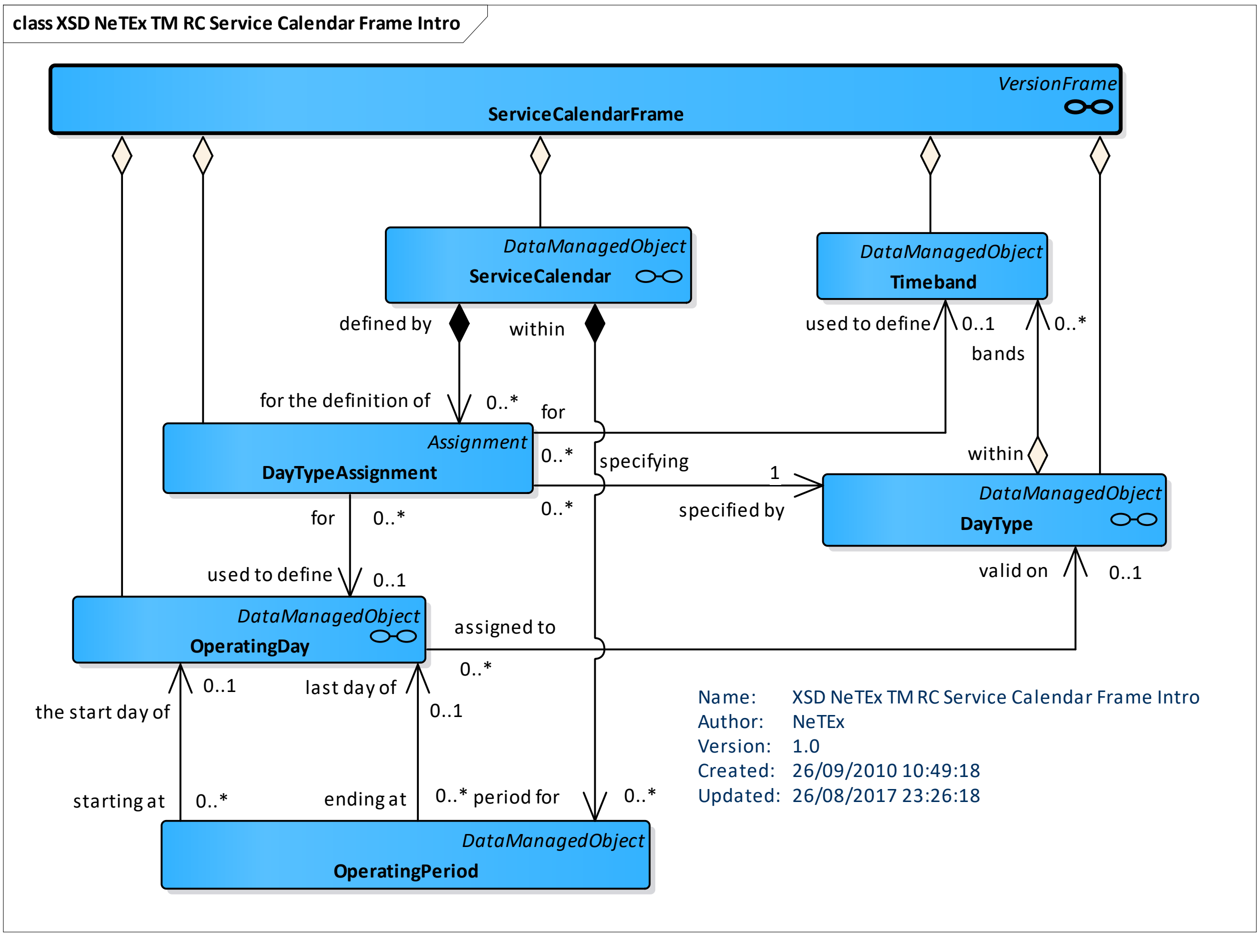


Figure 234 — Service Calendar FRAME – Physical MODEL (UML)

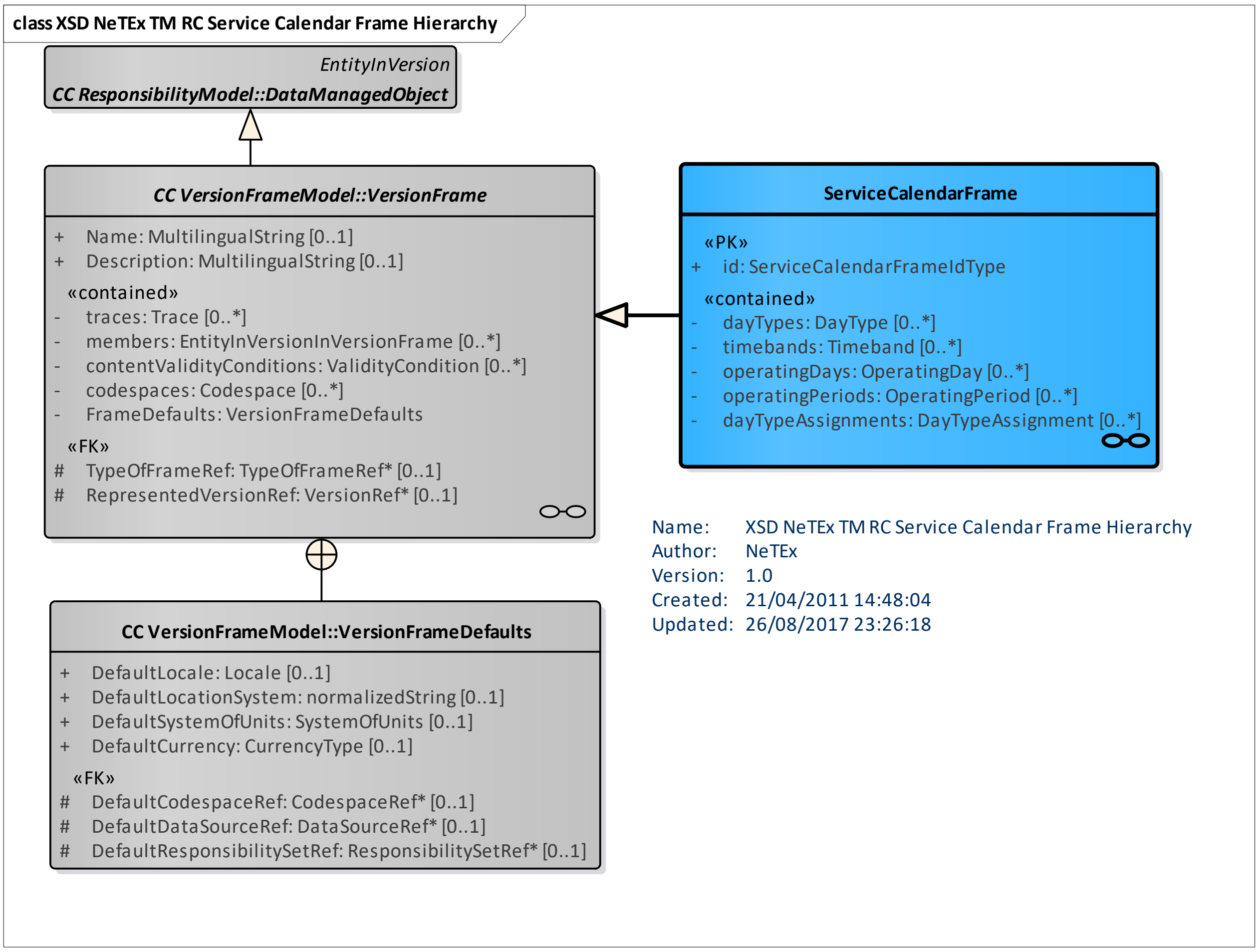


Figure 235 — Service Calendar FRAME – Physical MODEL (UML)

7.7.5.3 SERVICE CALENDAR – Conceptual MODEL

7.7.5.3.1 Day Types – ENTITY

In Transmodel, a DAY TYPE is defined as a combination of various different properties a day may have, and which will influence the transport demand and the running conditions (e.g. traffic flow for buses).

Any single condition that is relevant to the demand will be recorded as a particular PROPERTY OF DAY. For example, “*workday”, “Sunday”, “school holiday”, “market day*” would each be a PROPERTY OF DAY. A workday during school holidays, which is a market day, would be a DAY TYPE, formed with the combination of those three PROPERTies OF DAY.

The most classical PROPERTY OF DAY is the DAY OF WEEK (e.g. “Wednesday”). A DAY TYPE may associate different properties of the same type (e.g. “Tuesday or Thursday”).

The production elements designed during the planning process are characterised by a DAY TYPE and will be used any day of operations to which this DAY TYPE is assigned.

7.7.5.3.2 Operating Days – ENTITY

The day of operation, considered from the point of view of the transportation process control, is described by the entity OPERATING DAY.

The time limits of an OPERATING DAY will often deviate from the official date. One day of operation covers for instance the period from 3.00 a.m. to 1.59 a.m. the day after, the period from 0.00 to 1.59 on the second day being assigned to the operational day which started the day before.

Moreover, an OPERATING DAY may last more than 24 hours. It may be the case in some urban PT operations, for which two OPERATING DAYs overlap during the night. It is more frequent in long-distance railway operations, for which the journeys may last more than one day. However, in such a case, many parameters, such as the schedules, the fares or the passenger information are still based on a DAY TYPE, even if the DAY TYPEs and the OPERATING DAYs last more than 24 hours. The DAY TYPE assignment, in such a case, is usually published as for the date of departure and the passengers invited to refer to this assignment. Therefore, the date characterising an OPERATING DAY corresponds to one of the calendar dates covered by this OPERATING DAY, fixed arbitrarily and in most cases on the first calendar date.

A PERIOD is a continuous interval of several days between two particular OPERATING DAYs, which can be used for several purposes (e.g. VALIDITY CONDITION of a VERSION).

7.7.5.3.3 Day Type Assignment – ENTITY

The production planning requires that a DAY TYPE is assigned to each OPERATING DAY, which is frequently referred as “transportation calendar” or – in the conceptual MODEL – as SERVICE CALENDAR. Ordinarily, this is organised thanks to a default assignment table, which would apply to the whole network. This table determines in advance the DAY TYPE that is valid in the network, for each OPERATING DAY of a given period. This is expressed as a DAY TYPE ASSIGNMENT relationship between DAY TYPE and OPERATING DAY.

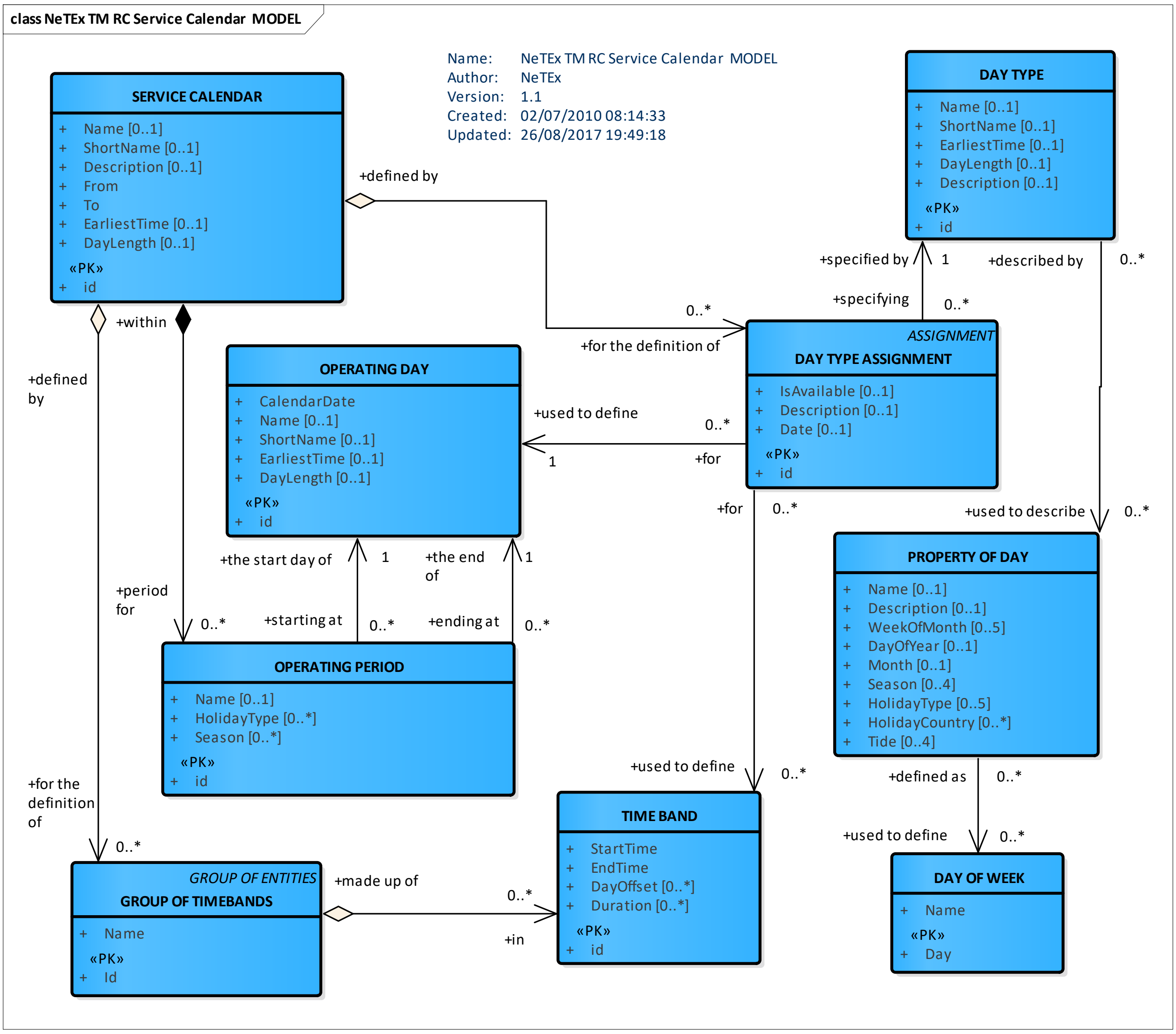


Figure 236 — Service Calendar – Conceptual MODEL (UML)

7.7.5.4 Service Calendar – Physical Model

The following figure shows the Service Calendar Physical Model.

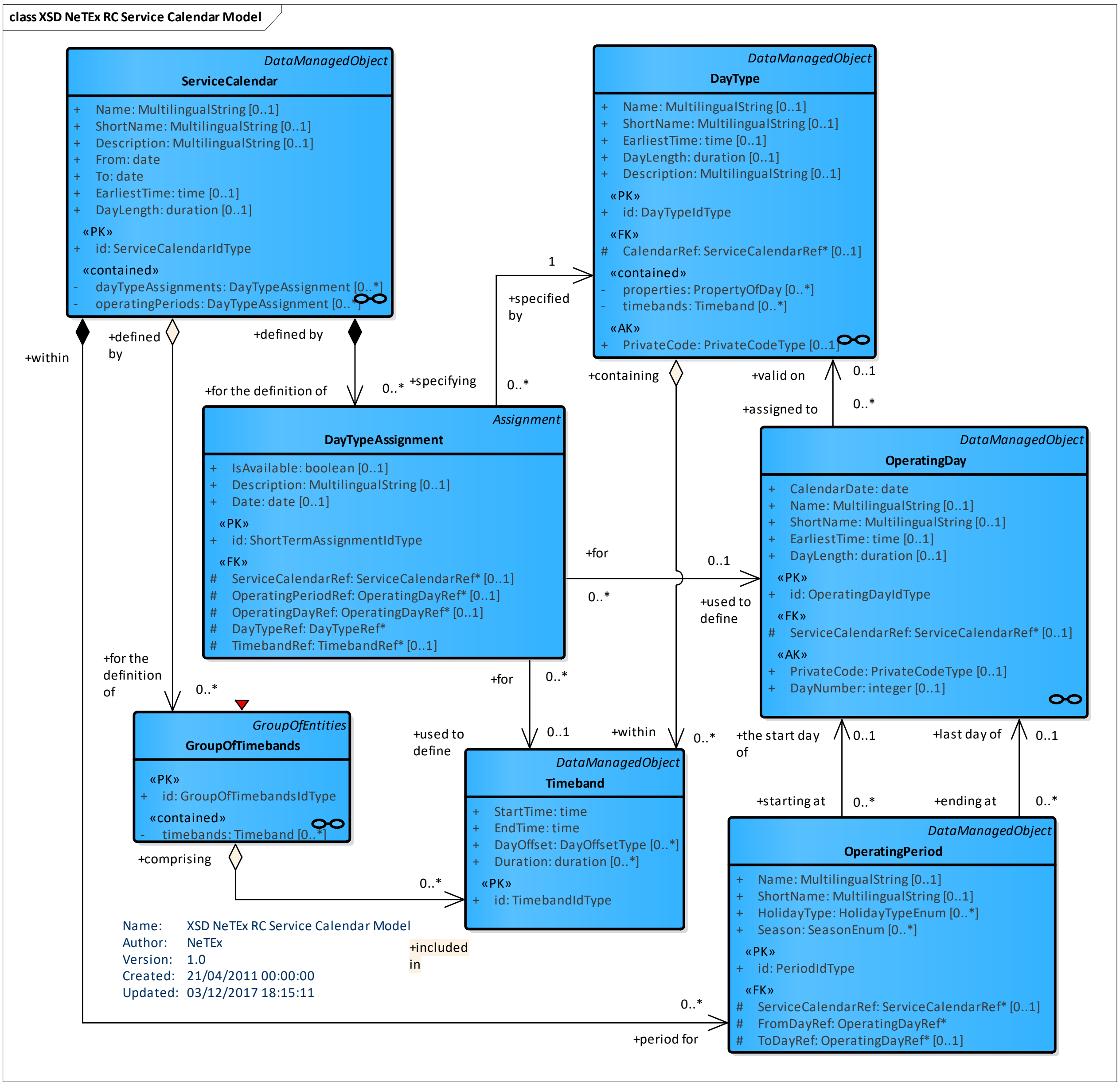


Figure 237 — Service Calendar – Physical Model (UML)

7.7.5.5 Service Calendar – Attributes and XSD

7.7.5.5.1 ServiceCalendar – Model Element

A collection of assignments of OPERATING DAYS to DAY TYPES

Table 182 — *ServiceCalendar – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	*DataManagedObject*	::>	SERVICE CALENDAR inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ServiceCalendarIdType*	1:1	Identifier of SERVICE CALENDAR.
	Name	*MultilingualString*	0:1	Name of SERVICE CALENDAR.
	ShortName	*MultilingualString*	0:1	Short name of SERVICE CALENDAR.
	FromDate	*xsd:date*	0:1	Inclusive start date for validity of SERVICE CALENDAR.
	ToDate	*xsd:date*	0:1	Inclusive end date for validity of SERVICE CALENDAR.
	EarliestTime	*xsd:time*	0:1	Earliest time that days start in SERVICE CALENDAR. Default to use if not specified on individual OPERATING DAY.
	DayLength	*xsd:duration*	0:1	Day length used with earliest time to work out end of day. Default to use if not specified on individual OPERATING DAY.
«cntd»	***dayTypes***	*DayType*	0:*	DAY TYPEs in SERVICE CALENDAR.
«cntd»	***timebands***	*Timeband*	0:*	TIMEBANDs in SERVICE CALENDAR.
«cntd»	***operatingDays***	*OperatingDay*	0:*	OPERATING DAYs in SERVICE CALENDAR.
«cntd»	***operatingPeriods***	*OperatingPeriod*	0:*	OPERATING PERIODs in SERVICE CALENDAR.
«cntd»	***dayType-Assignments***	*DayTypeAssignment*	0:*	DAY TYPE ASSIGNMENTs in SERVICE CALENDAR.

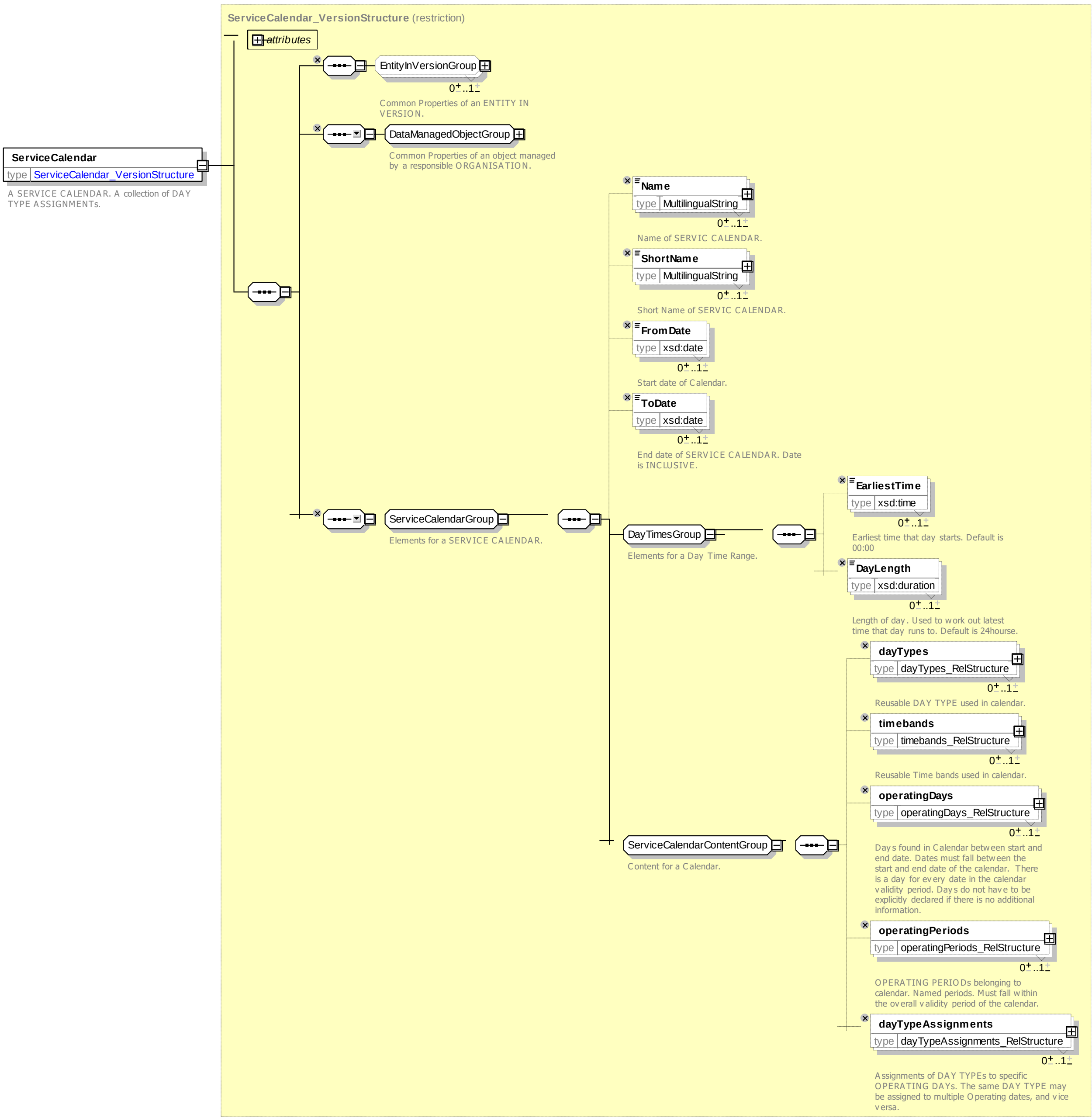


Figure 238 — *ServiceCalendar – XSD*

7.7.5.5.1.1 ServiceCalendarContentGroup – XML Group

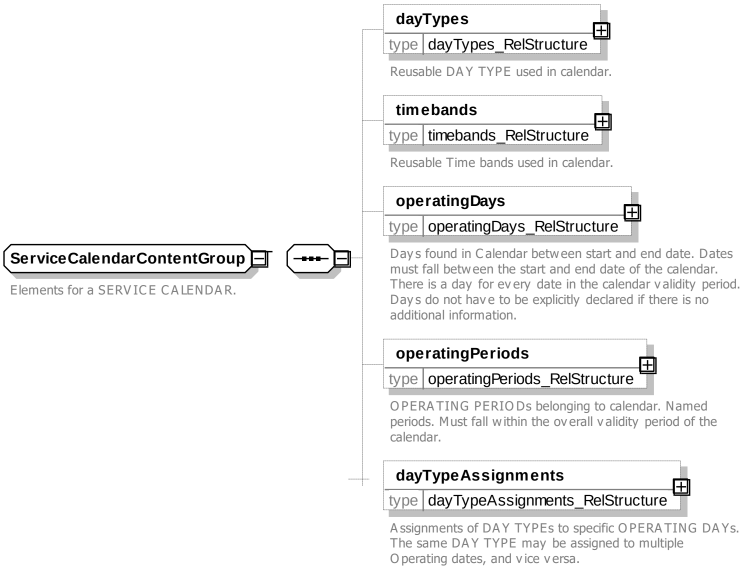


Figure 239 — *ServiceCalendarContentGroup – XSD*

7.7.5.5.2 DayType – Model Element

A type of day characterised by one or more properties which affect public transport operation. For example: weekday in school holidays.

Table 183 — *DayType* –Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	DAY TYPE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*DayTypeIdType*	1:1	Identifier of DAY TYPE.
	Name	*MultilingualString*	0:1	Name of DAY TYPE.
	ShortName	*MultilingualString*	0:1	Short name of DAY TYPE.
	Description	*MultilingualString*	0:1	Description of DAY TYPE.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative Identifier for DAY TYPE.
	EarliestTime	*xsd:time*	0:1	Earliest start time of DAY TYPE.
	DayLength	*xsd:duration*	0:1	Length of DAY TYPE.
«cntd»	***properties***	*PropertyOfDay*	0:*	PROPERTies of DAY TYPE.
«cntd»	***timebands***	*Timeband*	0:*	TIMEBANDs of DAY TYPE.

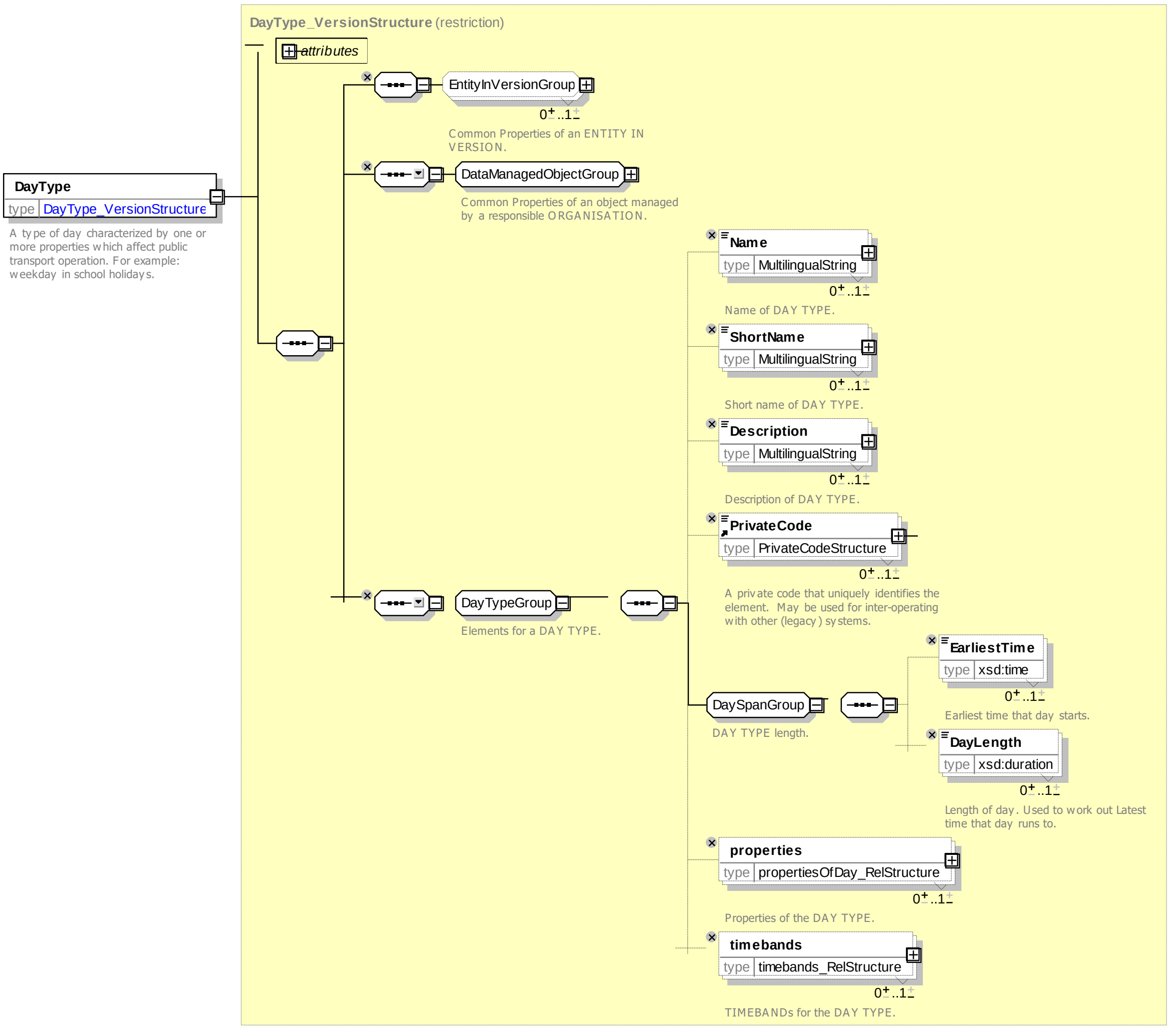


Figure 240 — *DayType – XSD*

7.7.5.5.3 OperatingDay – Model Element

A day of public transport operation in a specific calendar. An OPERATING DAY may last more than 24 hours.

Table 184 — *OperatingDay – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	OPERATING DAY inherits from DATA MANAGED OBJECT.
«PK»	***Id***	*OperatingDayIdType*	1:1	Identifier of OPERATING DAY.
	CalendarDate	*xsd:date*	1:1	Calendar date of OPERATING DAY.
«FK»	***ServiceCalendarRef***	*CalendarRef*	0:1	Reference to the SERVICE CALENDAR to which OPERATING DAY belongs.
	Name	*MultilingualString*	0:1	Name of OPERATING DAY.
	ShortName	*MultilingualString*	0:1	Short Name of DAY TYPE.

«AK»	***DayNumber***	*xsd:integer*	1:1	Day Number if different from Id.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative Identifier for OPERATIG DAY.
	EarliestTime	*xsd:time*	1:1	Start time of OPERATING DAY.
	DayLength	*xsd:duration*	1:1	Length of OPERATING DAY.

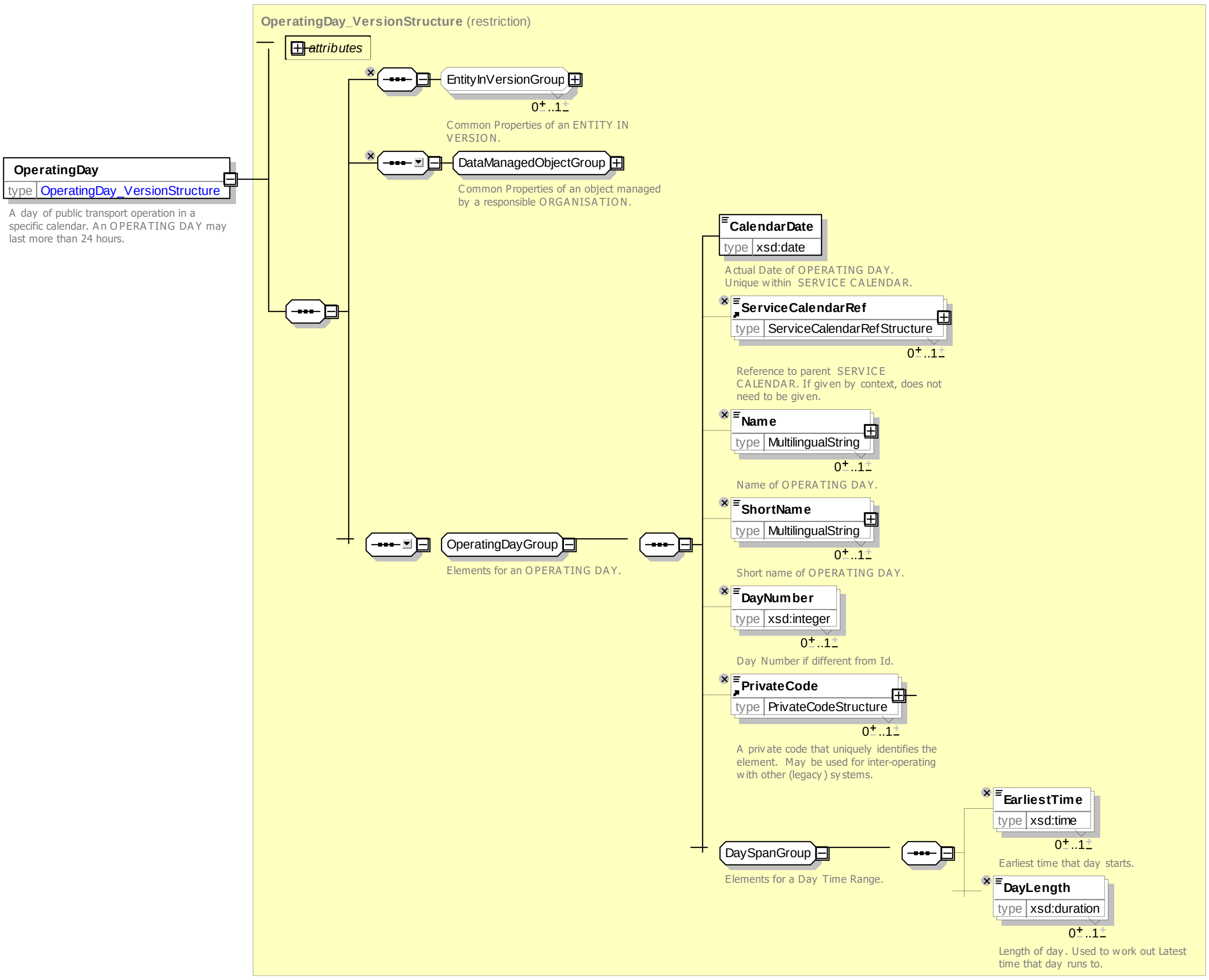


Figure 241 — *OperatingDay – XSD*

7.7.5.5.4 OperatingPeriod – Model Element

A continuous interval of time between two OPERATING DAYs which will be used to define validities.

Table 185 — *OperatingPeriod – Element*

Classific ation	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	OPERATING PERIOD inherits from DATA MANAGED OBJECT. NB a VALIDITY CONDITION is not allowed

«PK»	***id***	*PeriodIdType*	1:1	Identifier of OPERATING PERIOD.
«FK»	***ServiceCalendarRef***	*CalendarRef*	0:1	SERVICE CALENDAR to which PERIOD belongs.
	Name	*MultilingualString*	0:1	Name of OPERATING PERIOD.
	ShortName	*MultilingualString*	0:1	Short Name of OPERATING PERIOD.
«FK»	a ***From-OperatingDayRef***	*OperatingDayRef*	1:1	Start OPERATING DAY of OPERATING PERIOD.
	b ***FromDate***	*xsd:dateTime*	1:1	Start OPERATING DATE (optimisation).
«FK»	a ***ToOperating-DayRef***	*OperatingDayRef*	1:1	End OPERATING DAY of OPERATING PERIOD.
«atr»	b ***ToDate***	*xsd:dateTime*	1:1	End OPERATING DATE (optimisation).
«enum»	***HolidayType***	*HolidayTypeEnum*	0:*	Holiday type of OPERATING PERIOD. See allowed values below.
«enum»	***Season***	*SeasonEnum*	0:*	Season of OPERATING PERIOD. See allowed values below.

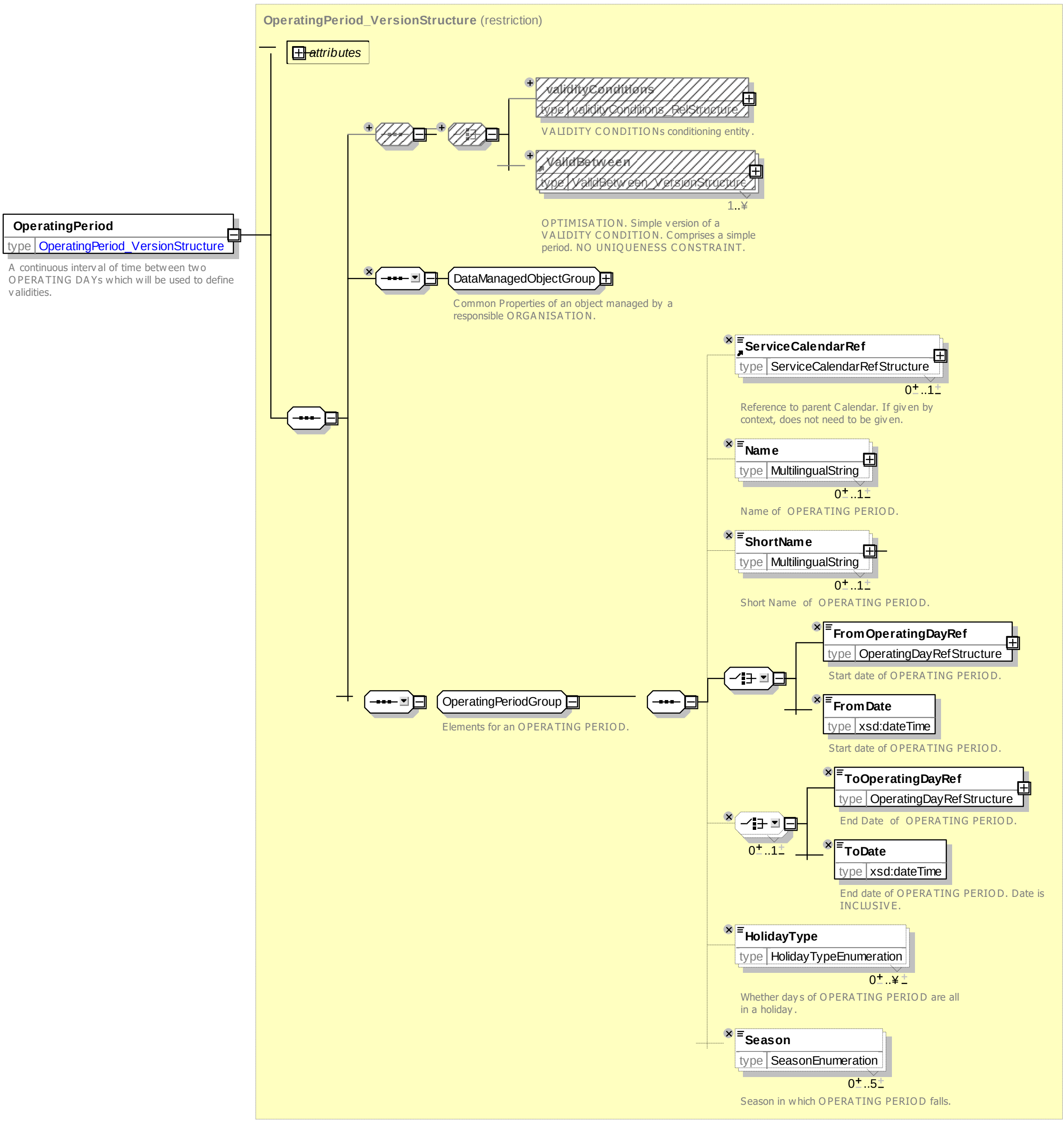


Figure 242 — *OperatingPeriod – XSD*

7.7.5.5.5 DayTypeAssignment – Model Element

A specification of a particular DAY TYPE which will be valid during a TIME BAND on an OPERATING DAY for a GROUP OF LINEs. This assignment overrides the DAY TYPE which was generally chosen for this OPERATING DAY in the overall DAY TYPE assignment plan.

Table 186 — *DayTypeAssignment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Assignment*	::>	DAY TYPE ASSIGNMENT inherits from ASSIGNMENT.

«PK»	***id***	*DayTypeAssignment-IdType*	1:1	Identifier of DAY TYPE ASSIGNMENT.
«FK»	***ServiceCalendar-Ref***	*CalendarRef*	0:1	SERVICE CALENDAR to which DAY TYPE ASSIGNMENT belongs.
«FK»	***OperatingPeriod-Ref***	*OperatingDayRef*	1:1	Reference to OPERATING PERIOD assigned by this DAY TYPE ASSIGNMENT. Mutually exclusive with OPERATING DAY.
«FK»	***OperatingDayRef***	*OperatingDayRef*	1:1	Reference to OPERATING DAY assigned by this DAY TYPE ASSIGNMENT.
	Date	*xsd:date*	1:1	Date - may be used instead of OPERATING DAY.
«FK»	***DayTypeRef***	*DayTypeRef*	1:1	Reference to DAY TYPE assigned by this DAY TYPE ASSIGNMENT.
«FK»	***TimeBandRef***	*TimeBandRef*	0:1	Reference to TIME BAND assigned by this DAY TYPE ASSIGNMENT.
	isAvailable	xsd:boolean	0:1	Whether available on assigned day

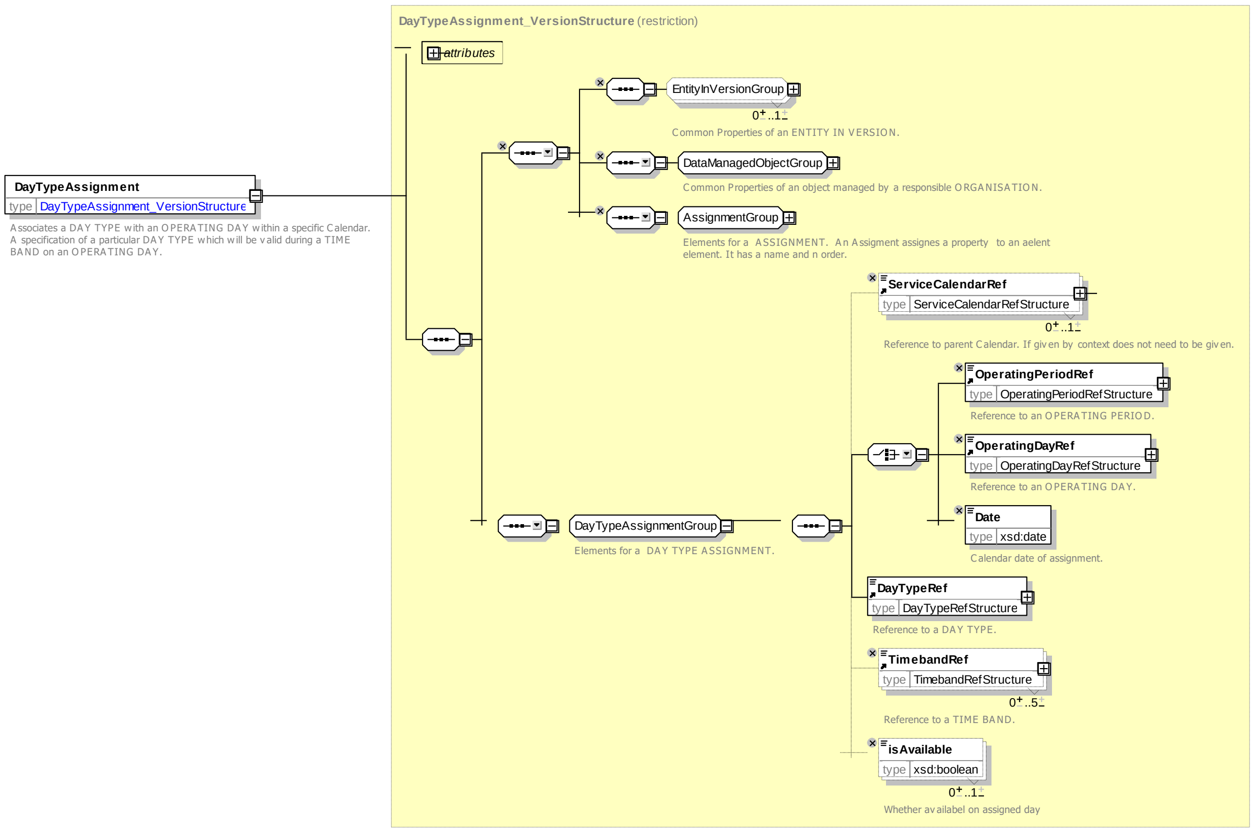


Figure 243 — *DayTypeAssignment – XSD*

7.7.5.5.6 Timeband – Model Element

A period in a day, significant for some aspect of public transport, e.g. similar traffic conditions or fare category.

Table 187 —*Timeband – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	TIME BAND inherits from DATA MANAGED OBJECT.
«FK»	***id***	*TimebandIdType*	1:1	Identifier of TIME BAND.
	Name	*MultilingualString*	0:1	Name of TIME BAND.
	StartTime	*xsd:time*	1:1	Inclusive start time of TIME BAND.
	EndTime	*xsd:time*	1:1	Inclusive end time of TIME BAND.
	DayOffset	*xsd:integer*	0:1	Day offset of end time from start time. If same day, zero.
	Duration	*xsd:duration*	0:1	Length of day - Alternative to use of end time.

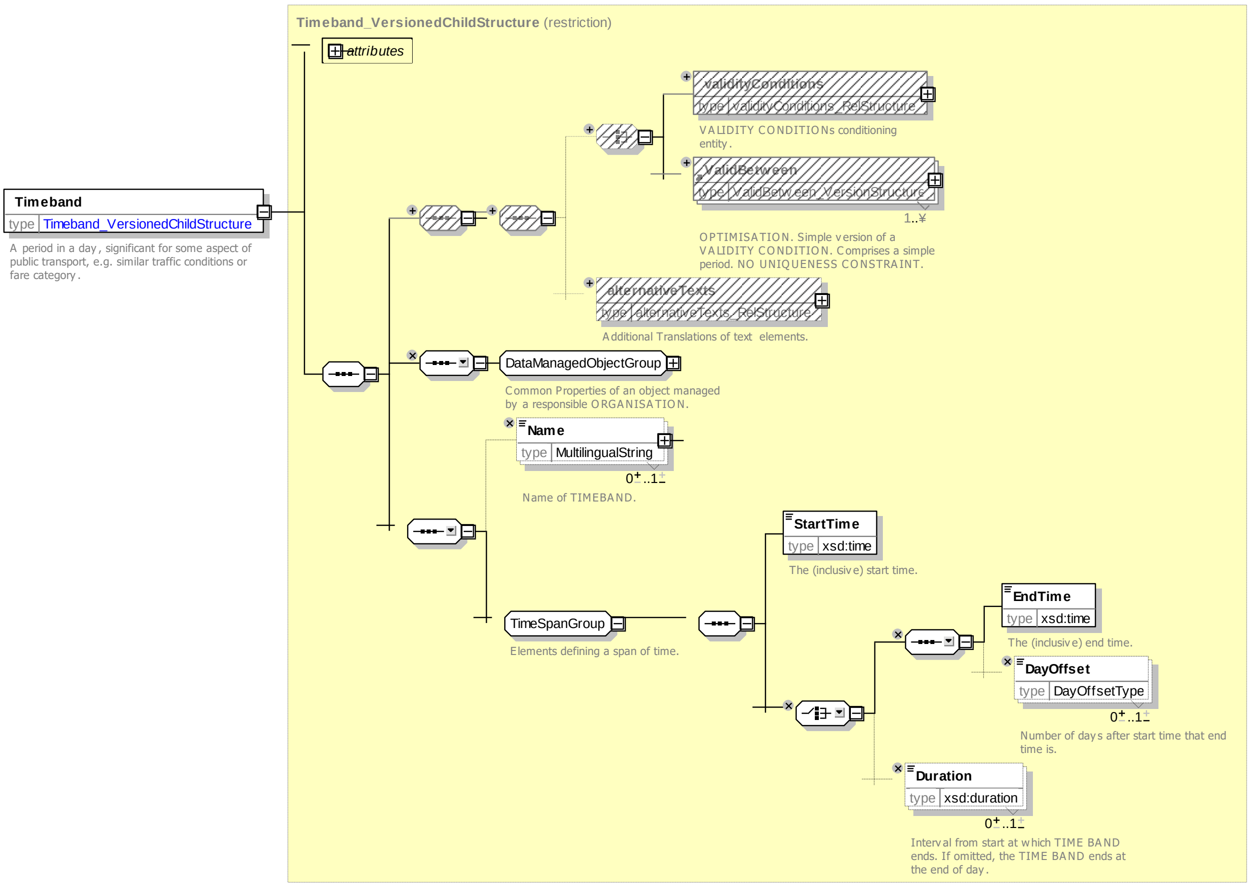


Figure 244 — *Timeband – XSD*

7.7.5.5.7 GroupOfTimebands – Model Element

A grouping of TIME BANDs.

Table 188 — *GroupOfTimebands – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF TIME BANDs inherits from GROUP OF ENTITies.
«PK»	***id***	*GroupOfTimebands-IdType*	1:1	Identifier of a GROUP OF TIME BANDs.
«cntd»	***timebands***	*Timeband*	0:*	TIMEBANDs of DAY TYPE.

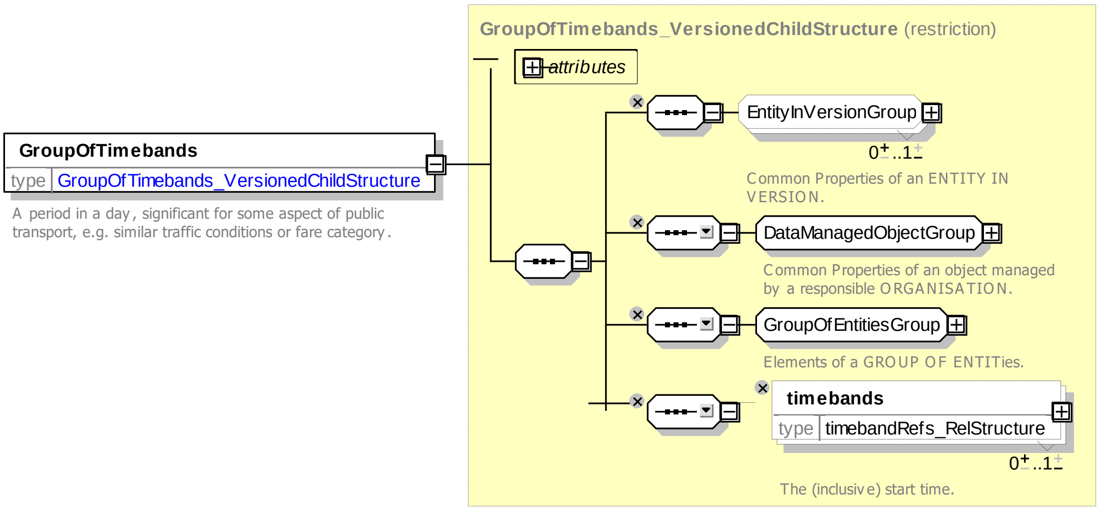


Figure 245 — *GroupOfTimebands – XSD*

7.7.5.6 Properties of Day – Physical Model

The DAY TYPE can be elaborated with a number of PROPERTIES OF DAY, defined with fixed enumerations of common properties.

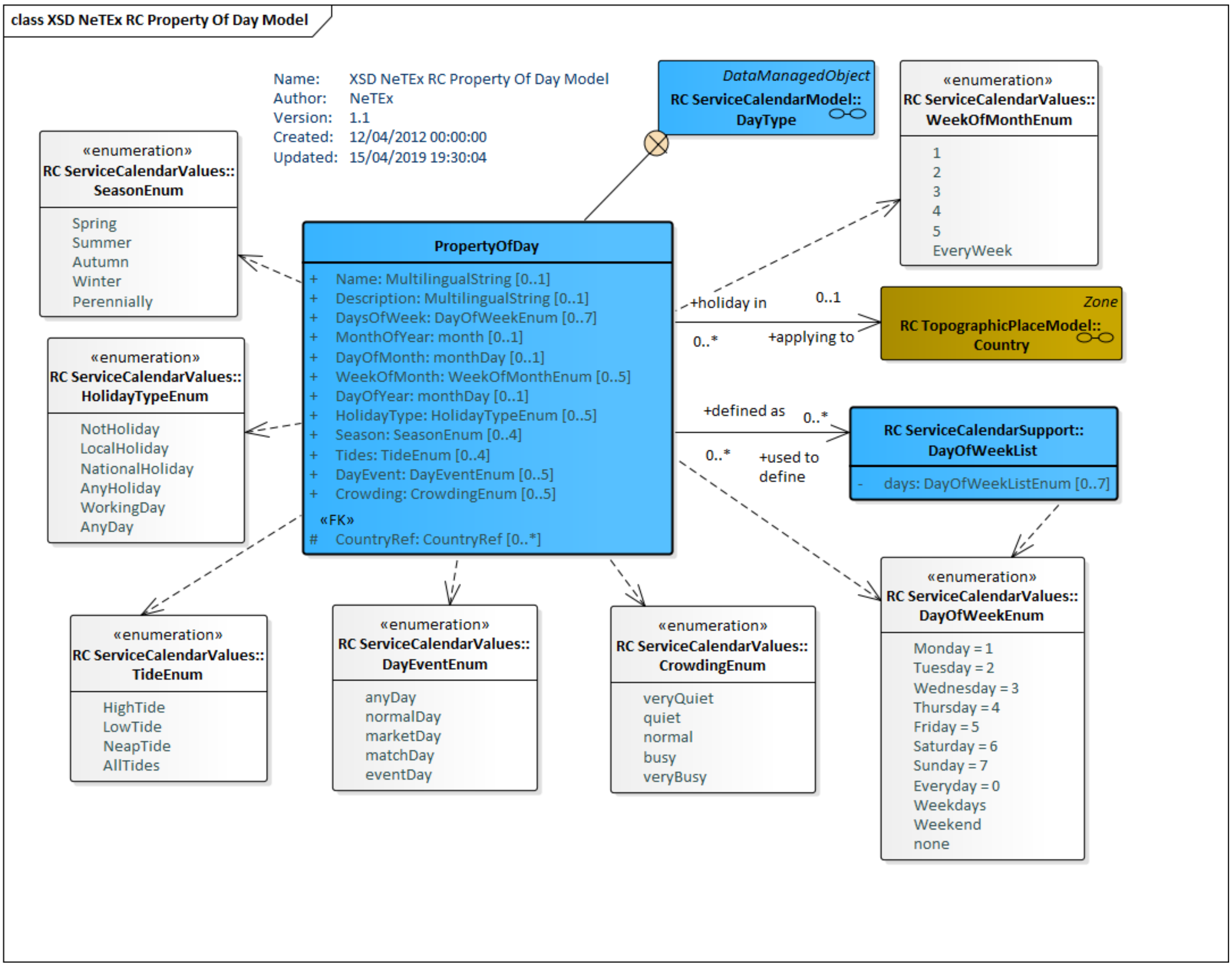


Figure 246 — Property Of Day – Physical Model (UML)

7.7.5.7 PropertyOfDay – Attributes and XSD

7.7.5.7.1 PropertyOfDay – Model Element

A property which a day may possess, such as school holiday, weekday, summer, winter etc. This may be used to generate a description of a day type in many different natural languages.

Table 189 — *PropertyOfDay – Element*

Classification	Name	Type	Cardinality	Description
	Name	*MultilingualString*	0:1	Name of PROPERTY OF DAY.
	Description	*MultilingualString*	0:1	Description of PROPERTY OF DAY.
«enum»	***DaysOfWeek***	*DayOfWeekEnum*	0:7	Days of week assigned to PROPERTY OF DAY.
«enum»	***WeeksOfMonth***	*WeekOfMonthEnum*	0:5	Weeks of month (1-5) assigned to PROPERTY OF DAY.
	a ***MonthOfYear***	*xsd:gMonth*	0:1	For those day types that occur on the same month every year; a month as a number.

	a	***DayOfMonth***	*xsd:gDay*	0:1	For those day types that occur on the same day of the month every year; a day as a number. +v1.1
	b	***DayOfYear***	*xsd:gMonthDay*	0:1	For those day types that occur on the same day every year, month and day, assigned to PROPERTY OF DAY.
«enum»	***CountryRef***		*CountryEnum*	0:*	Country of Holiday type for PROPERTY OF DAY.
«enum»	***HolidayTypes***		*HolidayTypeEnum*	0:5	Holiday type assigned to PROPERTY OF DAY. See allowed values below.
«enum»	***Seasons***		*SeasonEnum*	0:4	Season of year assigned to PROPERTY OF DAY. See allowed values below.
«enum»	***Tides***		*TideEnum*	0:4	State of tide assigned to PROPERTY OF DAY. See allowed values below.
«enum»	***DayEvent***		*DayEventEnum*	0:1	Events happening on day. See allowed values below.+V1.1
«enum»	***Crowding***		*CrowdingEnum*	0:1	How busy the day is considered to be. See allowed values below.+v1.1

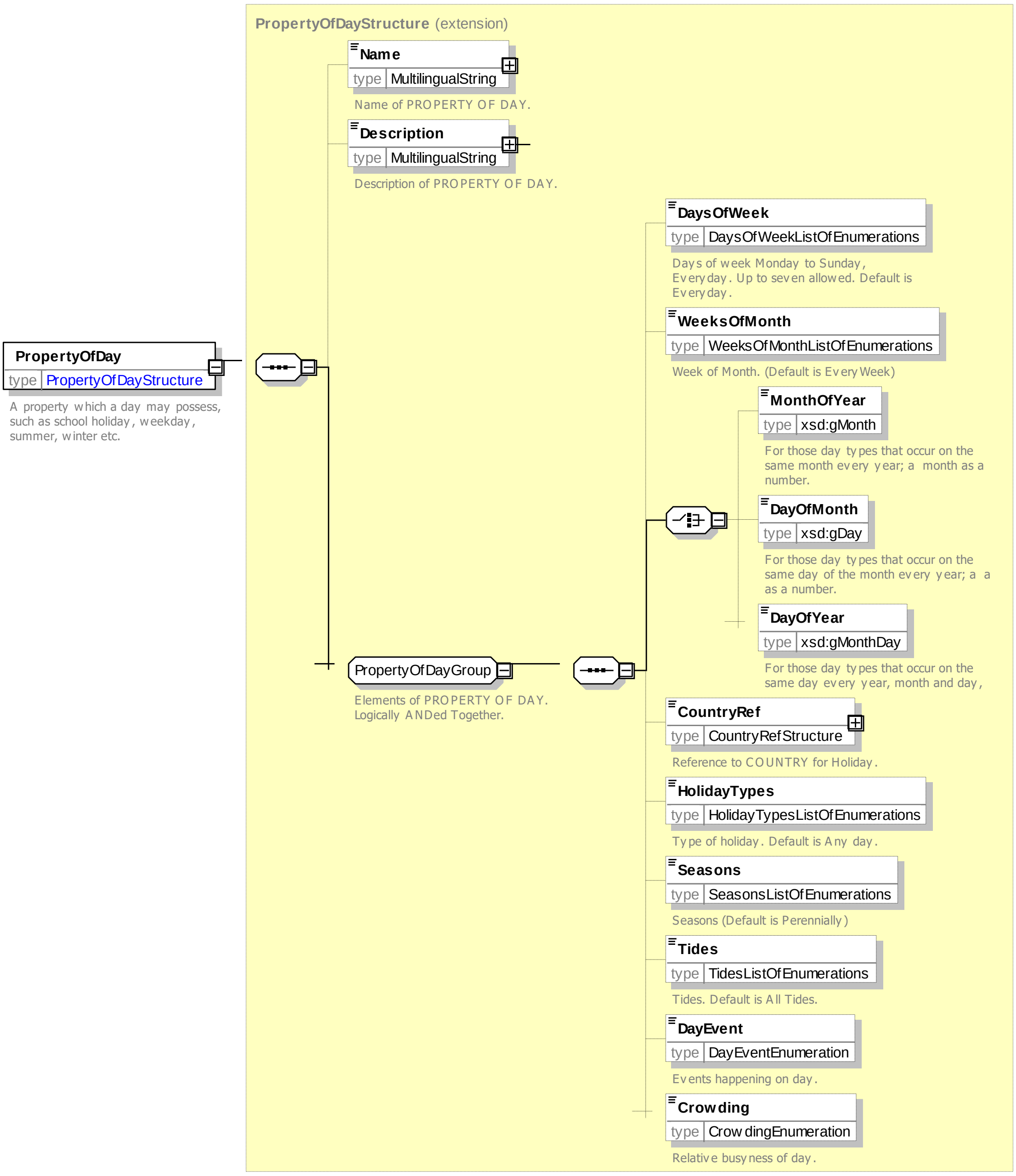


Figure 247 — *PropertyOfDay – XSD*

7.7.5.7.1.1 DayOfWeek – Allowed Values

Allowed values for DAY *OF WEEK* (*DayOfWeekEnumeration)*.

Table 190 — *DayOfWeek – AllowedValues*

Value	Description
Monday	Day of week is Monday.
Tuesday	Day of week is Tuesday.
Wednesday	Day of week is Wednesday.
Thursday	Day of week is Thursday.
Friday	Day of week is Friday.
Saturday	Day of week is Saturday.
Sunday	Day of week is Sunday.

Everyday	Day of week is Everyday.

7.7.5.7.1.2 **WeekOfMonth – Allowed Values**

Allowed values for Week of Month (*WeekOfMonthEnumeration)*.

Table 191 — *WeekOfMonth – AllowedValues*

Value	Description
1	First week of month.
2	Second week of month.
3	Third week of month.
4	Fourth week of month.
5	Fifth week of month.
Every Week	Every week of month.

7.7.5.7.1.3 **HolidayType – Allowed Values**

Allowed values for Holiday Type (*HolidayTypeEnumeration)*

Table 192 — *HolidayType – AllowedValues*

Value	Description
AnyDay	Day is of any type.
WorkingDay	Day is a working day.
SchoolDay	Day is a school day.
NotHoliday	Day is not a holiday.
NotWorkingDay	Day is not a working day.
NotSchoolDay	Day is not a school day.
AnyHoliday	Day is any type of holiday.
LocalHoliday	Day is a local holiday.
RegionalHoliday	Day is a Regional holiday.
NationalHoliday	Day is a national holiday.
Holiday-DisplacementDay	Day is a holiday displacement day (i.e. a day granted as a holiday because calendar holiday falls at weekend).
EveOfHoliday	Day is eve of a holiday.

7.7.5.7.1.4 **Season – Allowed Values**

Allowed values for *Season* (*SeasonEnumeration)*.

Table 193 — *Season – AllowedValues*

Value	Description
Spring	Spring season
Summer	Summer season
Autumn	Autumn season.
Winter	Winter season.
Perennially	All year round.

7.7.5.7.1.5 **Tides – Allowed Values**

Allowed values for ***Tide*** (*TideEnumeration)*.

Table 194 — *Tide – AllowedValues*

Value	Description
HighTide	High Tide.
LowTide	Low Tide.

NeapTide	Neap Tide.	*AllTides*	All tides.

7.7.5.7.1.6 Crowding – Allowed Values

Allowed values for ***Crowding*** (*CrowdingEnumeration)*.

Table 195 — *Crowding – AllowedValues*

Value	Description
veryQuiet	Day is characterised as very quiet.
quiet	Day is characterised as quiet.
normal	Day is characterised as normal.
busy	Day is characterised as busy.
veryBusy	Day is characterised as very busy.

7.7.5.7.2 DayOfWeek – Model Element

A particular week day (from Monday to Sunday). Implemenented with ***DayOfWeek*** enumeration.

Table 196 — *DayOfWeek – Element*

Classification	Name	Type	Cardinality	Description
«PK»	***Day***	*DayOfWeekRef*	1:1	Identifier of DAY OF WEEK.
	Name	*MultilingualString*	1:1	Name of DAY of WEEK.

7.7.5.8 XML Examples of Service Calendar

7.7.5.8.1 Basic Service Calendar with Day Type – *XML Example fragment*

The following example shows a very simple SERVICE CALENDAR with just a single DAY TYPE.

EXAMPLE

```
<ServiceCalendarFrame version="001" id="mybus:ntwkf001">
    <dayTypes>
        <DayType version="001" id="mybus:DT001_MF">
            <Name>Monday to Friday</Name>
            <properties>
                <PropertyOfDay>
                    <DaysOfWeek>Monday Tuesday Wednesday Thursday Friday</DaysOfWeek>
                    <HolidayTypes>WorkingDay</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
    </dayTypes>
</ServiceCalendarFrame>
```

7.7.5.8.2 Service Calendar with Timebands & Assignments – *XML Example fragment*

The following example (simplified from the London 2012 Olympics) shows the use of a SERVICE CALENDAR with two DAY TYPEs that are assigned to specific dates with DAY TYPE ASSIGNMENTs. The DAY TYPEs themselves have TIMEBANDs.

EXAMPLE

```
<!-- - ==== SERVICE CALENDAR === -->
<ServiceCalendarFrame version="any" id="oda:CAL_01">
    <Name>Olympic Park2 2012 </Name>
    <ServiceCalendar version="any" id="oda:CAL_01">
        <FromDate>2012-07-27</FromDate>
        <ToDate>2012-08-07</ToDate>
    </ServiceCalendar>
    <dayTypes>
        <DayType version="any" id="oda:DT02_Busy">
            <Name>Busy Day</Name>
            <timebands>
                <TimebandRef version="any" ref="oda:TMB01_ingress"/>
                <TimebandRef version="any" ref="oda:TMB02_during"/>
                <TimebandRef version="any" ref="oda:TMB03_egress"/>
            </timebands>
        </DayType>
        <DayType version="any" id="oda:DayType:DT03_Normal">
            <Name>Normal Day</Name>
            <timebands>
                <TimebandRef version="any" ref="oda:TMB01_ingress"/>
                <TimebandRef version="any" ref="oda:TMB02_during"/>
                <TimebandRef version="any" ref="oda:TMB03_egress"/>
            </timebands>
        </DayType>
        <DayType version="any" id="oda:DayType:DT04_QUIET">
            <Name>Quiet Day</Name>
            <properties>
                <PropertyOfDay>
                    <Crowding>quiet</Crowding>
                </PropertyOfDay>
            </properties>
            <timebands>
                <TimebandRef version="any" ref="oda:TMB01_ingress"/>
                <TimebandRef version="any" ref="oda:TMB02_during"/>
                <TimebandRef version="any" ref="oda:TMB03_egress"/>
            </timebands>
        </DayType>
    </dayTypes>
    <timebands>
        <Timeband version="any" id="oda:TMB01_ingress">
            <StartTime>09:00:00</StartTime>
            <EndTime>10:00:00</EndTime>
        </Timeband>
        <Timeband version="any" id="oda:TMB02_during">
            <StartTime>11:00:00</StartTime>
            <EndTime>15:00:00</EndTime>
        </Timeband>
        <Timeband version="any" id="oda:TMB03_egress">
            <StartTime>15:00:00</StartTime>
            <EndTime>19:00:00</EndTime>
        </Timeband>
    </timebands>
    <dayTypeAssignments>
        <DayTypeAssignment version="any" id="oda:DayAsgn_2012-07-27">
            <Description>Friday 2012-07-27 Opening</Description>
            <Date>2012-07-27</Date>
            <DayTypeRef version="any" ref="oda:DT02_ Busy"/>
        </DayTypeAssignment>
        <DayTypeAssignment version="any" id="oda:DayAsgn_2012-07-28">
            <Description>Saturday 2012-07-28</Description>
            <Date>2012-07-28</Date>
            <DayTypeRef version="any" ref="oda:DT02_ Busy"/>
        </DayTypeAssignment>
        <DayTypeAssignment version="any" id="oda:DayAsgn_2012-07-29">
:>.
        <DayTypeAssignment version="any" id="oda:DayAsgn_2012-07-30">
            <Description>Monday 2012-07-30</Description>
            <Date>2012-07-30</Date>
            <DayTypeRef version="any" ref="oda:DT03_Normal"/>
        </DayTypeAssignment>
        <DayTypeAssignment version="any" id="oda:DayAsgn_2012-07-31">
            <Description>Tuesday 2012-07-31</Description>
            <Date>2012-07-31</Date>
            <DayTypeRef version="any" ref="oda:DT03_Normal"/>
:>….
        <DayTypeAssignment version="any" id="oda:DayAsgn_2012-08-05">
```

```xml
                <Description>Sunday 2012-08-05</Description>
                <Date>2012-08-05</Date>
                <DayTypeRef version="any" ref="oda:DT02_Busy"/>
            </DayTypeAssignment>
        </dayTypeAssignments>
    </ServiceCalendarFrame>
```

7.7.5.8.3 Service Calendar with assignments – *XML Example fragment*

The following example shows the use of DAY TYPEs to specify some holidays, OPERATING DAY to specify the working day for the system.

EXAMPLE

```xml
<ServiceCalendarFrame version="1" id="hde:CAL_02">
    <Name>Service Calendar Nov to April 2010 (Compact Coding) </Name>
    <ServiceCalendar version="any" id="hde:CAL_02">
        <FromDate>2010-11-01</FromDate>
        <ToDate>2010-11-14</ToDate>
    </ServiceCalendar>
    <dayTypes>
        <DayType version="any" id="hde:DT_01-MF-Not+Holiday">
            <Name>Weekdays unless a holiday</Name>
            <properties>
                <PropertyOfDay>
                    <DaysOfWeek>Monday Tuesday Wednesday Thursday Friday</DaysOfWeek>
                    <HolidayTypes>NotHoliday</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <DayType version="any" id="hde:DT_02-Everyday-Not_Holiday">
            <Name>Everyday unless a holiday</Name>
            <properties>
                <PropertyOfDay>
                    <DaysOfWeek>Everyday</DaysOfWeek>
                    <HolidayTypes>NotHoliday</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <DayType version="any" id="hde:DT_04-AA-Not_Holiday">
            <Name>Holidays</Name>
            <properties>
                <PropertyOfDay>
                    <DaysOfWeek>Everyday</DaysOfWeek>
                    <HolidayTypes>AnyHoliday</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <!-- === holidays ==== -->
        <DayType version="any" id="hde:DT_Christmas_Eve">
            <Name>Christmas Eve </Name>
            <properties>
                <PropertyOfDay>
                    <DayOfYear>12-24</DayOfYear>
                    <HolidayTypes>EveOfHoliday</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <DayType version="any" id="hde:DT_Christmas_Day">
            <Name>Christmas Day </Name>
            <properties>
                <PropertyOfDay>
                    <DayOfYear>12-25</DayOfYear>
                    <HolidayTypes>NationalHoliday</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <DayType version="any" id="hde:DT_Christmas_Day_Displacement">
            <Name>Christmas Day Displacement holiday if it happens on a weekday</Name>
            <properties>
                <PropertyOfDay>
                    <DaysOfWeek>Monday Tuesday Wednesday Thursday Friday</DaysOfWeek>
                    <HolidayTypes>HolidayDisplacementDay</HolidayTypes>
```

```xml
                </PropertyOfDay>
            </properties>
        </DayType>
        <!-- === holidays New year==== -->
        <DayType version="any" id="hde:DT_NewYearsDay">
            <Name>NewYear's Day </Name>
            <properties>
                <PropertyOfDay>
                    <DayOfYear>01-01</DayOfYear>
                    <HolidayTypes>NationalHoliday </HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <DayType version="any" id="hde:DayType:DT_NewYearsDay_Displacement">
            <Name>NewYear's Day Displacement holiday </Name>
            <properties>
                <PropertyOfDay>
                    <DaysOfWeek>Monday Tuesday Wednesday Thursday Friday</DaysOfWeek>
                    <HolidayTypes> HolidayDisplacementDay</HolidayTypes>
                </PropertyOfDay>
            </properties>
        </DayType>
        <!-- === holidays Easter year==== -->
    </dayTypes>
    <operatingDays>
        <OperatingDay version="any" id="hde:OD_2010-11-01">
            <CalendarDate>2010-11-01</CalendarDate>
            <Name>Monday 2010-11-01</Name>
            <EarliestTime>02:00:00</EarliestTime>
            <DayLength>PT24H</DayLength>
        </OperatingDay>
        <OperatingDay version="any" id="hde:OD_2011-04-30">
            <CalendarDate>2011-04-30</CalendarDate>
            <Name>Saturday 2011-04-30</Name>
            <EarliestTime>02:00:00</EarliestTime>
            <DayLength>PT24H</DayLength>
        </OperatingDay>
    </operatingDays>
    <operatingPeriods>
        <OperatingPeriod version="any" id="hde:op_010">
            <Name>Winter 2011</Name>
            <FromOperatingDayRef version="any" ref="hde:OD_2010-11-01"/>
            <ToOperatingDayRef version="any" ref="hde:OD_2011-04-30"/>
            <HolidayType>NotHoliday</HolidayType>
            <Season>Winter</Season>
        </OperatingPeriod>
    </operatingPeriods>
```

7.7.5.8.4 Service Calendar with Operating Periods – *XML Example fragment*

The following example shows the use of a SERVICE CALENDAR to defined teh OPERATING PERIODs of a SERVICE ORGANISATION.

EXAMPLE

```xml
<ServiceCalendarFrame version="1" id="dth:DTH_01">
    <Name>Dotheboys Hall School Terms </Name>
    <ServiceCalendar version="any" id="dth:DTH_01">
        <FromDate>2010-09-01</FromDate>
        <ToDate>2011-08-01</ToDate>
    </ServiceCalendar>
    <operatingPeriods>
        <OperatingPeriod version="1" id="dth:OP_01">
            <Name>Autumn term </Name>
            <FromDate>2010-09-01T00:00:00</FromDate>
            <ToDate>2010-09-24T00:00:00</ToDate>
            <HolidayType>SchoolDay</HolidayType>
        </OperatingPeriod>
        <OperatingPeriod version="1" id="dth:OP_02">
            <Name>Spring term </Name>
            <FromDate>2011-01-01T00:00:00</FromDate>
            <ToDate>2010-04-01T00:00:00</ToDate>
            <HolidayType>SchoolDay</HolidayType>
```

```
            </OperatingPeriod>
            <OperatingPeriod version="1" id="dth:OP_03">
                <Name>Summer term </Name>
                <FromDate>2011-04-11T00:00:00</FromDate>
                <ToDate>2010-08-01T00:00:00</ToDate>
                <HolidayType>SchoolDay</HolidayType>
            </OperatingPeriod>
        </operatingPeriods>
    </ServiceCalendarFrame>
```

7.7.6 Availability Condition

7.7.6.1 AVAILABILITY CONDITION – Conceptual MODEL

AVAILABILITY CONDITION is a specialisation of VALIDITY CONDITION to specify precise temporal conditions. For example, an ENTRANCE of a STOP PLACE may be valid (it exists) but not available for some of the time (it is closed between 9 pm and 6 am). Both VALIDITY CONDITIONs and AVAILABILITY CONDITIONs may be associated for the same entity.

An AVAILABILITY CONDITION can be defined by specific DAY TYPEs and/or OPERATING DAYs. It may be further qualified by one or more of TIME BANDs. The DATED AVAILABILITY CONDITION being the instance of VALIDITY CONDITION on a specific CALENDAR DAY.

Examples of use of AVAILABILITY CONDITION include ENTRANCEs, EQUIPMENTs, STOP PLACEs, etc.

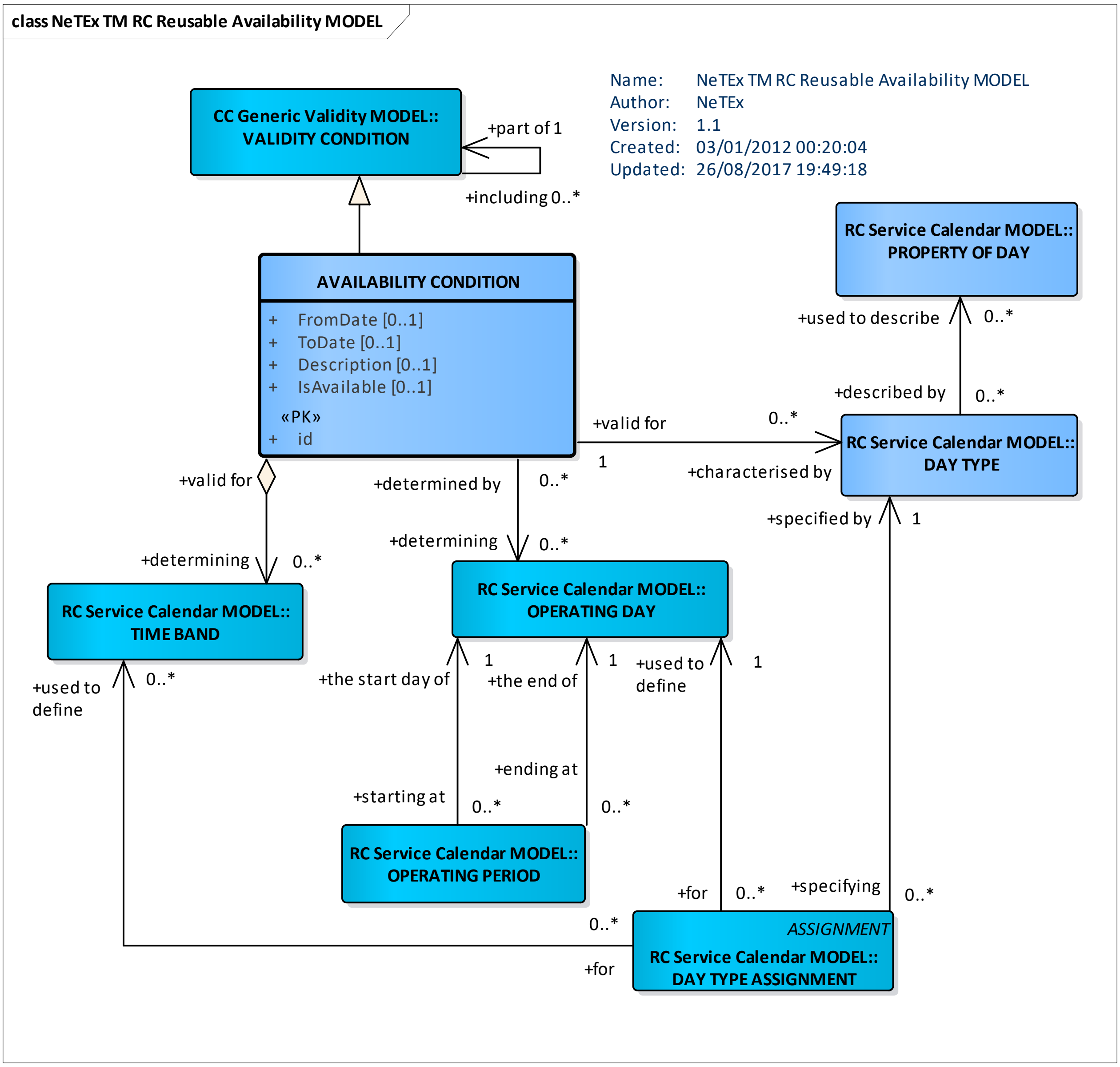


Figure 248 — Reusable Availability – Conceptual MODEL (UML)

7.7.6.2 Availability Condition – Physical Model

The following figure shows the Physical Model for an AVAILABILITY CONDITION.

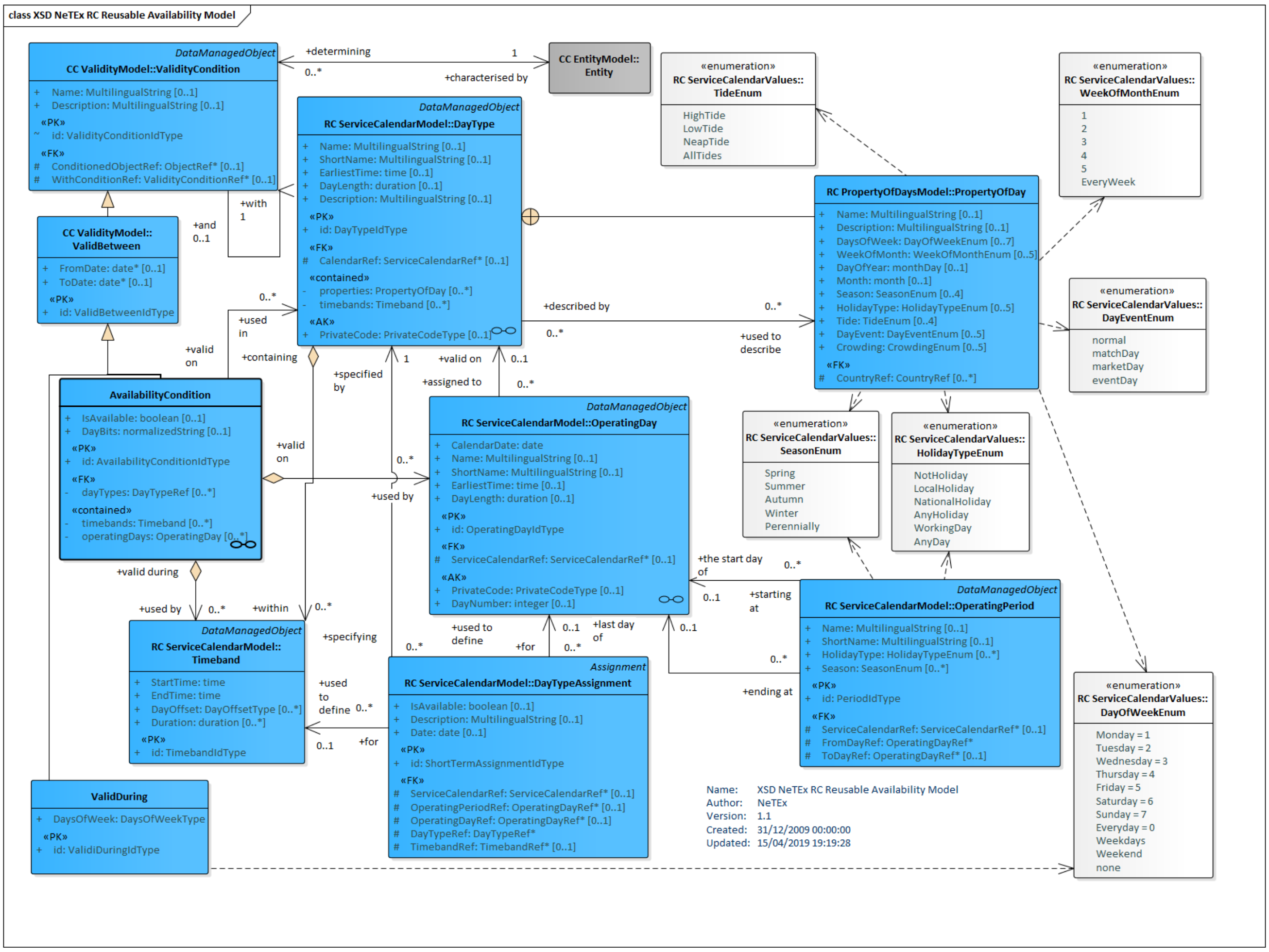


Figure 249 — Reusable Availability – Physical Model (UML)

7.7.6.3 Availability Condition – Attributes and XSD

7.7.6.3.1 AvailabilityCondition – Model Element

A specific type of VALIDITY CONDITION used to specify a set of temporal conditions that can be associated with an ENTITY, for example that a STOP PLACE is open on a particular DAY TYPE.

Table 197 — *AvailabilityCondition – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*ValidityCondition*	::>	AVAILABILITY CONDITION inherits from VALIDITY CONDITION.
«PK»	***id***	*AvailabilityConditionIdType*	1:1	Identifier of AVAILABILITY CONDITION.
	FromDate	*xsd:dateTime*	0:1	Inclusive start date for validity of AVAILABILITY CONDITION.
	ToDate	*xsd:dateTime*	0:1	Inclusive end date for validity of AVAILABILITY CONDITION.

	IsAvailable	*xsd:boolean*	1:1	Whether the AVAILABILITY CONDITION makes the resource available or not available.
«FK»	***dayTypes***	*DayTypeRef*	0:*	DAY TYPEs during which AVAILABILITY CONDITION applies.
	validDayBits	*bitString*	0:1	Alternative way of stating the validity of days within the start and end date; 1 = *valid* and 0 = *not valid*. There must be one bit for each day.
«cntd»	***timeBands***	*TimeBand*	0:*	TIME BANDs during which AVAILABILITY CONDITION applies.

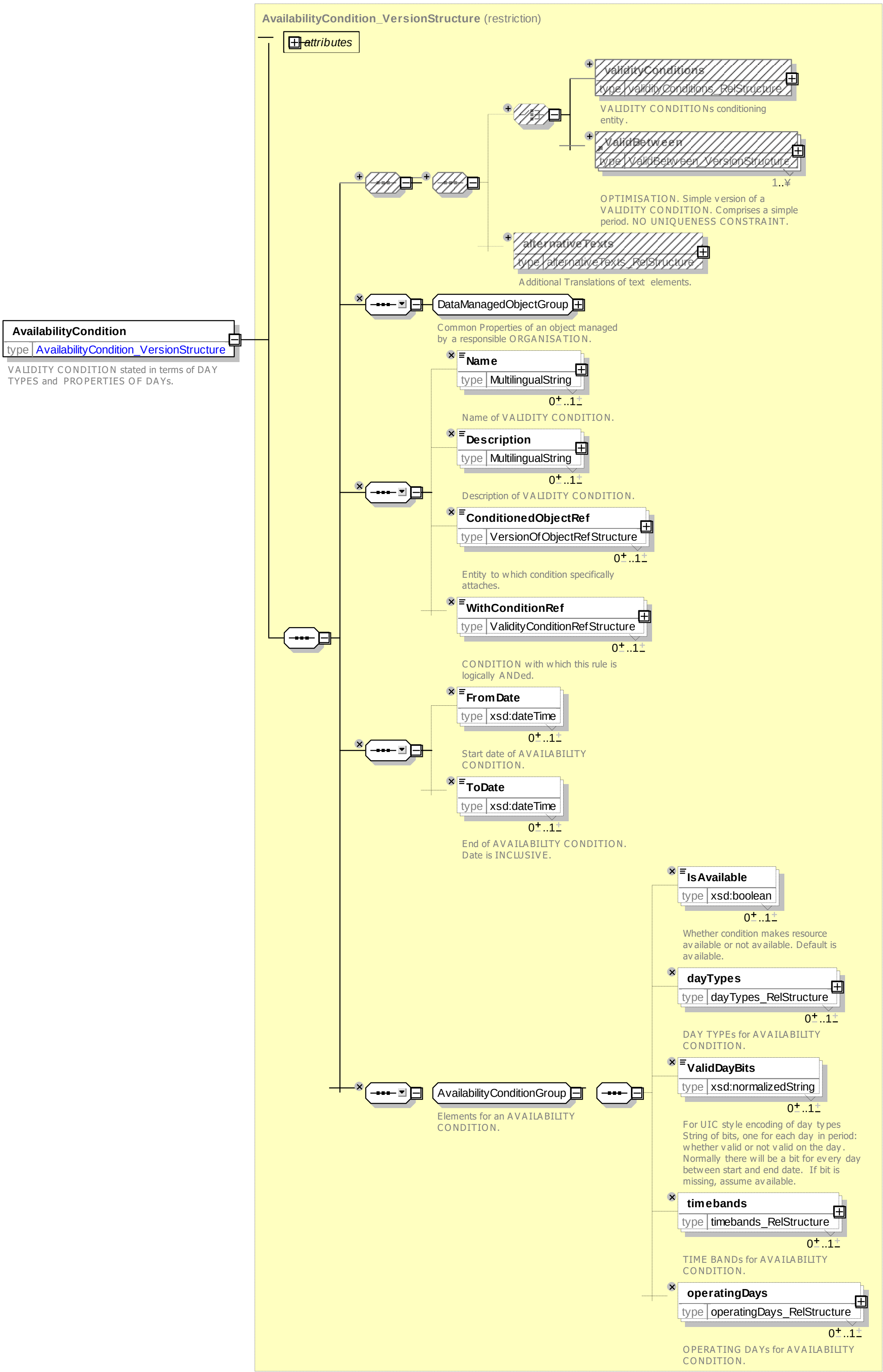


Figure 250 — *AvailabilityCondition – XSD*

7.7.6.4 XML Examples of Availability Condition

7.7.6.4.1 Basic Service Calendar with Day Type – *XML Example fragment*

The following example shows an AVAILABILITY CONDITION that applies on two DAY TYPE.

EXAMPLE

```xml
<AvailabilityCondition id="mybus:VC002_mf">
    <Description>Use when no ice</Description>
    <FromDate>2010-05-17T00:00:00Z</FromDate>
    <ToDate>2011-05-17T00:00:00Z</ToDate>
    <dayTypes>
        <DayTypeRef version="001"  ref="mybus:DT001_MF"/>
        <DayTypeRef version="001"  ref="mybus:DT002_MFHols"/>
    </dayTypes>
</AvailabilityCondition>
```

7.7.7 Topographic Place

7.7.7.1 TOPOGRAPHIC PLACE – Conceptual MODEL

The TOPOGRAPHIC PLACE model represents the name settlements and other places to which PT data may be related. It also includes a Postal and Road Address model, effectively providing a Gazetteer which can be used for stop finding and other purposes.

A TOPOGRAPHIC PLACE can be located within a COUNTRY. TOPOGRAPHIC PLACEs may overlap. They may also be contained inside another TOPOGRAPHIC PLACE.

ROAD ADDRESS and POSTAL ADDRESS can also be located within a COUNTRY.

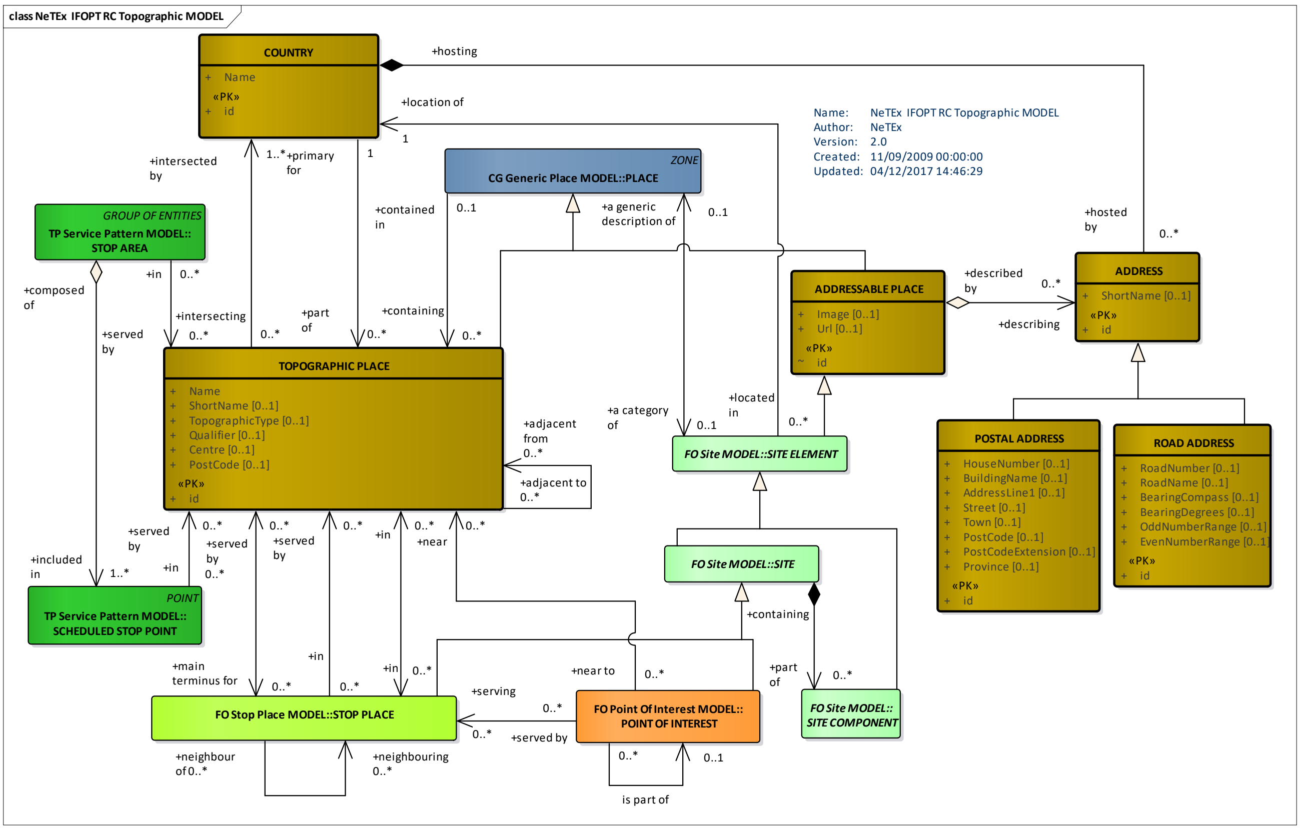


Figure 251 — Topographic Place – Conceptual MODEL (UML)

7.7.7.2 TopographicPlace – Physical Model

The following figure shows detailed attributes of the TOPOGRAPHIC PLACE elements.

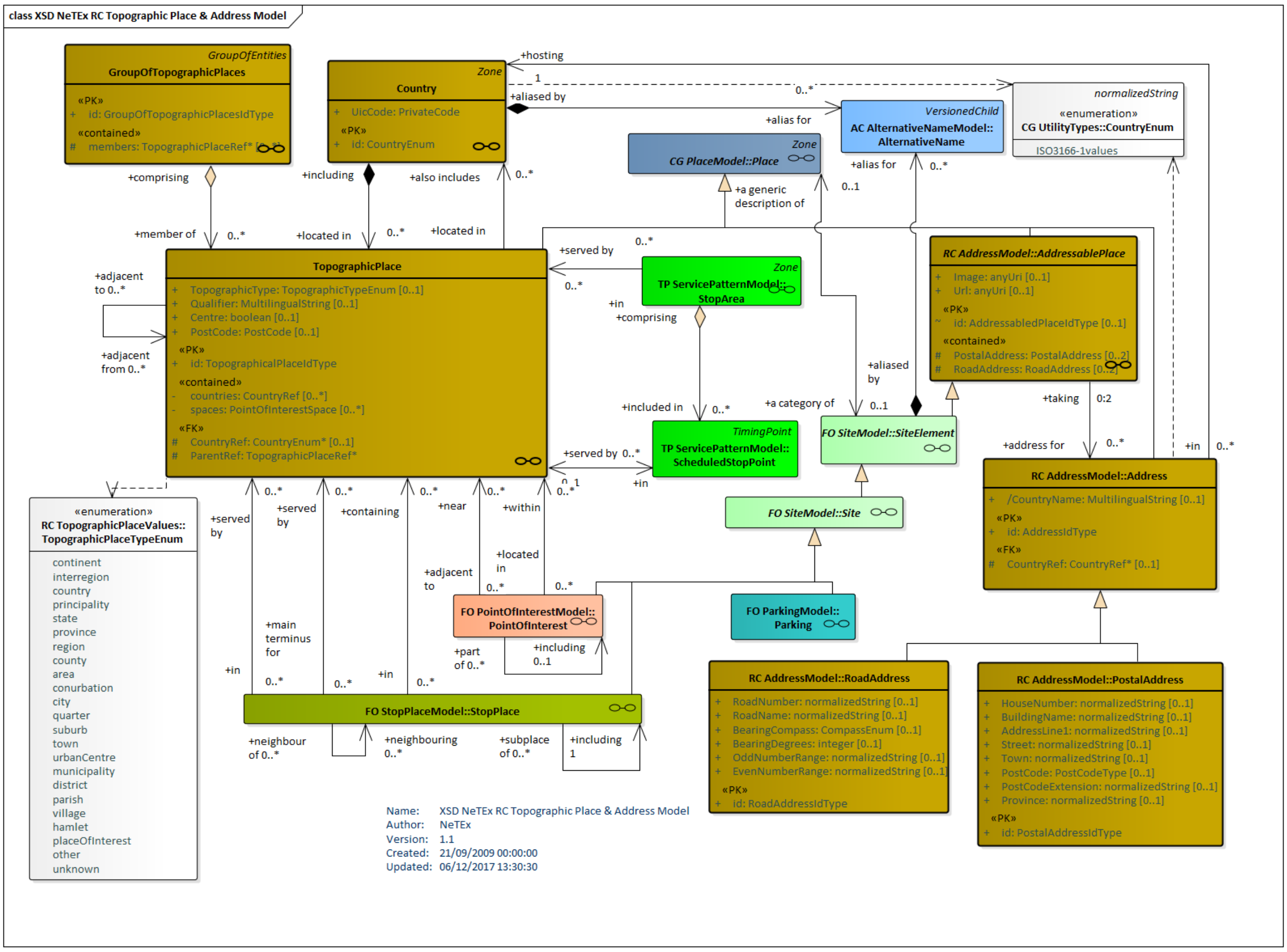


Figure 252 — Topographic Place – Physical Model (UML)

7.7.7.2.1 TopographicPlaceProjection – Physical Model

The TOPOGRAPHIC PLACE PROJECTION model allows any ZONE to be associated with a TOPOGRAPHICAL PLACE, for example a TARIFF ZONE, ADMINISTRATIVE ZONE or ACCESS ZONE can be stated to be equivalent to a TOPOGRAPHICAL PLAE.

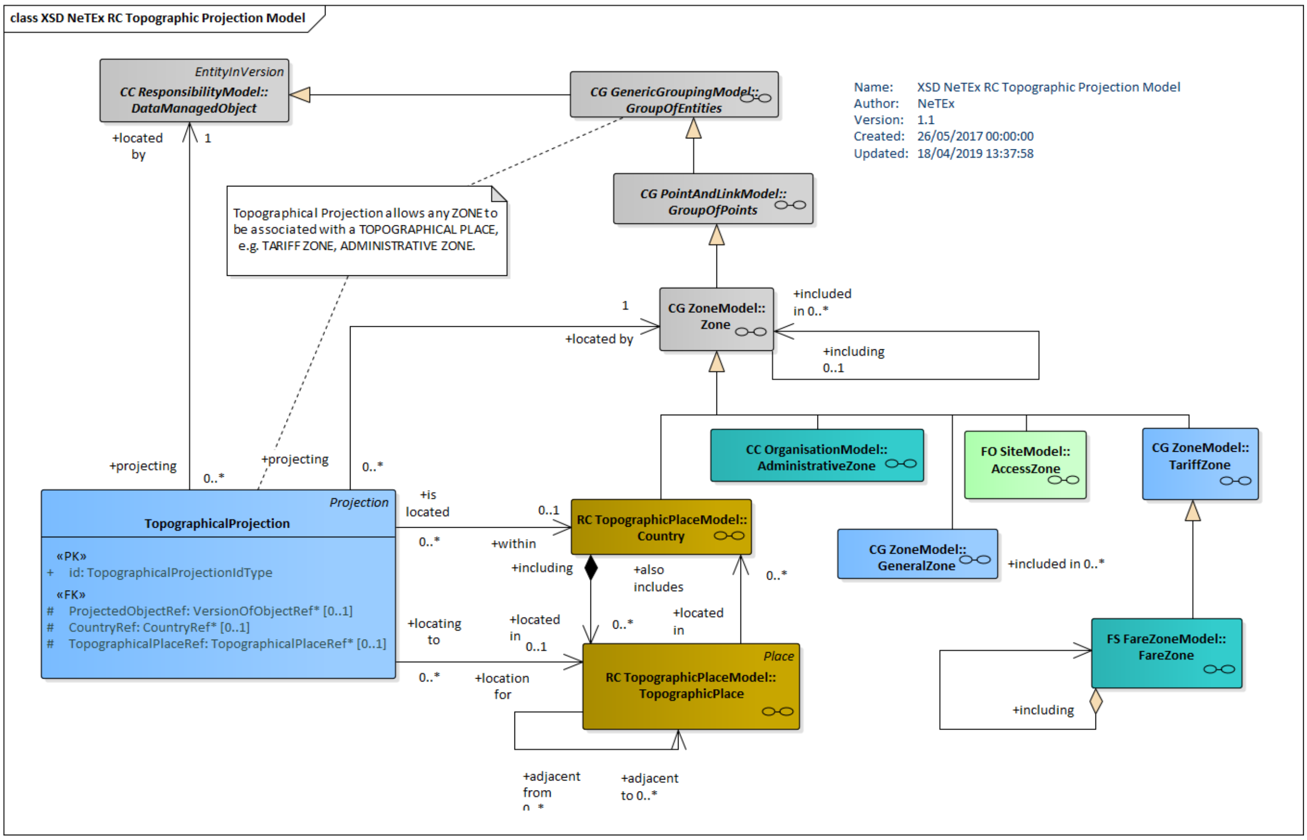


Figure 253 — Topographic Place Projection – Physical Model (UML)

7.7.7.3 Topographic Place – Attributes and XSD

7.7.7.3.1 Country – Model Element

A jurisdictional geographic boundary. A COUNTRY normally has a two character IANA identifier.

Table 198 — *Country – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Place*</u>	::>	COUNTRY inherits from PLACE.
«PK»	***id***	*CountryCodeType*	1:1	Identifier of COUNTRY.
«cntd»	***uicCode***	*PrivateCodeStructure*	01	UIC Code for Country.
«cntd»	***alternativeNames***	<u>*AlternativeName*</u>	0:*	Alternative Names for Country.

Country Ids are implemented by an enumeration of allowed values.

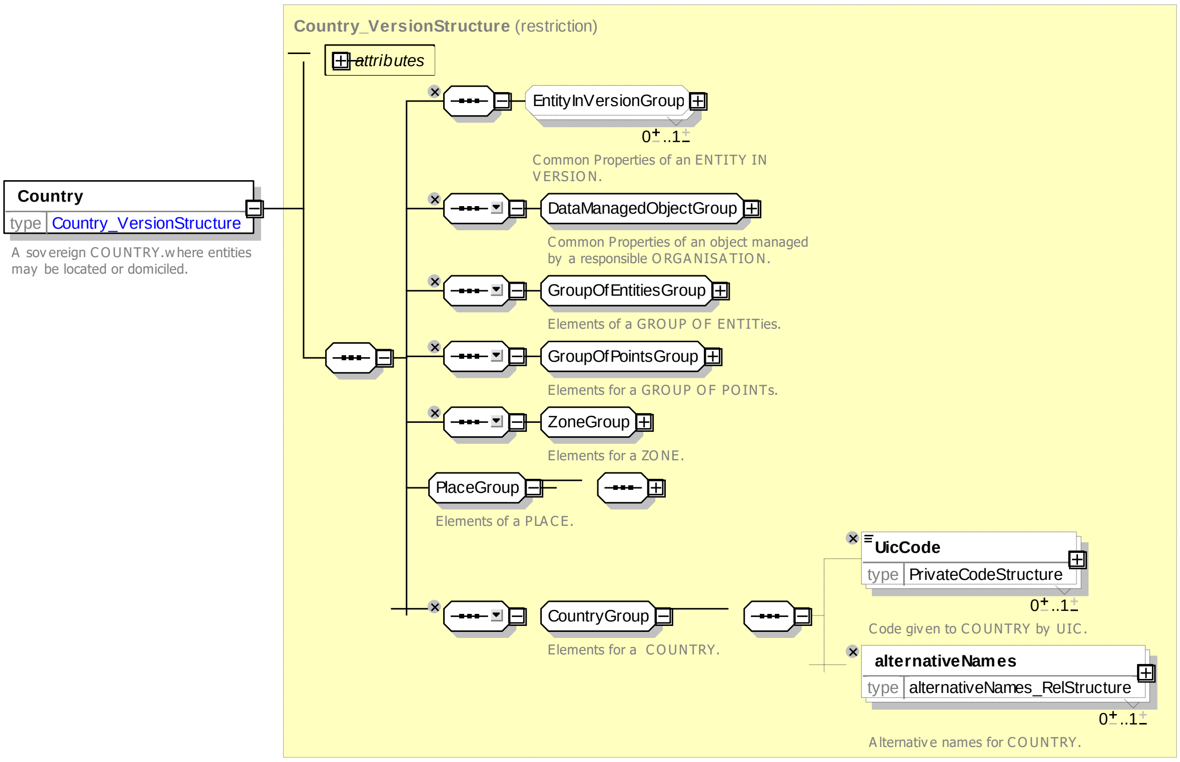


Figure 254 — *Country – XSD*

7.7.7.3.2 TopographicPlace – Model Element

A geographical settlement which provides topographical context when searching for or presenting travel information, for example as the origin or destination of a trip. It may be of any size (e.g. County, City, Town, Village) and of different specificity e.g. '*Greater London', 'London', 'West End', 'Westminster', 'St James's'*.

A TOPOGRAPHICAL PLACE must always have a canonical gazetteer name. It may be necessary to use the hierarchical topographical relationships of the TOPOGRAPHICAL PLACE to establish a unique context with which to distinguish between two TOPOGRAPHICAL PLACEs with the same name.

Table 199 — *TopographicPlace – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Place*	::>	TOPOGRAPHIC PLACE inherits from PLACE.
«PK»	***id***	*TopographicalPlaceIdType*	1:1	Identifier of a TOPOGRAPHIC PLACE.
XGRP	***TopographicPlace DescriptiveGroup***	***xmlGroup***	0:1	Elements describing the TOPOGRAPHIC PLACE.
XGRP	***TopographicPlace RelationsGroup***	***xmlGroup***	0:1	Elements describing the relationsgip of the TOPOGRAPHIC PLACE to other entities,.

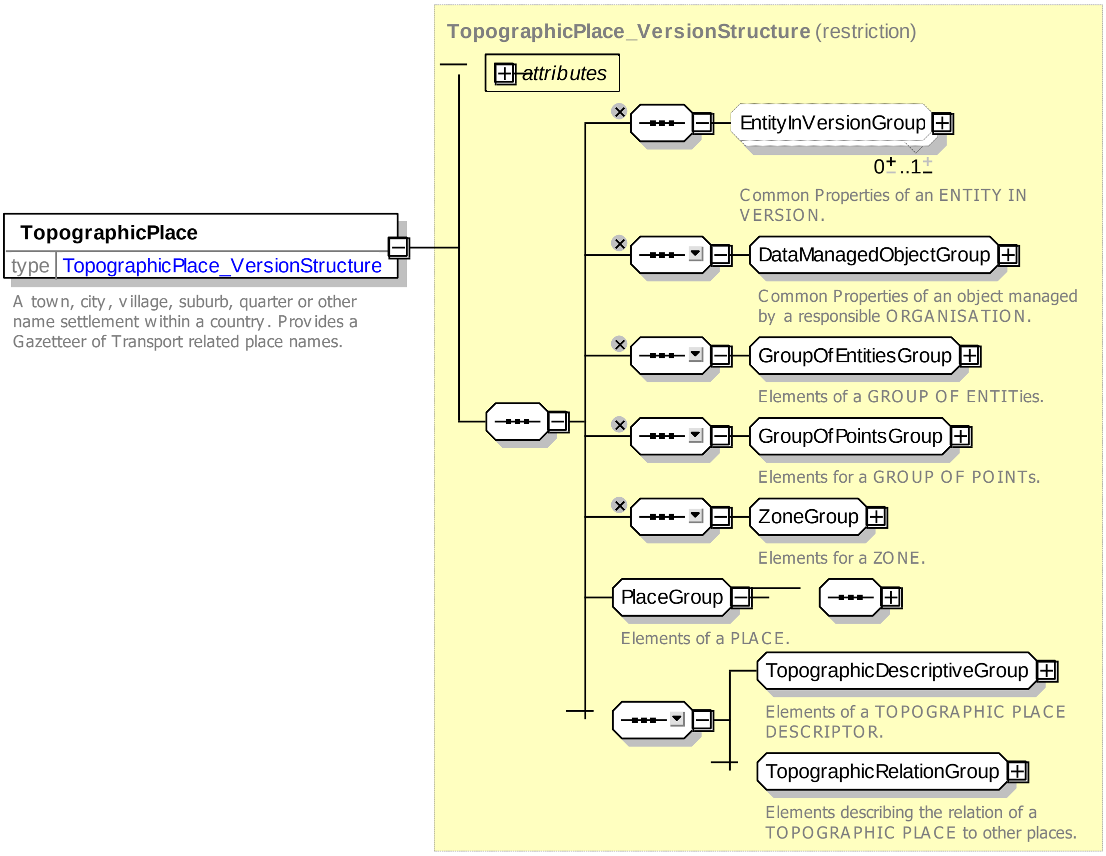


Figure 255 — *TopographicPlace – XSD*

7.7.7.3.2.1 TopographicDescriptiveGroup – XML Group

Elements describing the name of a TOPOGRAPHIC PLACE.

Table 200 — *TopographicPlaceDescriptorGroup – Group*

Classification	Name	Type	Cardinality	Description
«AK»	***IsoCode***	*IsoSubdvisionCodeType*	0:1	ISO 3166-22 code for a TOPOGRAPHIC PLACE. +v1.1
	Descriptor	*Descriptor*	1:1	Descriptiive elements for a TOPOGRAPHIC PLACE.
«cntd»	***Alternative-Descriptors***	*AlternativeDescriptor*	0:*	Alternative descriptors for TOPOGRAPHIC PLACE.
«enum»	***TopographicPlace Type***	*TopographicTypeEnum*	0:1	Type of a TOPOGRAPHIC PLACE. See allowed values below.
	PlaceCentre	*xsd:boolean*	0:1	Whether TOPOGRAPHIC PLACE lies at centre of settlement.
	PostCode	*xsd:normalizedString*	0:1	Post code or partial post code associated with area. +v1.1

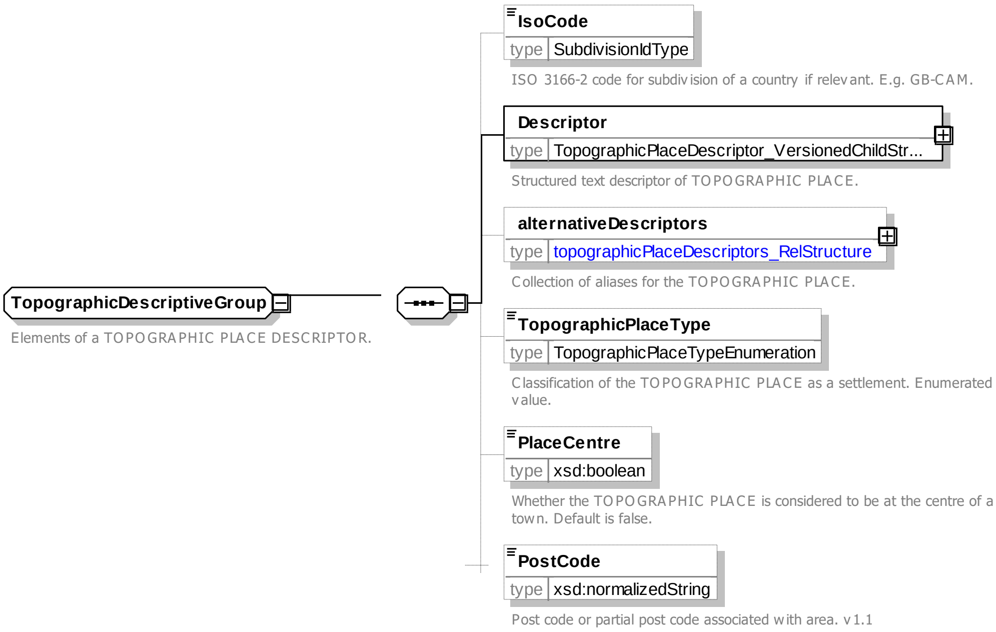


Figure 256 — *TopographicDescriptiveGroup – XSD*

7.7.7.3.2.2 TopographicRelationGroup – XML Group

Elements relating the TOPOGRAPHIC PLACE to other spatially relevant entities.

Table 201 — *TopographicRelationGroup – Group*

Classification	Name	Type	Cardinality	Description
«FK»	***CountryRef***	*CountryRef*	0:1	Reference to COUNTRY of a TOPOGRAPHIC PLACE.
«ctnd»	***otherCountries***	*CountryRef*	0:*	For TOPOGRAPHIC PLACEs thats span borders, references to additional COUNTRY or COUNTRIEs that place lies in.
«FK»	***Parent-TopographicPlaceRef***	*TopographicPlaceRef*	0:1	Reference to a parent TOPOGRAPHIC PLACE that contains the TOPOGRAPHIC PLACE.
«ctnd»	***adjacentPlaces***	*TopographicPlaceRef*	0:*	References to other TOPOGRAPHIC PLACE that are immediately adjacent to the TOPOGRAPHIC PLACE.
«ctnd»	***containedIn***	*TopographicPlaceRef*	0:*	References to other TOPOGRAPHIC PLACE within which the TOPOGRAPHIC PLACE is contained.
«ctnd»	***accesses***	*AccessRef*	0:*	References to ACCESS links that relate to the TOPOGRAPHIC PLACE.

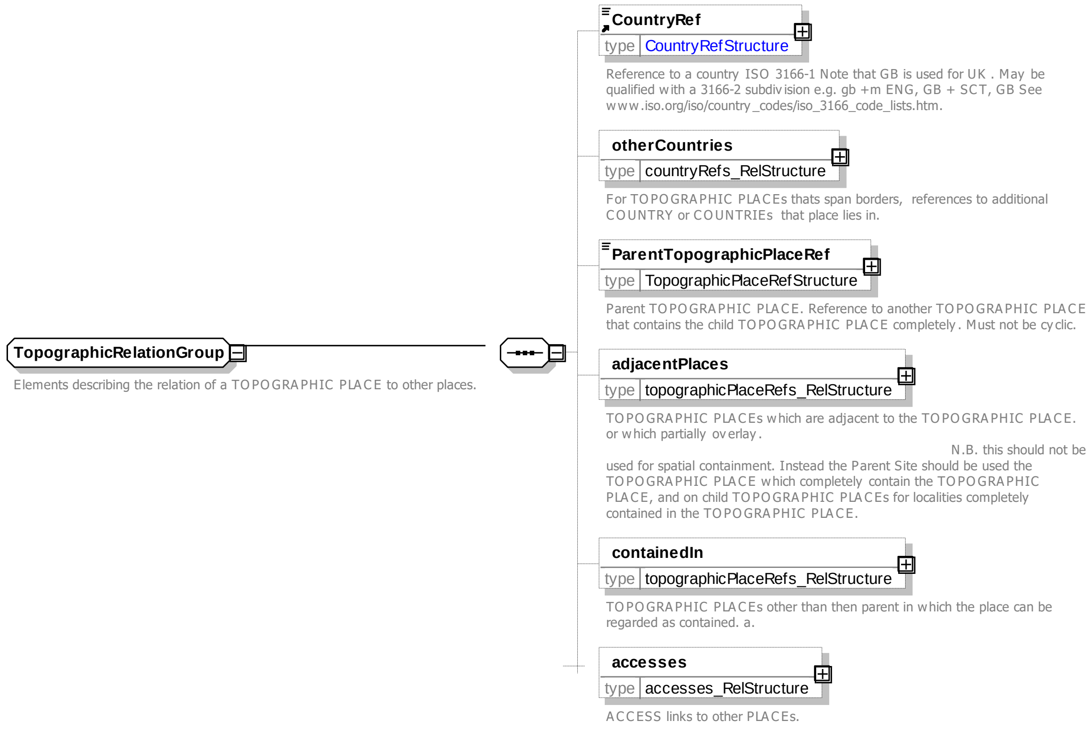


Figure 257 — *TopographicRelationGroup – XSD*

7.7.7.3.2.3 TopographicPlaceType – Allowed values

The following table shows the allowed values for *TopographicPlaceType* **(***TopographicPlaceTypeEnum***).**

Table 202 – *TopographicPlaceType* – Allowed Values

Value	Description
continent	PLACE is a continent. + V1.1
interregion	PLACE is a interregion. +V1.1
country	PLACE is a country. +V1.1
continent	PLACE is a continent
principality	PLACE is a principality
state	PLACE is a state
province	PLACE is a province
region	PLACE is a region
county	PLACE is a county
area	PLACE is an arbitrary area
conurbation	PLACE is a conurbation
city	PLACE is a city
municipality	PLACE is a municipality +V1.1
quarter	PLACE is a quarter
suburb	PLACE is a suburb
town	PLACE is a town
urbanCentre	PLACE is a urban centre
district	PLACE is a district
parish	PLACE is a parish
village	PLACE is a village
hamlet	PLACE is a hamlet
placeOfInterest	PLACE is a place Of Interest
other	other
unrecorded	Not yet specified

7.7.7.3.3 TopographicPlaceDescriptor – Model Element

The ***TopographicPlaceDescriptor*** is used both to name the TOPOGRAPHIC PLACE systematically and to allow the specification of alternative descriptors, either as aliases, or say in other languages. It includes qualifier elements that can be used according to context if the place name needs to be distinguished from another place of the same name, for example '*Newport (Isle of Wight)'*, versus '*NewPort (Gwent)*'.

Table 203 — *TopographicPlaceDescriptor – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	TOPOGRAPHIC PLACE DESCRIPTOR inherits from VERSIONED CHILD.
«PK»	***id***	*TopographicalPlace-DescriptorIdType*	1:1	Identifier of a TOPOGRAPHIC PLACE DESCRIPTOR.
	Name	*MultilingualString*	1:1	Name of a TOPOGRAPHIC PLACE DESCRIPTOR.
	ShortName	*MultilingualString*	0:1	Short Name of a TOPOGRAPHIC PLACE DESCRIPTOR.
«cntd»	***Qualify***	SEQUENCE	0:1	
	QualifierName	*MultilingualString*	1:1	Name used to distinguish TOPOGRAPHIC PLACE from other similar named TOPOGRAPHIC PLACEs. This should not be included in the Name but may be added by applications in context.
«FK»	***TopographicPlace-Ref***	*TopographicPlaceRef*	0:1	Reference to a TOPOGRAPHIC PLACE that qualifies the name of the TOPOGRAPHIC PLACE.

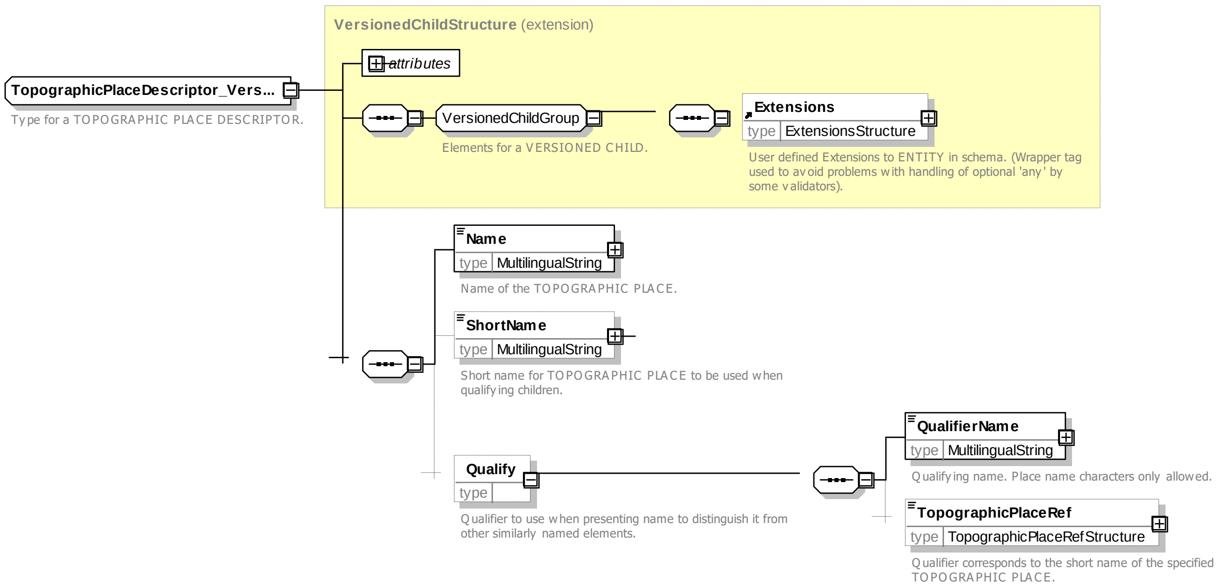


Figure 258 — *TopographicPlaceDescriptor – XSD*

7.7.7.4 Address – Attributes and XSD

7.7.7.4.1 AddressablePlace – Model Element

A PLACE can be specialised in an ADDRESSABLE PLACE whenever it refers to a place having an address, either a ROAD ADDRESS, either a POSTAL ADDRESS.

Table 204 — *AddressablePlace – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Place*	::>	ADDRESSABLE PLACE inherits from PLACE.
«PK»	***id***	*AddressablePlaceIdType*	1:1	Identifier of an ADDRESSABLE PLACE.
	Url	*xsd:anyURI*	0:1	Default URL for ADDRESSABLE PLACE.
	Image	*xsd:anyURI*	0:1	Default image for ADDRESSABLE PLACE.
«cntd»	***PostalAddress***	*PostalAddress*	0:1	A POSTAL ADDRESS to which mail can be sent.
«cntd»	***RoadAddress***	*RoadAddress*	0:1	ADDRESS of a numbered building on a named road.

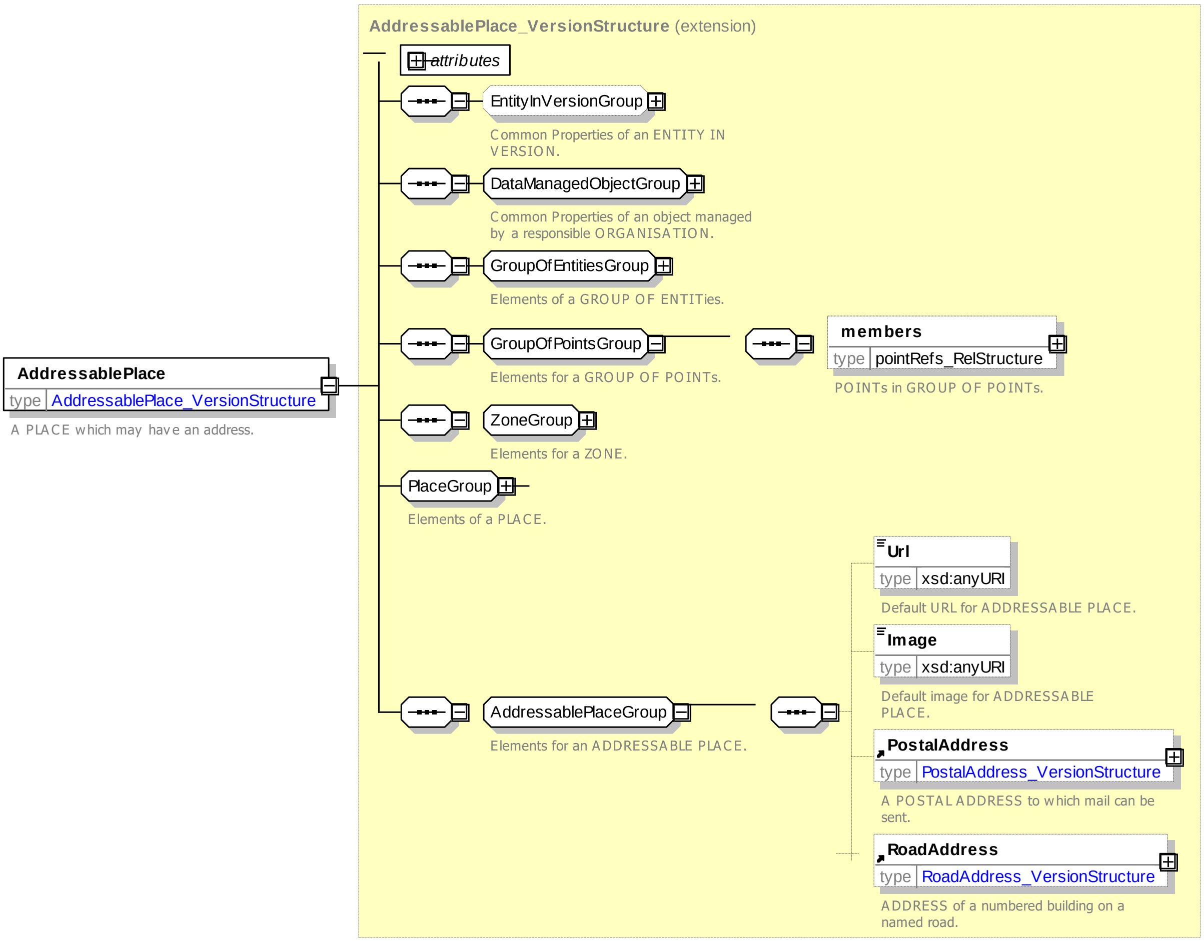


Figure 259 — *AddressablePlace – XSD*

7.7.7.4.2 Address – Model Element

An Address of a PLACE.

Table 205 — *Address – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Place*	::>	ADDRESS inherits from PLACE.
«PK»	***id***	*AddressIdType*	1:1	Identifier of an ADDRESS.
«FK»	***CountryRef***	*CountryRef*	0:1	COUNTRY for ADDRESS. ISO value
	CountryName	*MultilingualString*	0:1	Name of COUNTRY to use on ADDRESS.

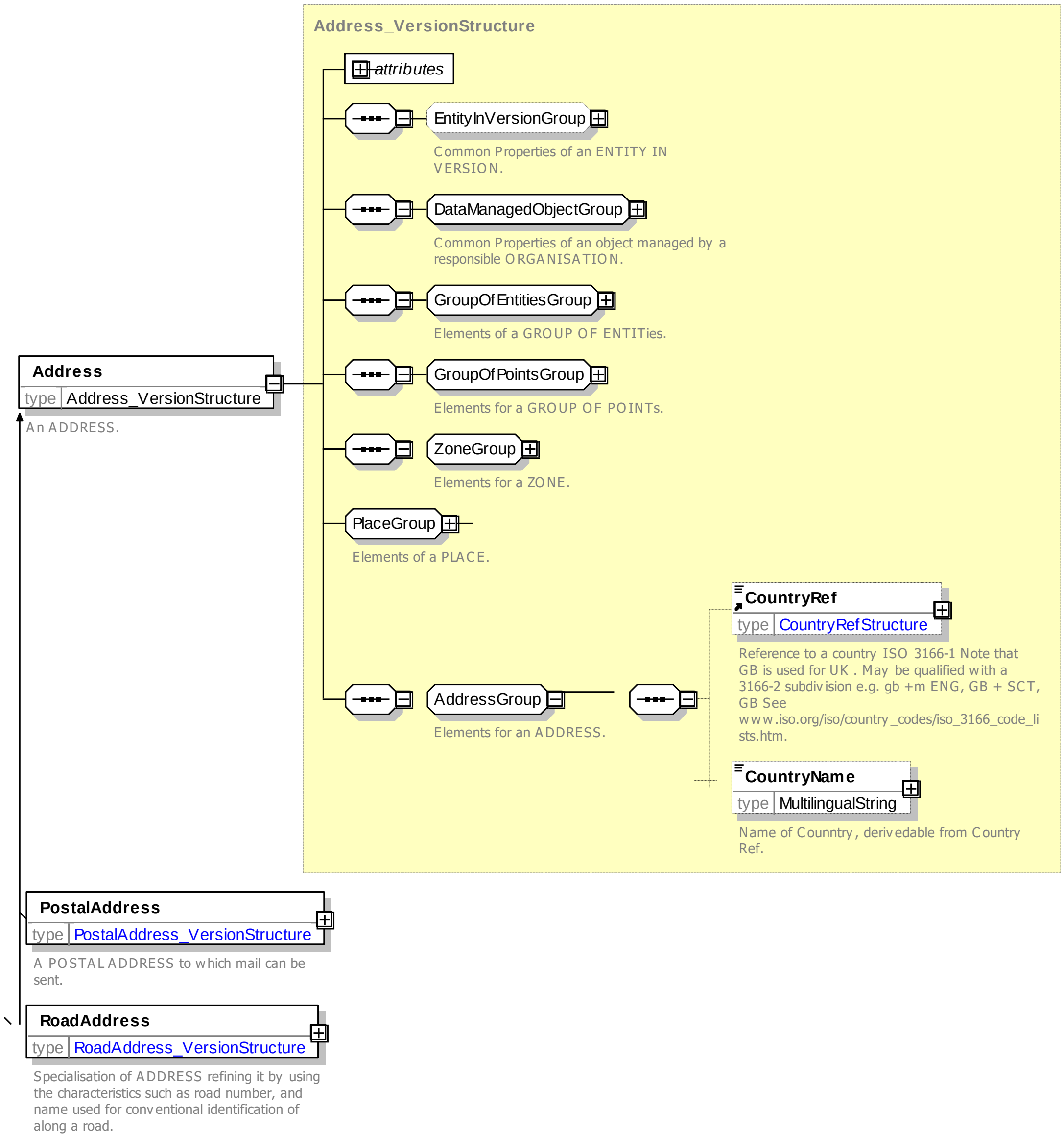


Figure 260 — *Address – XSD*

7.7.7.4.3 PostalAddress – Model Element

A specialisation of ADDRESS refining it by using the attributes used for conventional identification for mail. Comprises variously a building identifier, street name, post code and other descriptors.

Table 206 — *PostalAddress – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Address*	::>	POSTAL ADDRESS inherits from ADDRESS.
«PK»	***id***	*PostalAddressIdType*	1:1	Identifier of POSTAL ADDRESS.
XGRP	***PostalAddressGroup***	**xmlGroup**	0:1	Elements describing properties of POSTAL ADDRESS.
«FK»	***RoadAddressRef***	*RoadAddressRef*	0:1	ROAD ADDRESS associated with POSTAL ADDRESS.

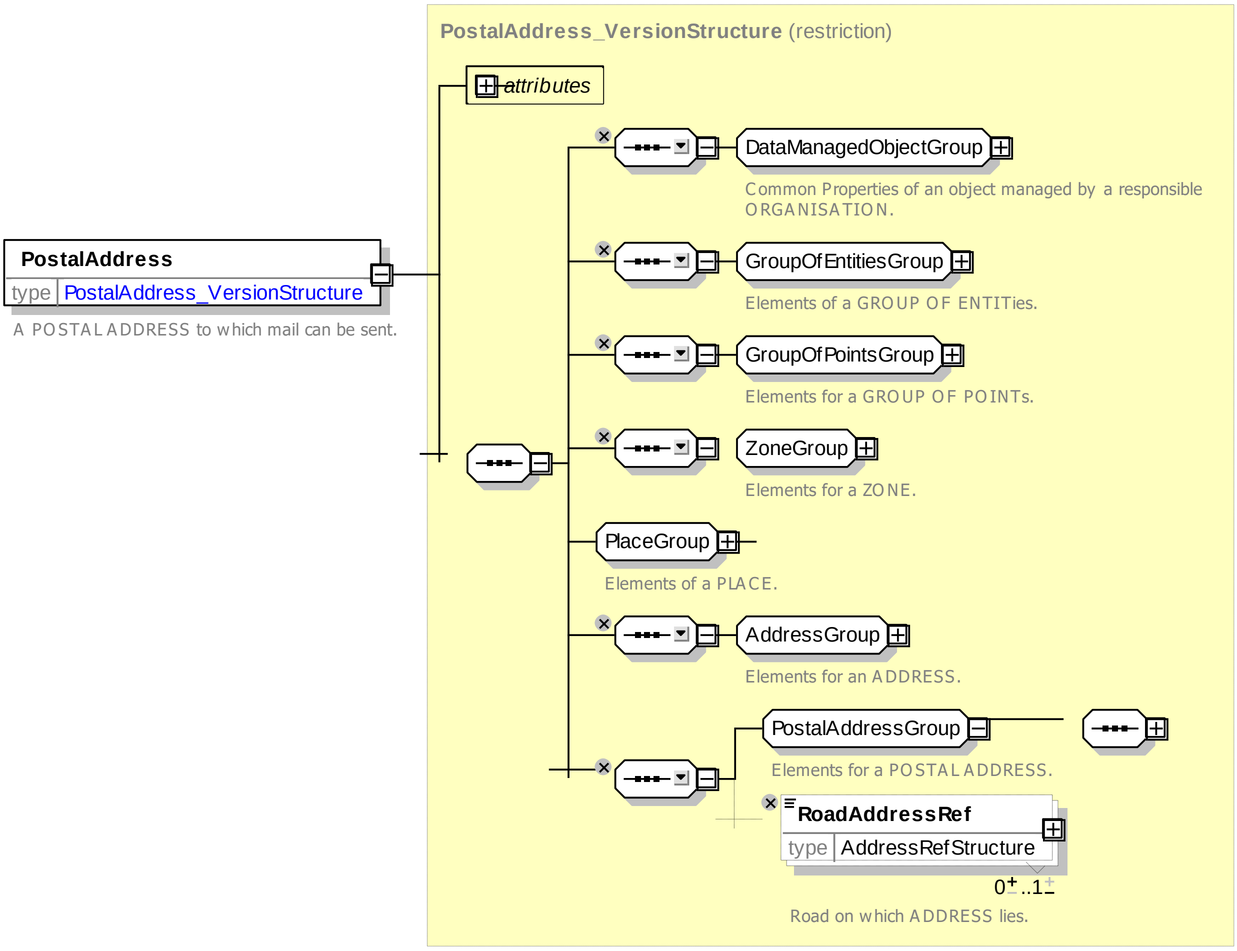


Figure 261 — *PostalAddress – XSD*

7.7.7.4.3.1 PostalAddressGroup – XML Group

Table 207 — *PostalAddressGroup – Group*

Classification	Name	Type	Cardinality	Description
	HouseNumber	*xsd:normalizedString*	0:1	House or building number of POSTAL ADDRESS.
	BuildingName	*xsd:normalizedString*	0:1	Building name of POSTAL ADDRESS.
	AddressLine1	*xsd:normalizedString*	0:1	First line of POSTAL ADDRESS.
	AddressLine2	*xsd:normalizedString*	0:1	Second line of POSTAL ADDRESS.
	Street	*xsd:normalizedString*	0:1	Street name of POSTAL ADDRESS.
	Town	*xsd:normalizedString*	0:1	Town of POSTAL ADDRESS.
	Suburb	*xsd:normalizedString*	0:1	Suburb of POSTAL ADDRESS.

	PostCode	*xsd:normalizedString*	0:1	Post code.
	PostCode-Extension	*xsd:normalizedString*	0:1	Post code extension.
	PostalRegion	*xsd:normalizedString*	0:1	Postal Region.
	Province	*xsd:normalizedString*	0:1	Postal Province.

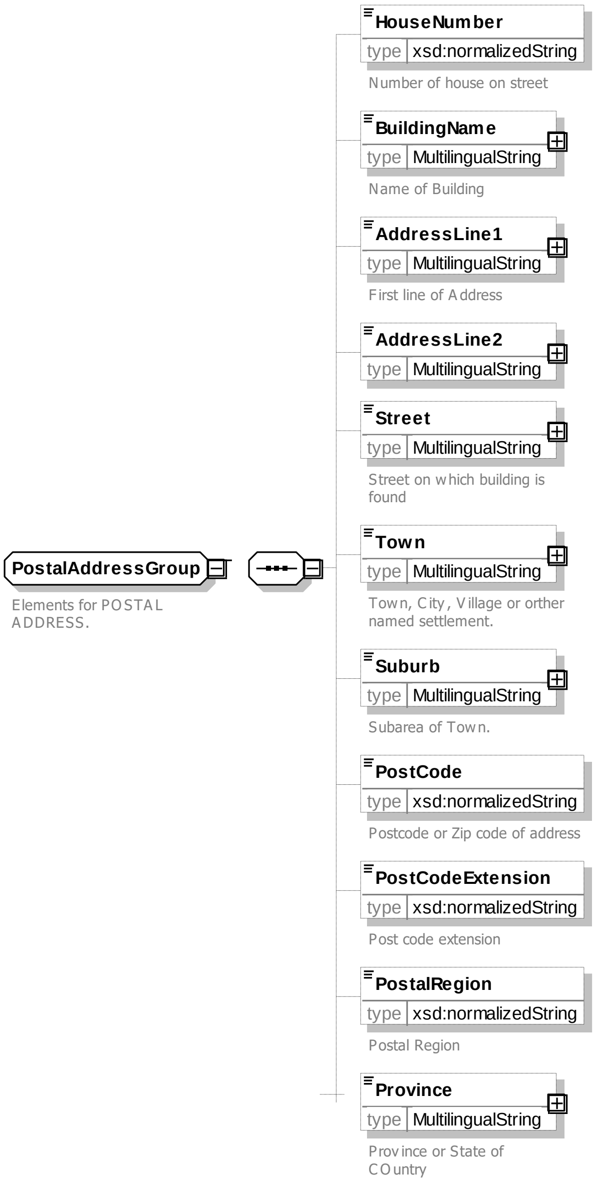


Figure 262 — *PostalAddressGroup – XSD*

7.7.7.4.4 RoadAddress – Model Element

A specialization of ADDRESS refining it by using the attributes such as road number, and name used for conventional identification of along a road.

Table 208 — *RoadAddress – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Address*	::>	ROAD ADDRESS inherits from ADDRESS.

«PK»	***id***	*RoadAddressIdType*	1:1	Identifier of a ROAD ADDRESS.
«AK»	***GisFeatureRef***	*xsd:normalizedString*	0:1	Reference to a GIS feature identifier.
	RoadNumber	*xsd:normalizedString*	0:1	Number of ROAD.
	RoadName	*xsd:normalizedString*	0:1	Name of ROAD.
«enum»	***BearingCompass***	*CompassEnum*	0:1	Compass Bearing of ROAD at point of ADDRESS. See Location utility types earlier for allowed values.
	BearingDegrees	*xsd:integer*	0:1	BEARING in degrees at point of ADDRESS.
«cntd»	***OddNumberRange***	*RoadNumberRange-Structure*	0:1	Odd number range of ADDRESS.
cntd	***EvenNumberRange***	*RoadNumberRange-Structure*	0:1	Even number range of ADDRESS on road.

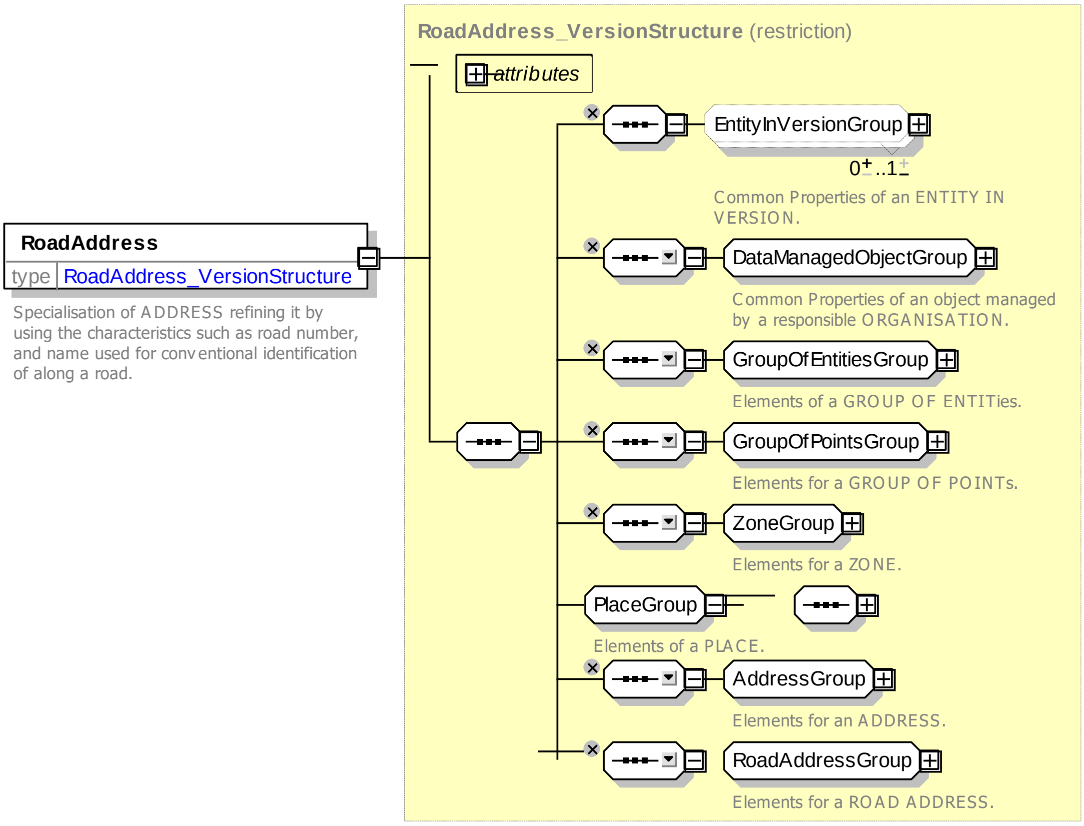


Figure 263 — *RoadAddress – XSD*

7.7.7.4.4.1 RoadAddressGroup - Group

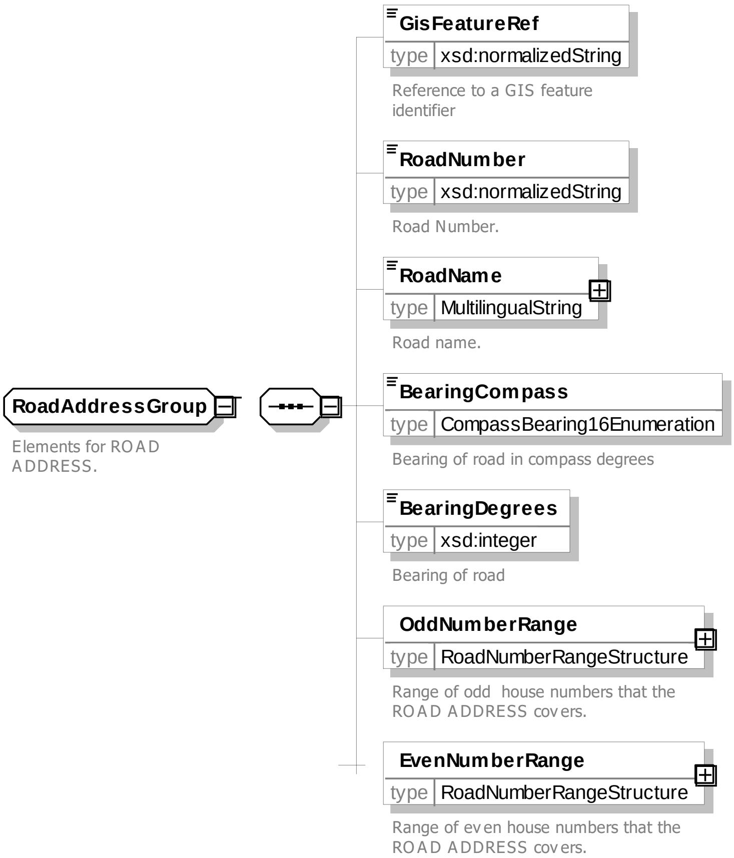


Figure 264 — *RoadAddressGroup – XSD*

7.7.7.4.4.2 RoadNumberRange - Structure

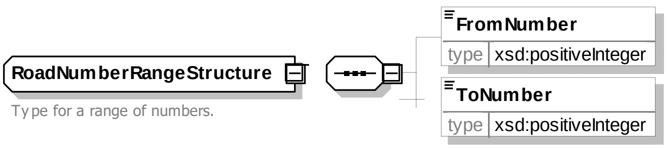


Figure 265 — *RoadNumberRange – XSD*

7.7.8 Transport Organisation

The TRANSPORT ORGANISATION model defines organisations who run Public Transport, specifically OPERATORs of Transport and the AUTHORITY. OPERATORs may be divided into OPERATING DEPARTMENTs.

The generic term OPERATOR expresses a rather general responsibility for a CONCESSIONARY CONTRACT for public transport, where the operational responsibility for the execution of this contract may be handed to a specific OPERATING DEPARTMENT of the ORGANISATIONAL UNIT. The OPERATOR acts as an alias for the ORGANISATIONAL UNIT. Part of the contract-execution can be subcontracted to another OPERATOR. Or even the public transport for a whole area can be divided into several contracts, where a GROUP OF OPERATORs are actually the executers of the public transport timetables for a whole area.

7.7.8.1 TRANSPORT ORGANISATION – Conceptual MODEL

An ORGANISATION PART of an ORGANISATION acts as an ORGANISATIONAL UNIT responsible for the determination of the PT Services, that need to be delivered in an OPERATIONAL CONTEXT often defined or limited to one TRANSPORT MODE or even to one VEHICLE MODE or SUBMODE of one of it's DEPARTMENTs. This defines the actual involved OPERATING DEPARTMENT that will acts as the serving OPERATOR for the ordered services by the public transport AUTHORITY. The serving OPERATORs can be combined for executing this service in a GROUP OF OPERATORs.

It is indeed possible to create a GROUP OF OPERATORs for a specific PURPOSE OF GROUPING, required for special functions or processes in public transport, e.g. CONTROL CENTREs, FARE COLLECTION, PASSENGER INFORMATION, etc.

A CONTROL CENTRE is an organisational concept for where operational management takes place.

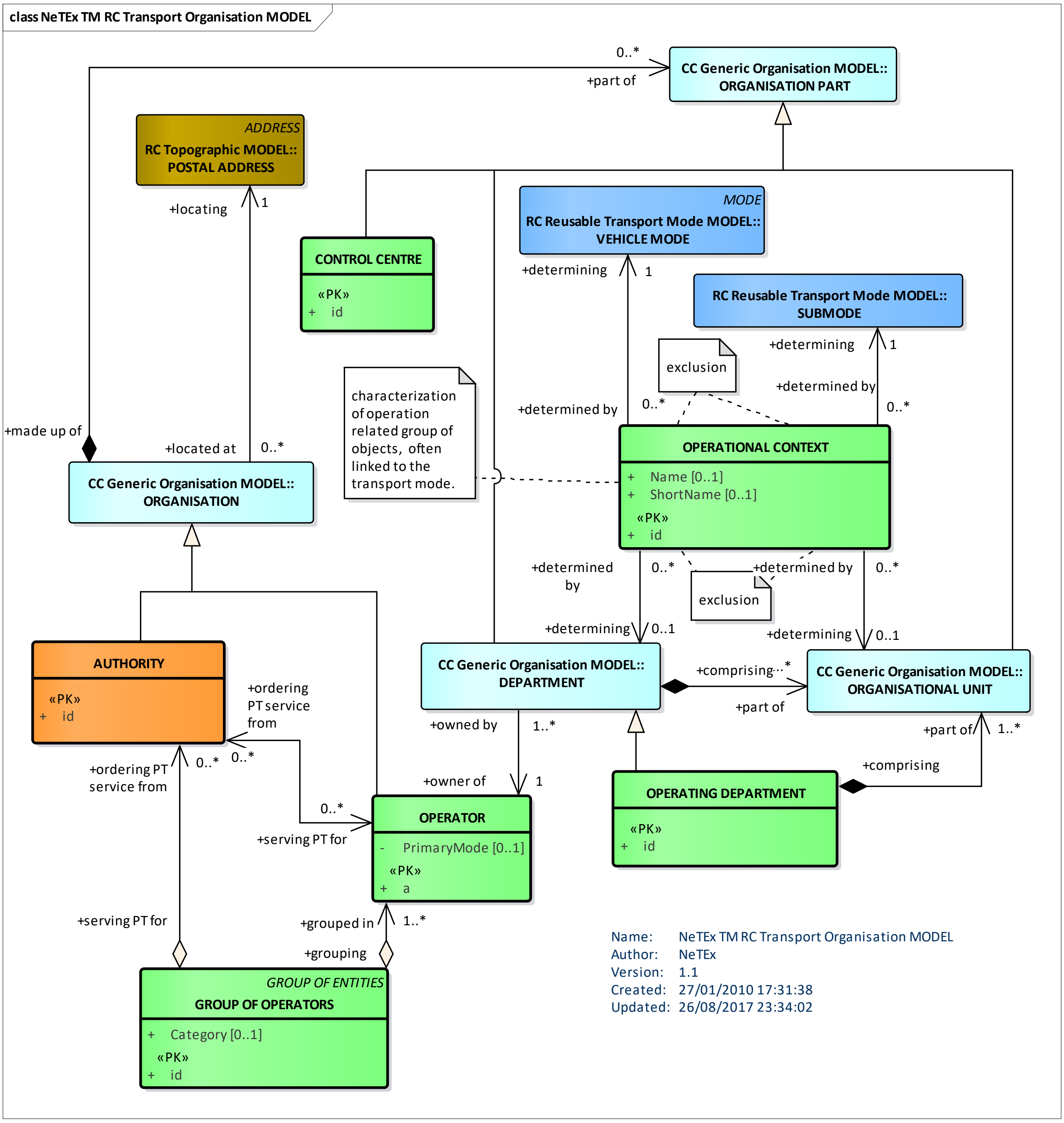
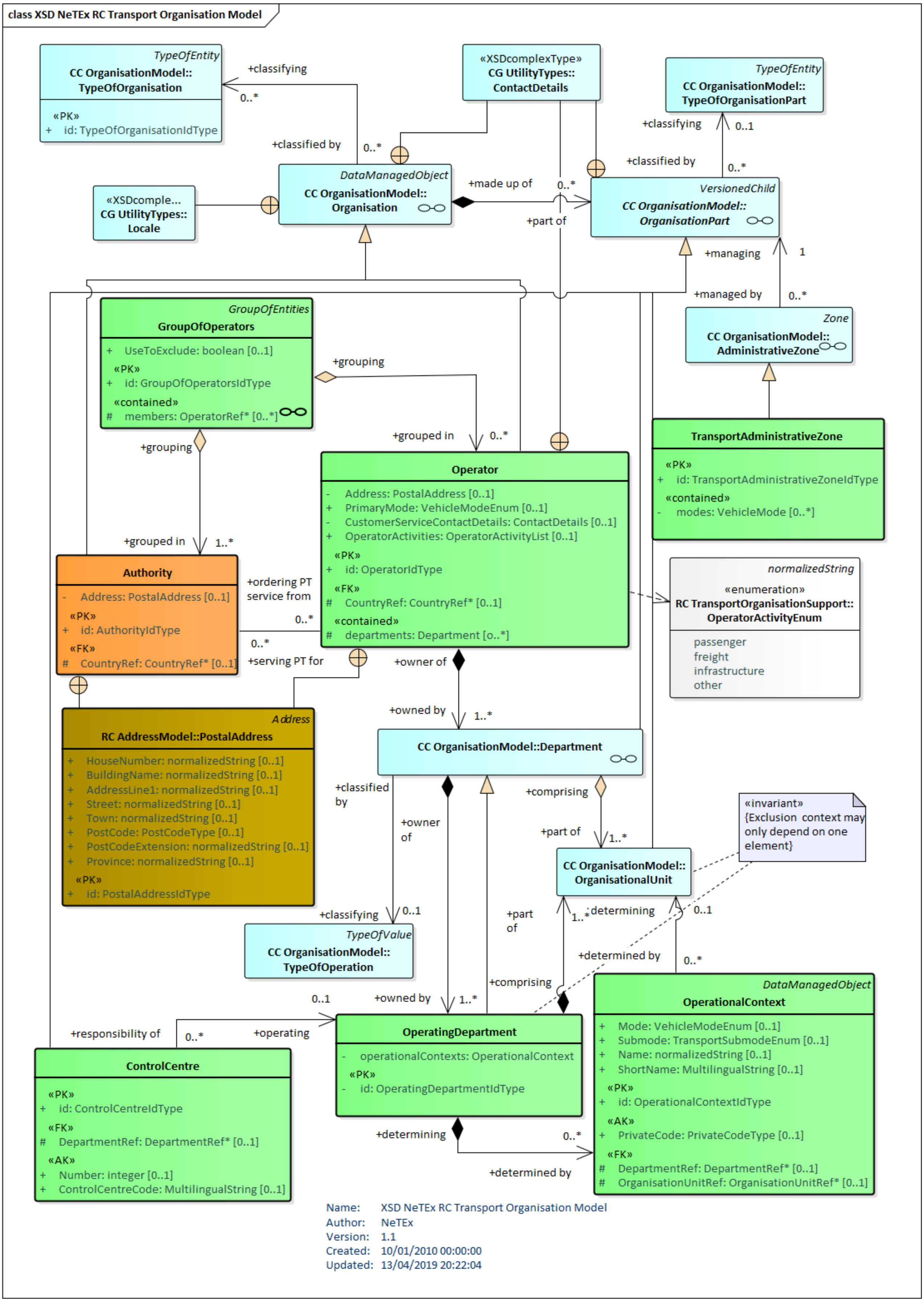


Figure 266 — Transport Organisations – Conceptual MODEL (UML)

7.7.8.2 Transport Organisations – Physical Model

The following figure shows the Physical Model for TRANSPORT ORGANISATIONs.



class XSD NeTEx RC Transport Organisation Model
TypeOfEntity
CC OrganisationModel::TypeOfOrganisation
«PK»
+ id: TypeOfOrganisationIdType
+classifying
0..*
+classified by
0..*
«XSDcomplexType»
CG UtilityTypes::ContactDetails
TypeOfEntity
CC OrganisationModel::TypeOfOrganisationPart
+classifying
0..1
+classified by
0..*
«XSDcomple...
CG UtilityTypes::Locale
DataManagedObject
CC OrganisationModel::Organisation
+made up of
0..*
+part of
VersionedChild
CC OrganisationModel::OrganisationPart
+managing
1
+managed by
0..*
Zone
CC OrganisationModel::AdministrativeZone
GroupOfEntities
GroupOfOperators
+ UseToExclude: boolean [0..1]
«PK»
+ id: GroupOfOperatorsIdType
«contained»
members: OperatorRef* [0..*]
+grouping
+grouped in
0..*
+grouping
+grouped in
1..*
TransportAdministrativeZone
«PK»
+ id: TransportAdministrativeZoneIdType
«contained»
- modes: VehicleMode [0..*]
Operator
- Address: PostalAddress [0..1]
+ PrimaryMode: VehicleModeEnum [0..1]
- CustomerServiceContactDetails: ContactDetails [0..1]
+ OperatorActivities: OperatorActivityList [0..1]
«PK»
+ id: OperatorIdType
«FK»
CountryRef: CountryRef* [0..1]
«contained»
departments: Department [o..*]
normalizedString
«enumeration»
RC TransportOrganisationSupport::OperatorActivityEnum
passenger
freight
infrastructure
other
Authority
- Address: PostalAddress [0..1]
«PK»
+ id: AuthorityIdType
«FK»
CountryRef: CountryRef* [0..1]
+ordering PT service from
0..*
0..*
+serving PT for
+owner of
+owned by
1..*
Address
RC AddressModel::PostalAddress
+ HouseNumber: normalizedString [0..1]
+ BuildingName: normalizedString [0..1]
+ AddressLine1: normalizedString [0..1]
+ Street: normalizedString [0..1]
+ Town: normalizedString [0..1]
+ PostCode: PostCodeType [0..1]
+ PostCodeExtension: normalizedString [0..1]
+ Province: normalizedString [0..1]
«PK»
+ id: PostalAddressIdType
CC OrganisationModel::Department
+classified by
+owner of
+comprising
+part of
1..*
«invariant»
{Exclusion context may only depend on one element}
CC OrganisationModel::OrganisationalUnit
+classifying
0..1
TypeOfValue
CC OrganisationModel::TypeOfOperation
+part of
1..*
+determining
0..1
+determined by
0..*
+comprising
DataManagedObject
OperationalContext
+ Mode: VehicleModeEnum [0..1]
+ Submode: TransportSubmodeEnum [0..1]
+ Name: normalizedString [0..1]
+ ShortName: MultilingualString [0..1]
«PK»
+ id: OperationalContextIdType
«AK»
+ PrivateCode: PrivateCodeType [0..1]
«FK»
DepartmentRef: DepartmentRef* [0..1]
OrganisationUnitRef: OrganisationUnitRef* [0..1]
0..1
+owned by
1..*
+responsibility of
0..*
+operating
OperatingDepartment
- operationalContexts: OperationalContext
«PK»
- id: OperatingDepartmentIdType
ControlCentre
«PK»
+ id: ControlCentreIdType
«FK»
DepartmentRef: DepartmentRef* [0..1]
«AK»
+ Number: integer [0..1]
+ ControlCentreCode: MultilingualString [0..1]
+determining
0..*
+determined by
Name: XSD NeTEx RC Transport Organisation Model
Author: NeTEx
Version: 1.1
Created: 10/01/2010 00:00:00
Updated: 13/04/2019 20:22:04

Figure 267 — Transport Organisations – Physical Model (UML)

7.7.8.3 Transport Organisations – Attributes and XSD

7.7.8.3.1 Authority – Model Element

The ORGANISATION under which the responsibility of organising the transport service in a certain area is placed.

Table 209 — *Authority – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Organisation*	::>	AUTHORITY inherits from ORGANISATION.
«PK»	***id***	*AuthorityIdType*	1:1	Identifier of AUTHORITY.
«FK»	***CountryRef***	*CountryRef*	0:1	Reference to a country ISO 3166-1. +v1.1
«cntd»	***Address***	*PostalAddress*	1:1	Postal ADDRESS of ORGANISATION.
«cntd»	***authorityTypes***	*TypeOfOrganisationRef*	0:*	Classification of AUTHORITY.

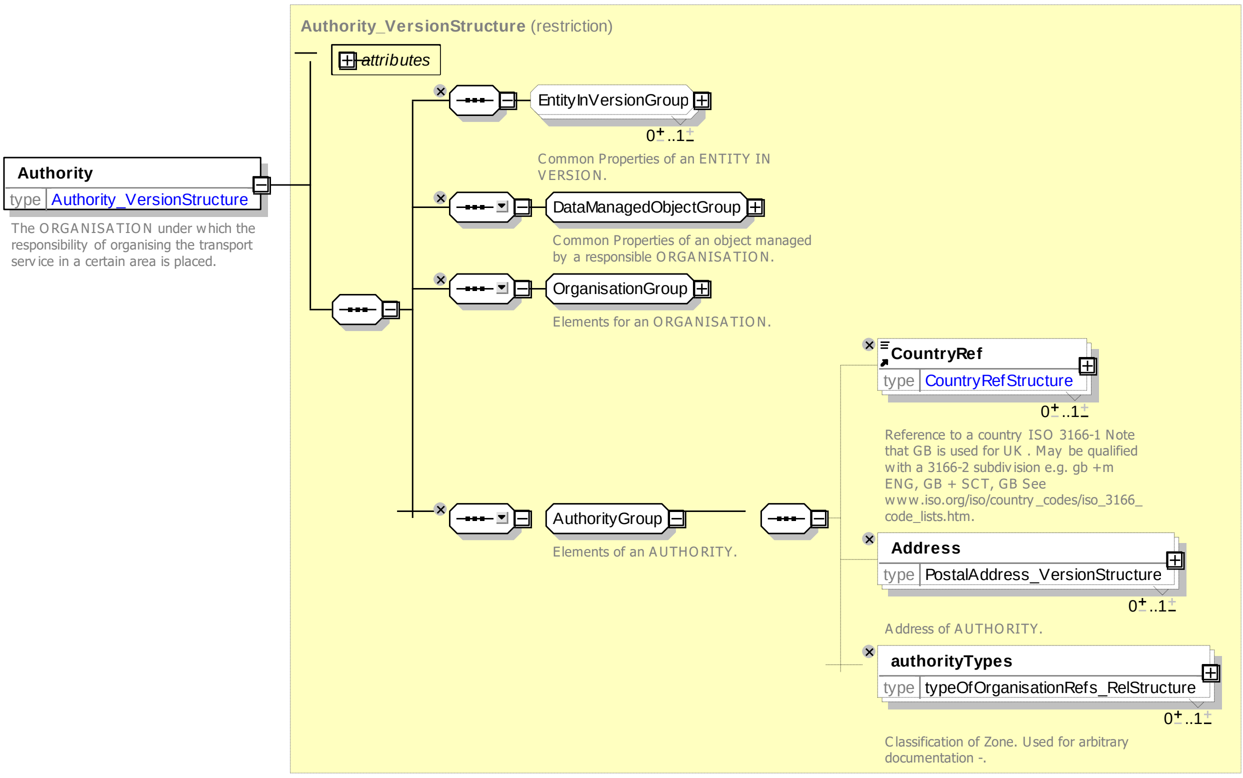


Figure 268 — *Authority – XSD*

7.7.8.3.2 Operator – Model Element

A company providing public transport services.

Table 210 — *Operator – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Organisation*</u>	::>	OPERATOR inherits from ORGANISATION.
«PK»	***id***	*OperatorIdType*	1:1	Identifier of OPERATOR.
«FK»	***CountryRef***	*CountryRef*	0:1	Reference to a country ISO 3166-1
«cntd»	***Address***	<u>*PostalAddress*</u>	0:1	Postal ADDRESS of ORGANISATION.
«enum»	***PrimaryMode***	*VehicleModeEnum*	0:1	Primary TRANSPORT MODE of ORGANISATION. See TRANSPORT MODE model above for allowed values.
«enum»	***Operator-Activities***	*OperatorActivitiesEnum*	0:*	Activities undertaken by OPERATOR. See allowed values below.
«cntd»	***CustomerService ContactDetails***	<u>*ContactDetails*</u>	0:1	Contact details for ORGANISATION's customer services.
«cntd»	***departments***	<u>*Department | DepartmentRef*</u>	0:*	DEPARTMENTs for ORGANISATION.

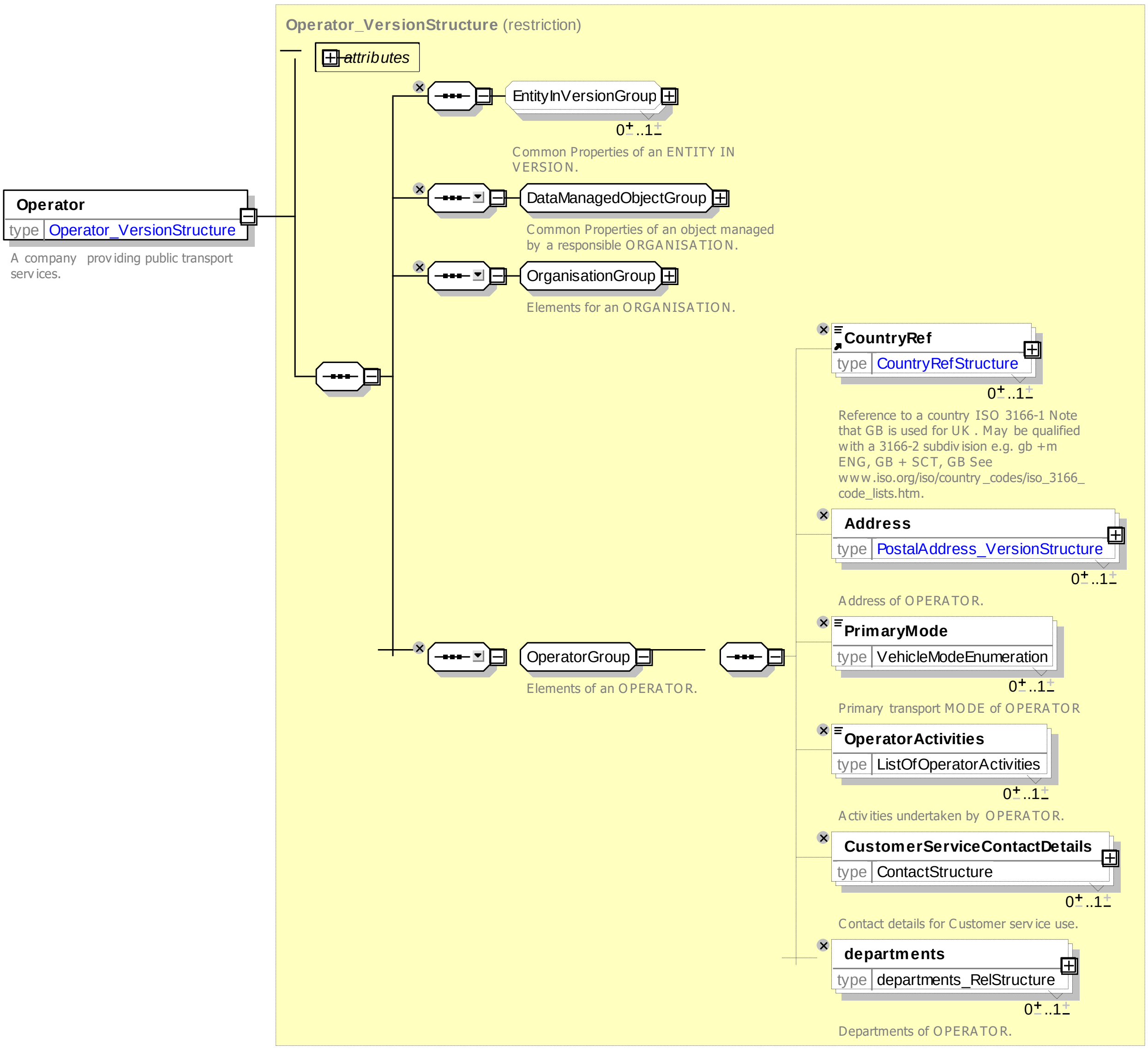


Figure 269 — *Operator – XSD*

7.7.8.3.2.1 OperatorActivities – Allowed Values

Allowed values for ***OperatorActivity*** (*OperatorActivityEnum*).

Table 211 — *OperatorActivity – AllowedValues*

Value	Description
passengers	Operator carries passengers.
freight	Operator carries freight.
infrastructure	Operator maintains infrastructure.
other	Other activites.

7.7.8.3.3 GroupOfOperators – Model Element

A group of OPERATORs having for instance common schemes for fare collection or passenger information.

Table 212 — *GroupOfOperators – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF OPERATORs inherits from GROUP OF ENTITies.
«PK»	***id***	*GroupOfOperatorsIdType*	1:1	Identifier of GROUP OF OPERATORs.
	UseToExclude	*xsd:boolean*	0:1	Whether contents of group represent excluded OPERATORs. +v1.1
«cntd»	***members***	*(TransportOrganisationRef) OperatorRef | AuthorityRef*	0:*	OPERATORs or AUTHORITies in GROUP OF OPERATORs.

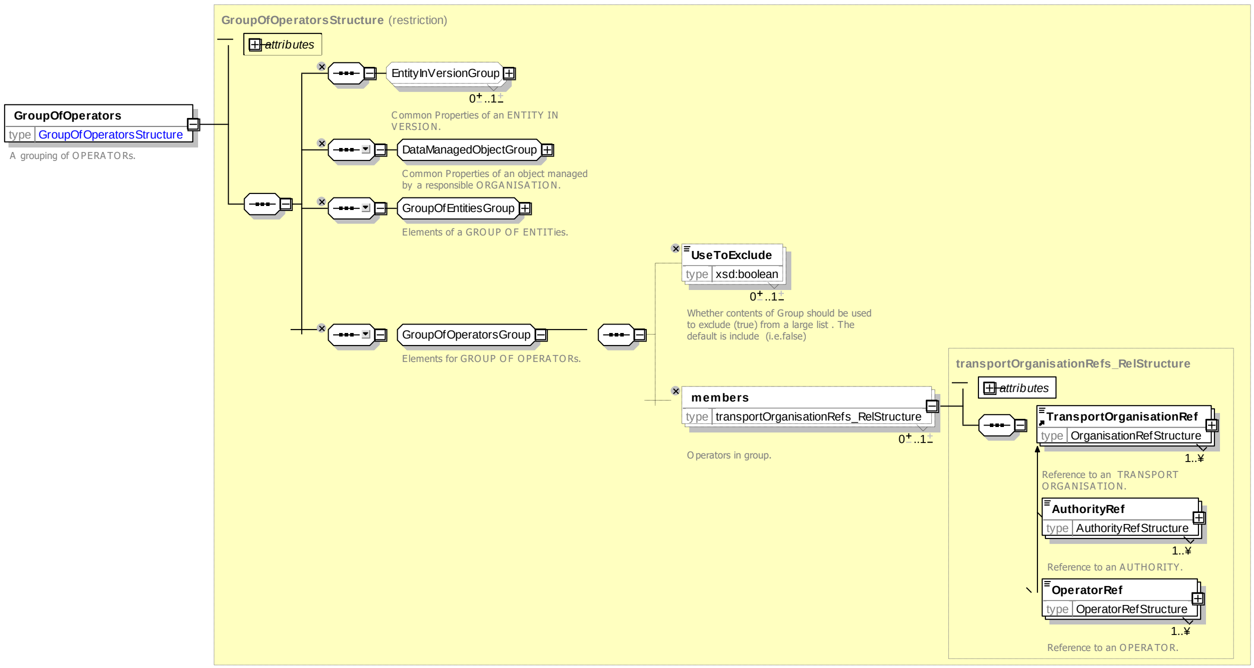


Figure 270 — *GroupOfOperators – XSD*

7.7.8.3.4 OperationalContext – Model Element

Characterization of a set of operational objects, such as timing or links determined either by a DEPARTMENT or by an ORGANISATIONAL UNIT.

Table 213 — *OperationalContext – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	OPERATIONAL CONTEXT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*OperationalContextIdType*	1:1	Identifier of OPERATIONAL CONTEXT.
	Name	*xsd:normalizedString*	0:1	Name of OPERATIONAL CONTEXT.
	ShortName	*MultilingualString*	0:1	Short name of OPERATIONAL CONTEXT.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative identifier of OPERATIONAL CONTEXT.

«FK»	***OrganisationPart Ref***	*OrganisationPartRef*	0:1	Reference to an ORGANISATIONAL Part associated with OPERATIONAL CONTEXT.
«enum»	***VehicleMode***	*VehicleModeEnum*	0:1	VEHICLE MODE associated with OPERATIONAL CONTEXT. See TTRANSPORT MODE model for allowed values.
«enum»	***Transport-Submode***	*TransportSubmodeEnum*	0:1	SUBMODE associated with MODE.

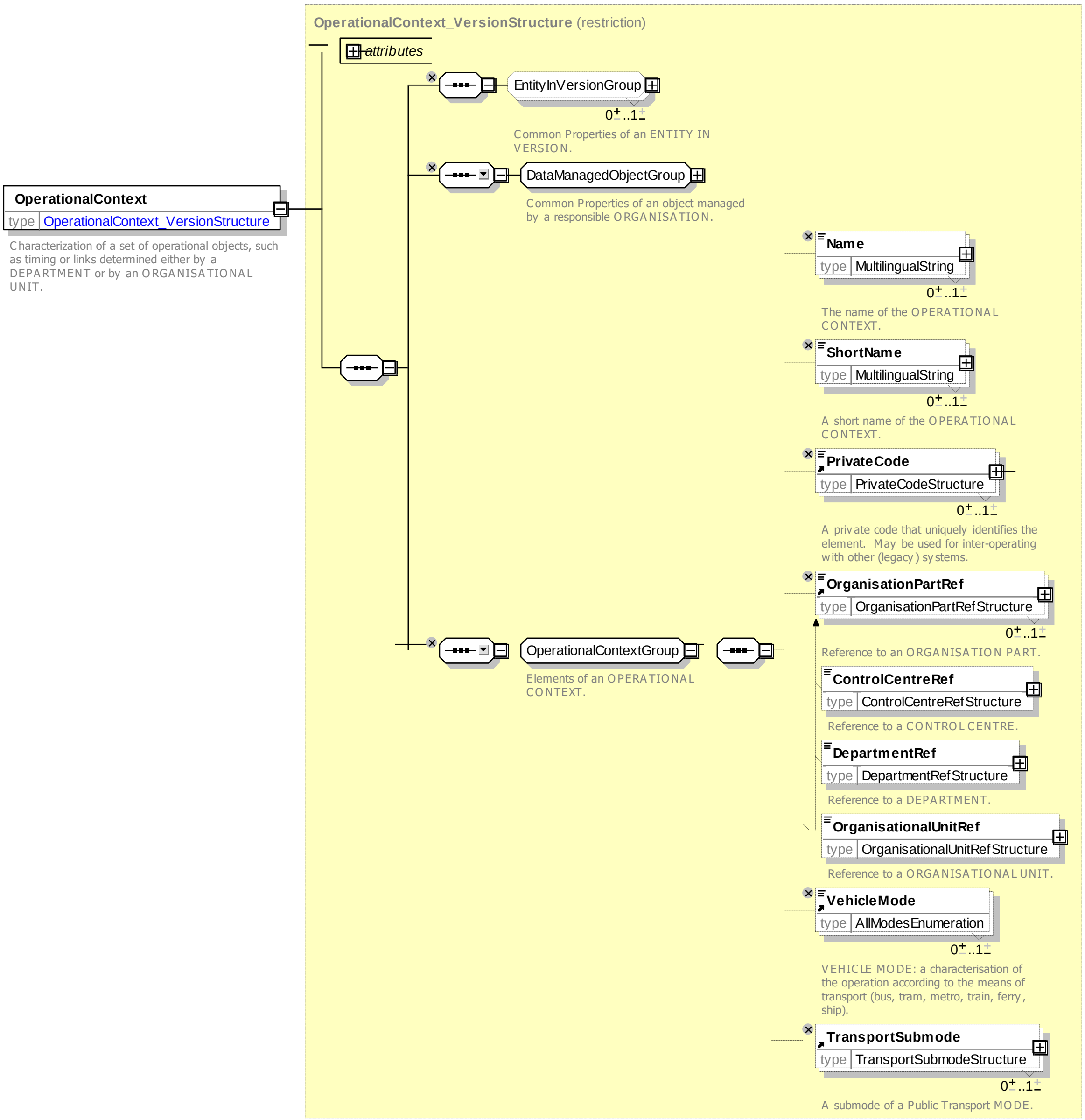


Figure 271 — *OperationalContext – XSD*

7.7.8.3.5 OperatingDepartment – Model Element

The operating department which administers certain LINEs.

Table 214 — *OperatingDepartment – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	*Department*	::>	OPERATING DEPARTMENT inherits from DEPARTMENT.
«PK»	***id***	*DepartmentIdType*	1:1	Identifier of OPERATING DEPARTMENT.
«cntd»	***Operational-Contexts***	*OperationalContextRef*	0:*	List of references to OPERATIONAL CONTEXTs.

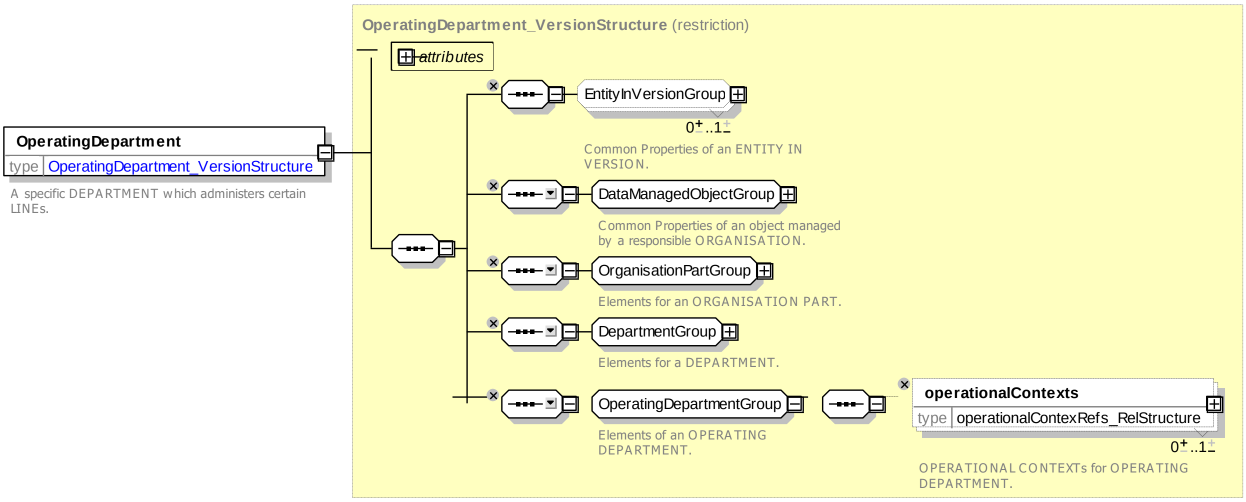


Figure 272 — *OperatingDepartment – XSD*

7.7.8.3.6 ControlCentre – Model Element

An ORGANISATION PART for an operational team who are responsible for issuing commands to control the services.

Table 215 — *ControlCentre – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	CONTROL CENTRE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*ControlCentreIdType*	1:1	Identifier of CONTROL CENTRE.
	Name	*MultilingualString*	0:1	Name of CONTROL CENTRE.
«AK»	***Number***	*xsd:integer*	0:1	Unique number to use to identify CONTROL CENTRE.
«AK»	***ControlCentreCode***	*MultilingualString*	0:1	Unique alphanumeric identification of CONTROL CENTRE used to identify source of request to external systems.
«FK»	***DepartmentRef***	*DepartmentRef*	0:1	Reference to an OPERATING DEPARTMENT that runs CONTROL CENTRE.

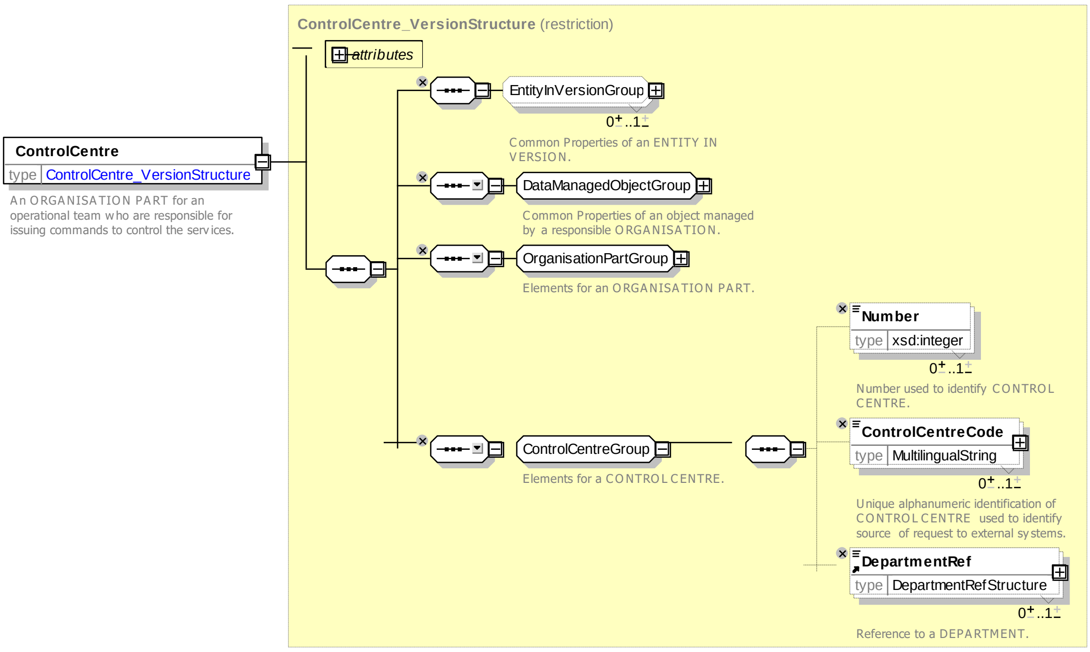


Figure 273 — *ControlCentre – XSD*

7.7.8.3.7 TransportAdministrativeZone – Model Element

The area of a district, a region, a city, a municipality, or other area with which an ORGANISATION has a RESPONSIBILITY ROLE.

Table 216 — *TransportAdministrativeZone – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*AdministrativeZone*	::>	TRANSPORT ADMINISTRATIVE ZONE inherits from ADMINISTRATIVE ZONE.
«PK»	***id***	*TransportAdministrative-ZoneIdType*	1:1	Identifier of a TRANSPORT ADMINISTRATIVE ZONE.
«enum»	***VehicleModes***	i	0:*	Transport responsibilities of the ORGANISATION for the ADMINISTRATIVE ZONE.

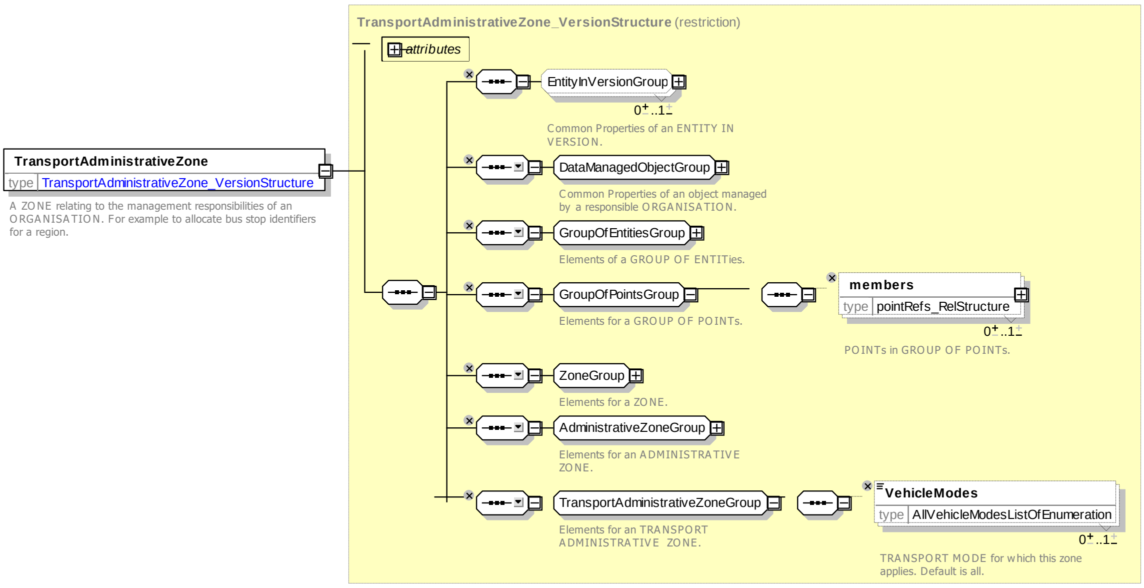


Figure 274 — *TransportAdministrativeZone – XSD*

7.7.8.4 XML Examples of Transport Organisations

7.7.8.4.1 Operator data – XML Fragment Example

The following XML fragment shows definitions for an AUTHORITY and an OPERATOR.

EXAMPLE

```
<organisations>
    <Authority version="any" id="hde:Org_Hd001">
        <Name>Town Authority </Name>
        <Description>Responsbile for networks schedules</Description>
    </Authority>
::>
    <Operator version="2" id="txc:Op_01">
        <PublicCode>ABC</PublicCode>
        <PrivateCode>12345</PrivateCode>
        <Name>ABC Buses</Name>
        <ShortName>ABC Buses</ShortName>
        <LegalName>ABC Buses and Services></LegalName>
        <TradingName>ABC Buses</TradingName>
        <ContactDetails>
            <ContactPerson>Sid Mouth</ContactPerson>
            <Email>enquiries@ABCBuses.com</Email>
            <Phone>0207654321</Phone>
            <Fax>0207654322</Fax>
            <Url>http://abcbuses.co.uk</Url>
        </ContactDetails>
        <Address id="txc:RoadAddress:txcexm:01">
            <HouseNumber>45</HouseNumber>
            <BuildingName>Pent House</BuildingName>
            <AddressLine1>Second Floor</AddressLine1>
            <Street>City Road</Street>
            <Town>London</Town>
            <PostCode>EC1V 3PH</PostCode>
        </Address>
        <PrimaryMode>bus</PrimaryMode>
    </Operator>
</organisations>
```

7.7.9 Generic Equipment

7.7.9.1 Generic EQUIPMENT – Conceptual MODEL

The Generic EQUIPMENT Model represents items of equipment which may be located on a Vehicle, Site etc. There are many different types of EQUIPMENT, each of which may have specific properties. These are classified under two main specialisations:

- INSTALLED EQUIPMENT: fixed EQUIPMENT that may be installed on a SITE, such as a door, lift, gate etc. or a VEHICLE. Its exact location may be specified by an EQUIPMENT PLACE. This is further characterised into:
  - PLACE EQUIPMENT: Equipment which may be located only on a SITE, such as a barrier, bench, Lift.
  - PASSENGER EQUIPMENT: Equipment which may be located on either a vehicle or a SITE, such as a display terminal, ticket validator or WC.
- LOCAL SERVICE : an intangible service that is provided at a site such selling tickets, porterage, etc.

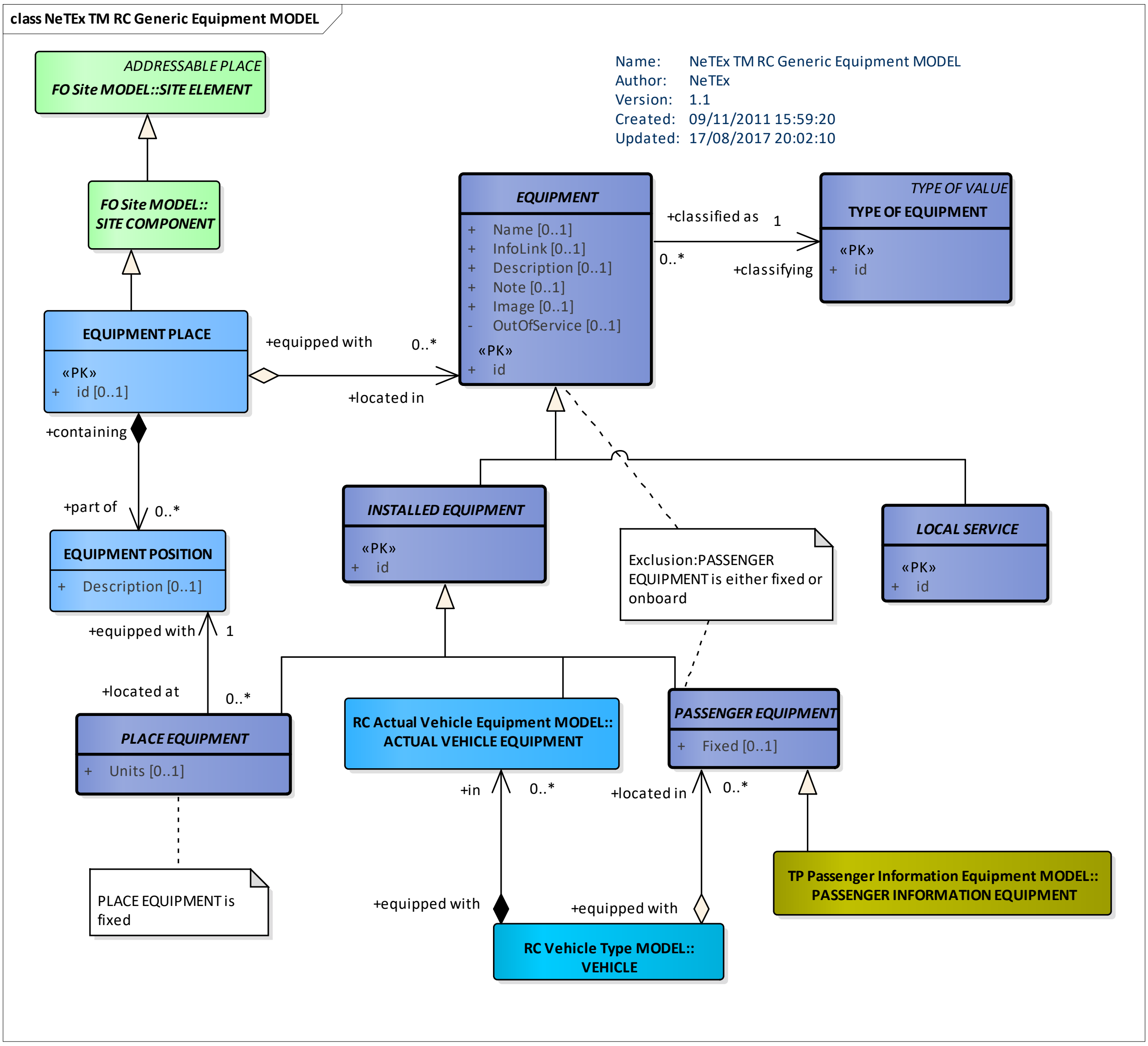


Figure 275 — Generic Equipment – Conceptual MODEL (UML)

7.7.9.1.1 Vehicle Equipment – Conceptual MODEL

ACTUAL VEHICLE EQUIPMENT can be used to specify the EQUIPMENT available on a VEHICLE of a specific VEHICLE TYPE.

The EQUIPMENT may be available for all or part of a SERVICE JOURNEY. For example a restaurant service may only be available for certain sections.

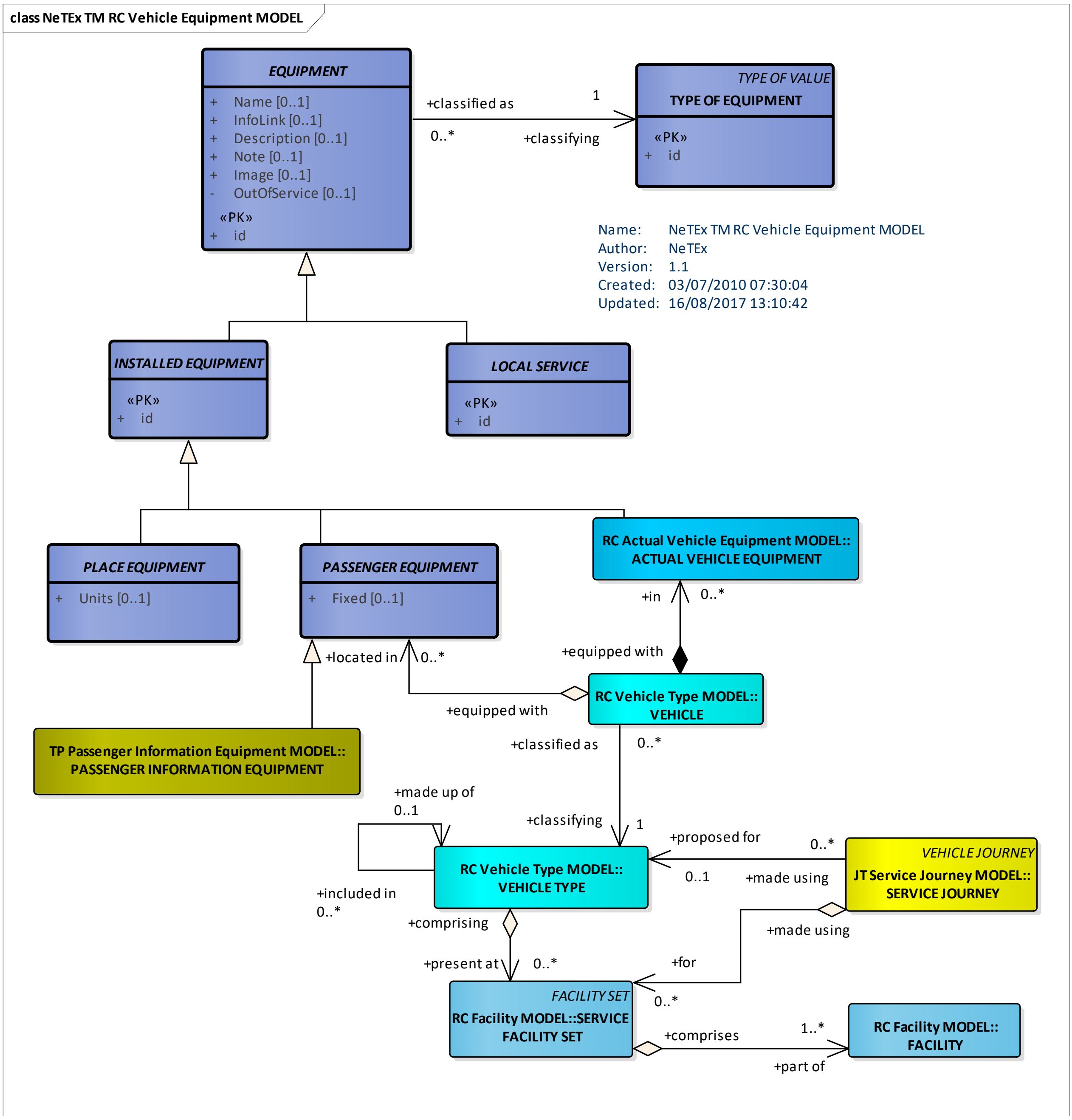


Figure 276 — Vehicle equipment – Conceptual MODEL (UML)

7.7.9.1.2 Site Equipment – Conceptual MODEL

The Site EQUIPMENT Model represents instances of EQUIPMENT located at a site, such as doors, stairs, ticket machines, signs, etc. Each type of Equipment can have specific properties.

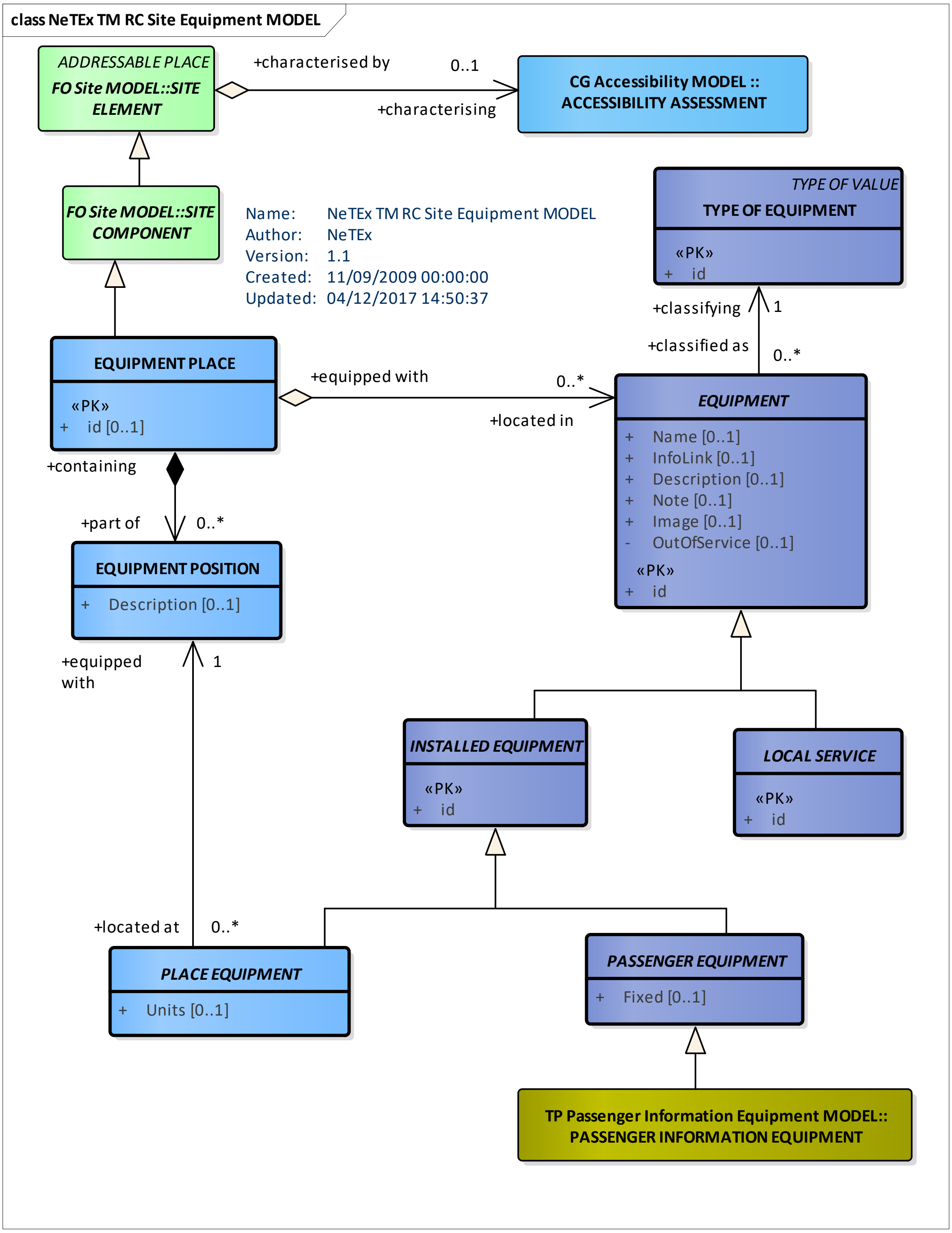


Figure 277 — Site Equipment – Conceptual MODEL (UML)

7.7.9.2 Generic Equipment – Physical Model

The Generic EQUIPMENT Physical model defines classes for the common properties of EQUIPMENT. These are specialised in other models.

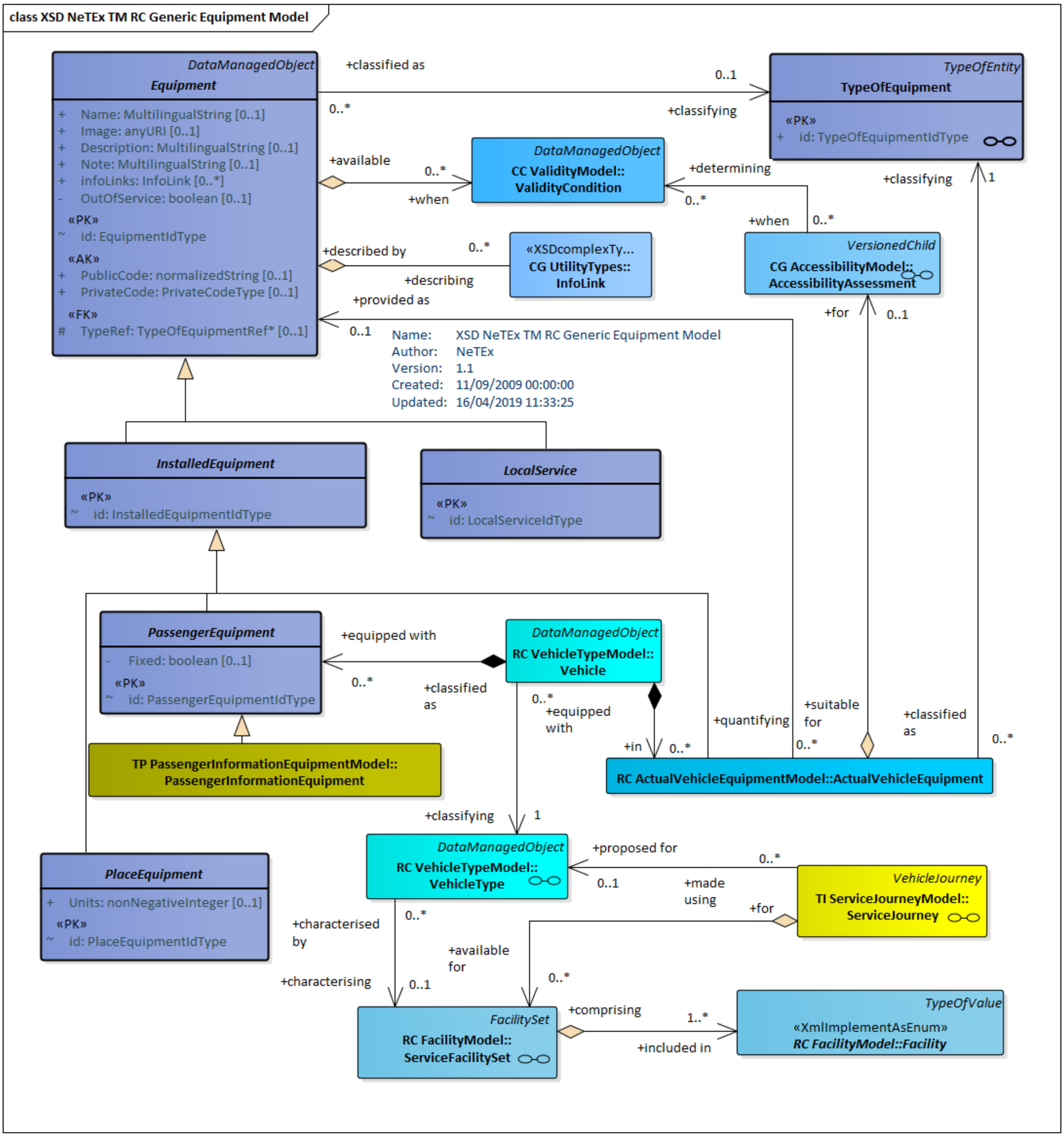


Figure 278 — Generic Equipment – Physical Model (UML)

7.7.9.2.1 Equipment Place – Physical Model

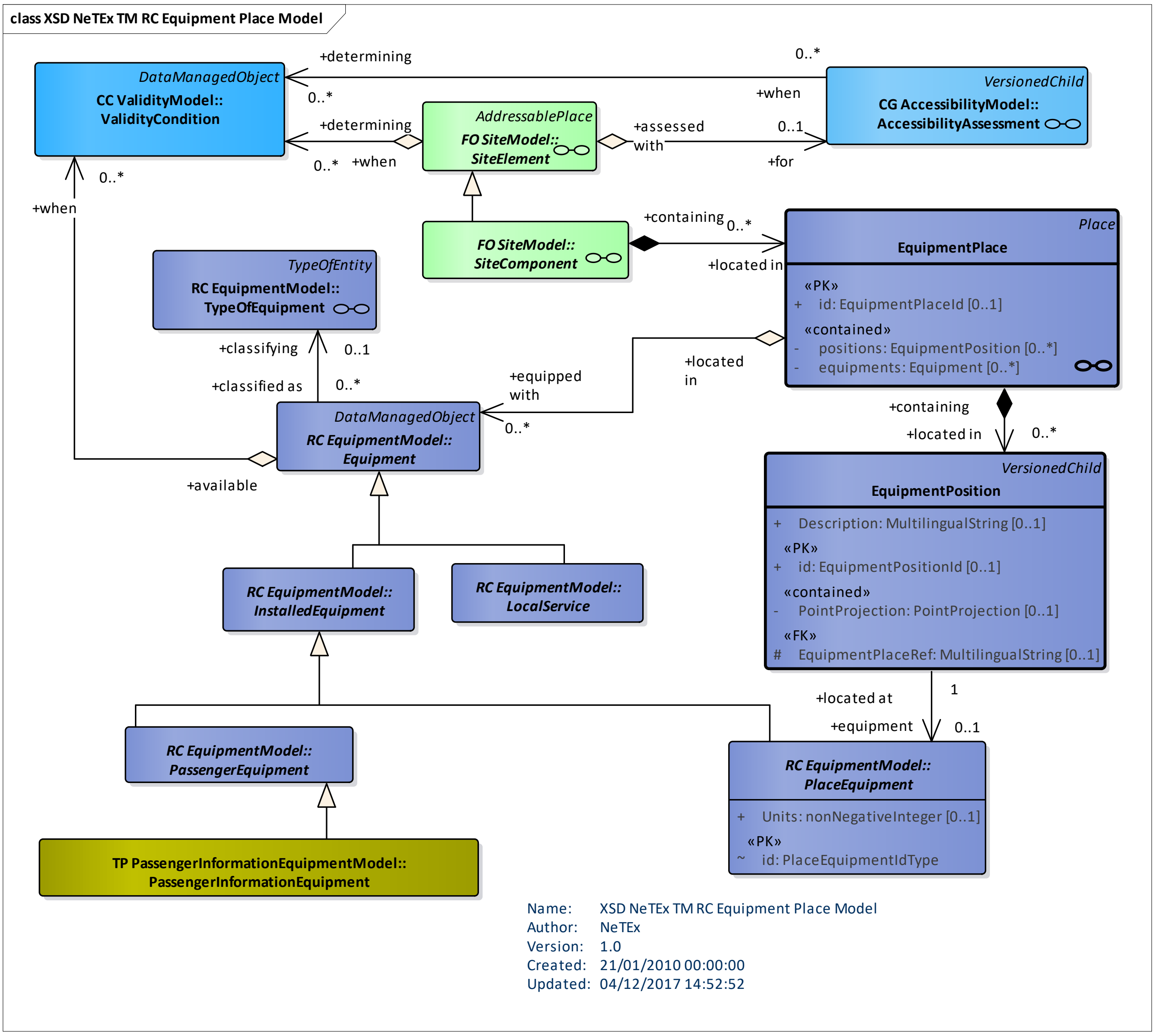


Figure 279 — Equipment Place – Physical Model (UML)

7.7.9.2.2 Summary of Equipment – Physical Model

The following figure summarises most of the specialisations of EQUIPMENT defined in NeTEx.

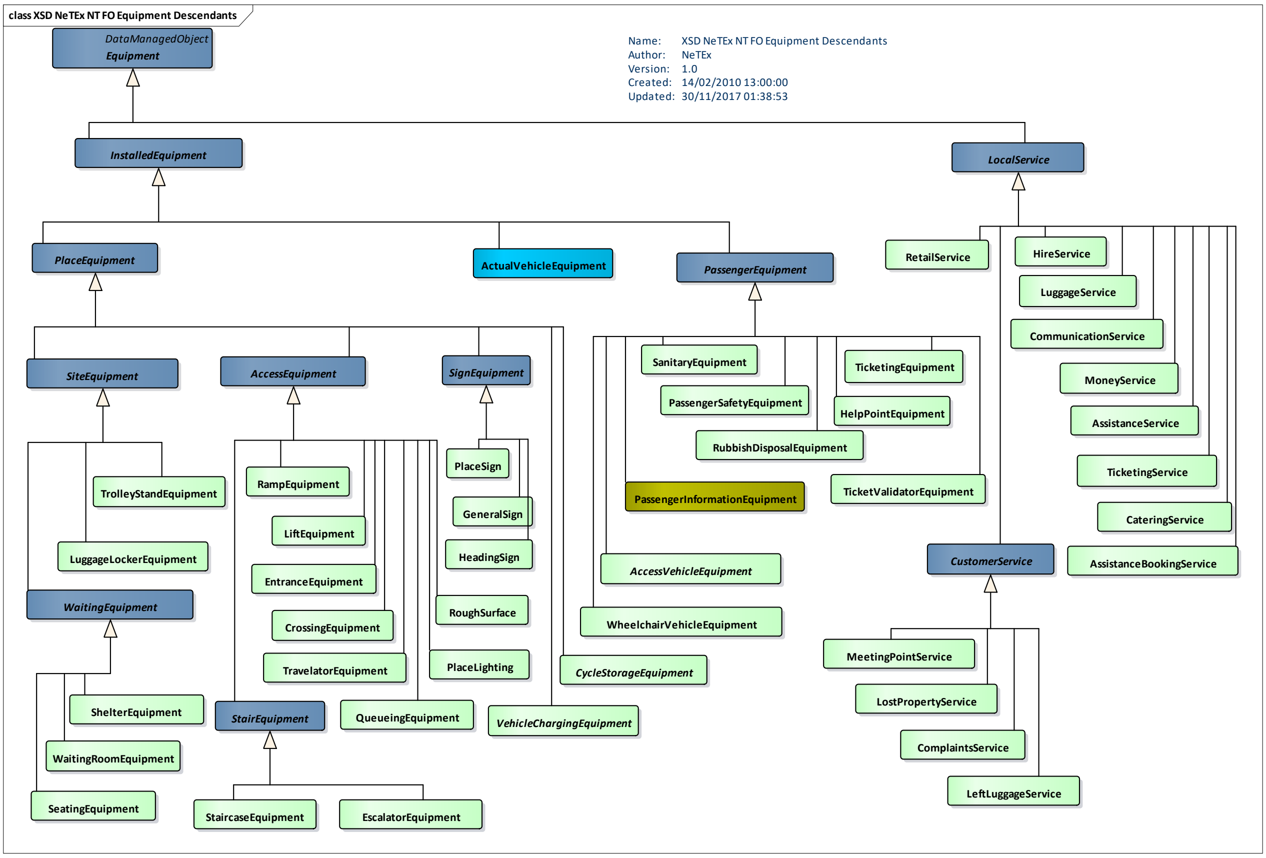


Figure 280 — Equipment Overview – Physical Model (UML)

7.7.9.2.3 Types of Equipment – Physical Model

EQUIPMENT & FACILITies may be classified using a TYPE OF EQUIPMENT. The following figure summarises most of the TYPEs of EQUIPMENT defined in NeTEx.

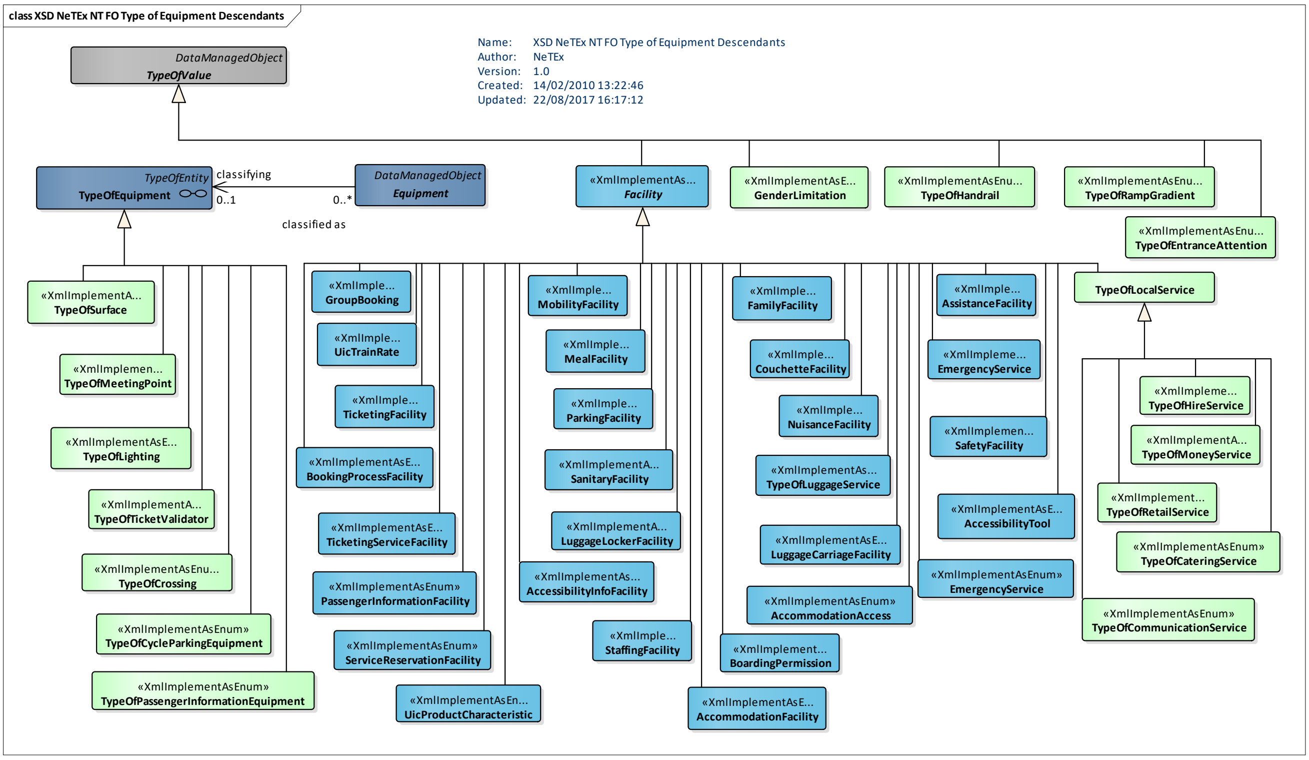


Figure 281 — Types of Equipment – Physical Model (UML)

STOP PLACE EQUIPMENT may comprise:

— INFORMATION EQUIPMENT,
— PASSENGER INFO EQUIPMENT,
— SIGNAGE,
— STOP SIGN,
— OTHER SIGN,
— HEADING SIGN,
— ACCESS EQUIPMENT,
— QUEUING EQUIPMENT,
— STAIR EQUIPMENT,
— LIFT EQUIPMENT,
— RAMP EQUIPMENT,
— CROSSING EQUIPMENT,
— ENTRANCE EQUIPMENT,
— ESCALATOR EQUIPMENT,
— STAIRCASE EQUIPMENT,
— ROUGH SURFACE,

or other service related equipment such as :

— TICKETING EQUIPMENT,
— TROLLEY STAND EQUIPMENT,
— WAITING EQUIPMENT,
— PASSENGER SAFETY EQUIPMENT,
— LUGGAGE LOCKER EQUIPMENT,
— SHELTER EQUIPMENT,
— WAITING ROOM EQUIPMENT,
— SANITARY FACILITY EQUIPMENT.

7.7.9.3 Generic Equipment – Attributes and XSD

7.7.9.3.1 Equipment – Model Element

An item of equipment installed either fixed (PLACE EQUIPMENT) or on-board vehicles (VEHICLE EQUIPMENT). A service (LOCAL SERVICE such as LEFT LUGGAGE, TICKETING SERVICE) is considered as immaterial equipment as well.

Table 217 — *Equipment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	EQUIPMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*Equipment*	1:1	Identifier of EQUIPMENT.
	Name	*MultilingualString*	0:1	Name of EQUIPMENT.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative identifier of EQUIPMENT.
«AK»	***PublicCode***	*PrivateCode*	0:1	A Public code which may be displayed on equipment to identify it.
	Image	*xsd:anyURI*	0:1	Image of EQUIPMENT.
«FK»	***TypeOfEquipment-Ref***	*TypeOfEquipmentRef*	0:1	Reference to Type of EQUIPMENT.
	Description	*MultilingualString*	0:1	Description of EQUIPMENT.
	Note	*MultilingualString*	0:1	Note about EQUIPMENT.
«cntd»	***InfoLink***s	*InfoLink*	0:1*	INFOLINKs associated with EQUIPMENT.
	OutOfService	*xsd:boolean*	0:1	Whether the EQUIPMENT is out of service for protracted time. A separate Real time services should be used for short term outages. e.g. SIRI FM.

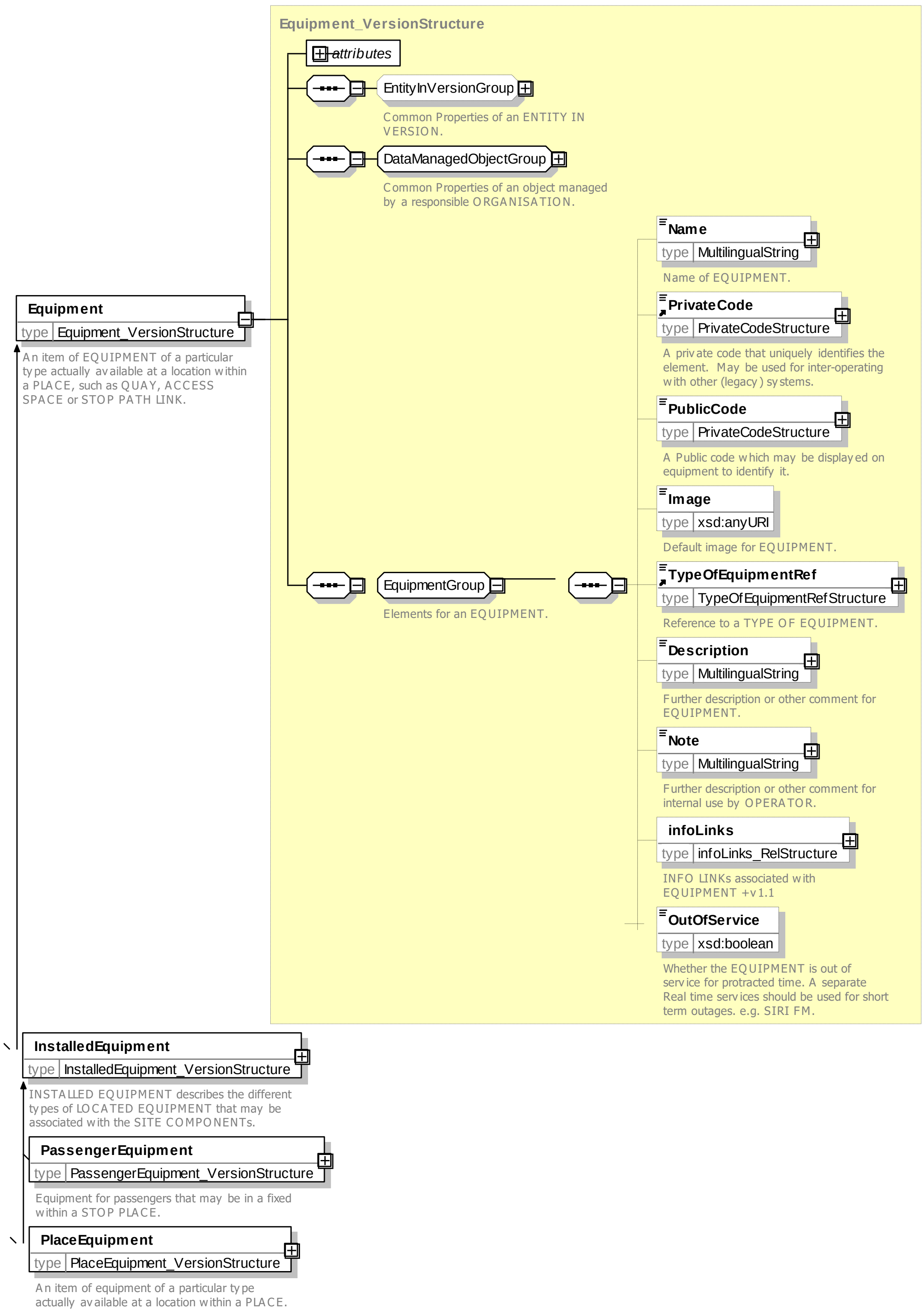


Figure 282 — *Equipment – XSD*

7.7.9.3.2 InstalledEquipment – Model Element

An item of equipment either fixed (PLACE EQUIPMENT) or on board i.e. associated with vehicles. This equipment is materialised as opposed to a service (LOCAL SERVICE) considered as an immaterial EQUIPMENT.

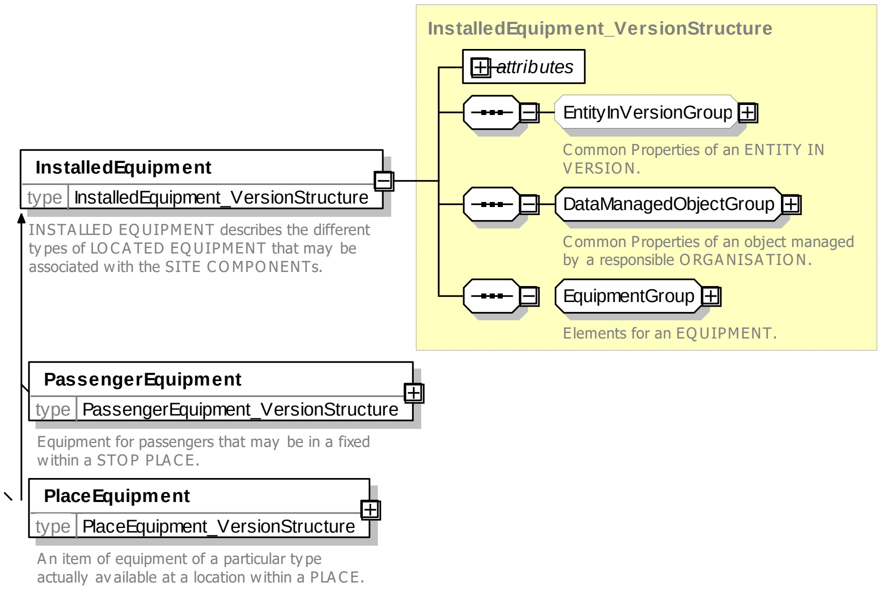


Figure 283 — *InstalledEquipment – XSD*

7.7.9.3.3 PlaceEquipment – Model Element

An item of equipment of a particular type actually available at a location within a PLACE.

Table 218 — *PlaceEquipment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	PLACE EQUIPMENT inherits from DATA MANAGED OBJECT.

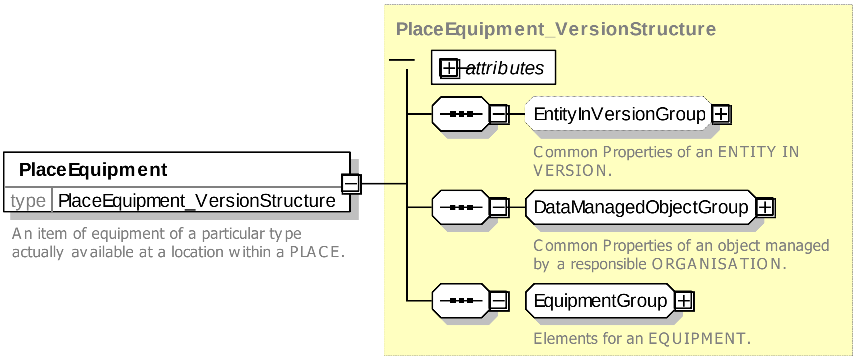


Figure 284 — *PlaceEquipment – XSD*

7.7.9.3.4 PassengerEquipment – Model Element

An item of equipment of a particular type actually available at a location within a PLACE or a VEHICLE.

Table 219 — *PassengerEquipment – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	<u>*Equipment*</u>	::>	PASSENGER EQUIPMENT inherits from EQUIPMENT.
	Fixed	*xsd:boolean*	0:1	Whether this Equipment is fixed at a PLACE or in a mobile vehicle.

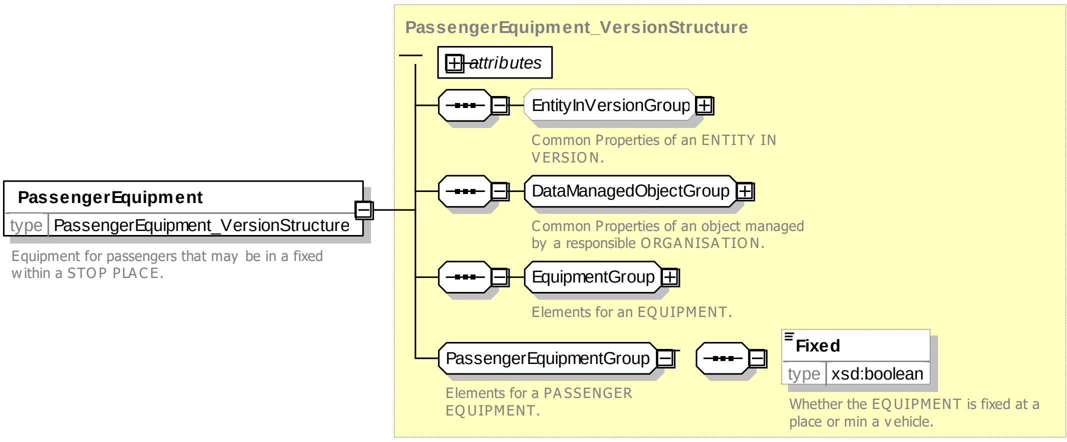


Figure 285 — *PassengerEquipment – XSD*

7.7.9.3.5 LocalService – Model Element

A named service relating to the use of the SITE or transport services at a particular location, for example porterage, assistance for disabled users, booking offices etc. The service may have a VALIDITY CONDITION associated with it. A LOCAL SERVICE is treated as a form of non-material EQUIPMENT.

Table 220 — *LocalService – Elements*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Equipment*</u>	::>	LOCAL SERVICE inherits from EQUIPMENT.
«PK»	***id***	*LocalServiceIdType*	1:1	Identifier of LOCAL SERVICE.
«cntd»	***typesOfService-Feature***	<u>*TypeOfServiceFeature*</u>	0:*	References to TYPEs OF SERVICE FEATURE that classify LOCAL SERVICE.

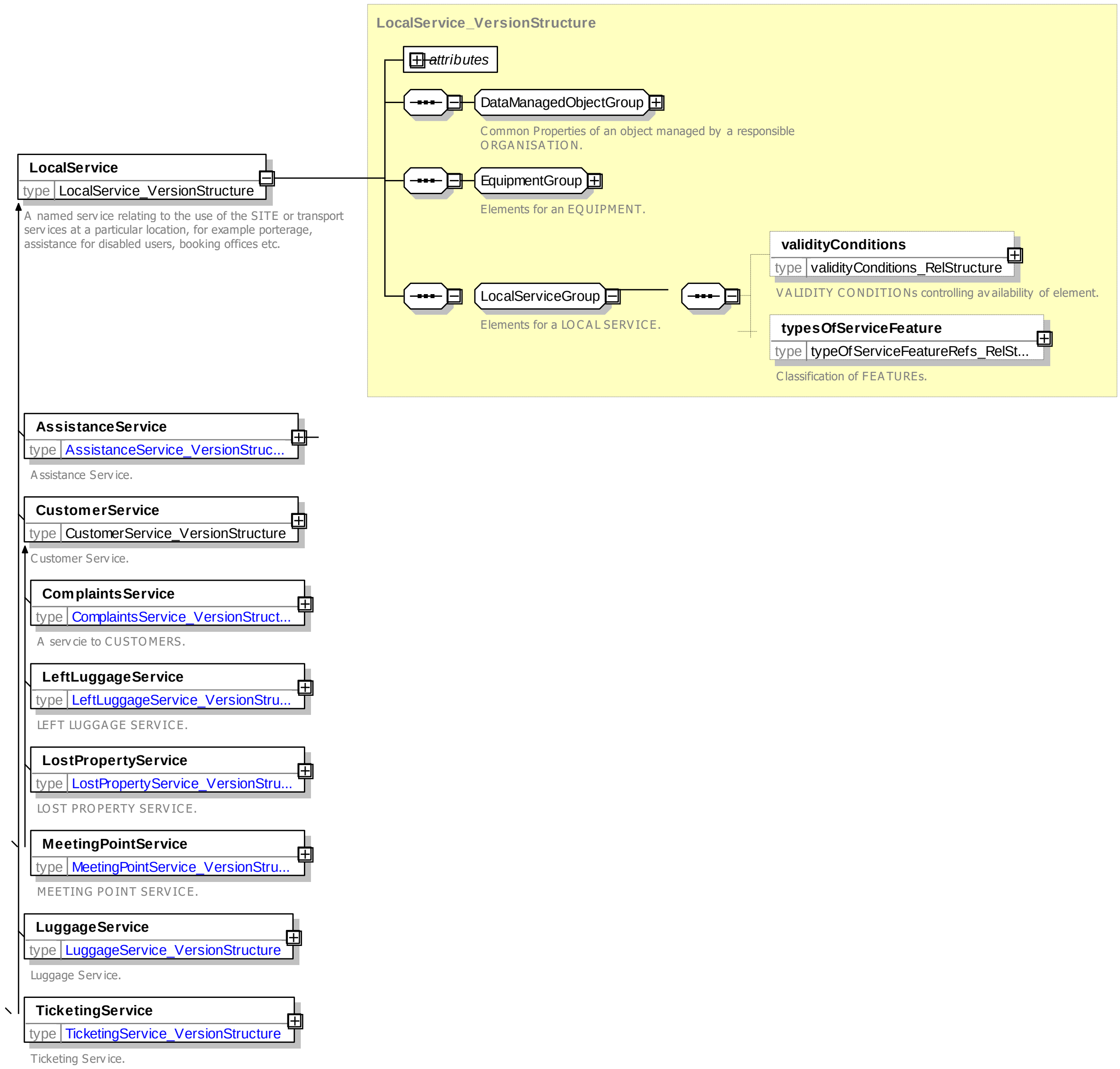


Figure 286 — *LocalService – XSD*

7.7.9.3.6 TypeOfEquipment – Model Element

A classification of equipment items to be installed at stop points or on-board vehicles, for instance.

Table 221 — *TypeOfEquipment – Element*

Classifi cation	Name	Type	Cardina lity	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF EQUIPMENT inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfEquipmentIdType*	1:1	Identifier of TYPE OF EQUIPMENT.

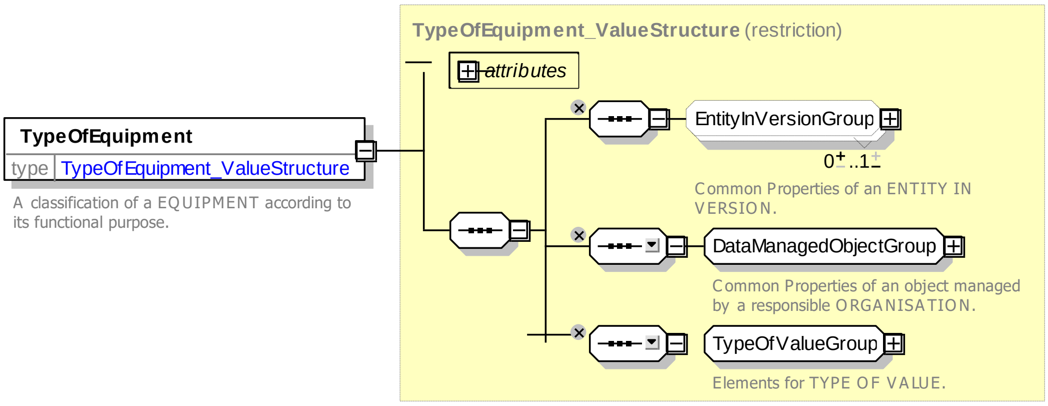


Figure 287 — *TypeOfEquipment – XSD*

7.7.9.3.7 EquipmentPlace – Model Element

A SITE COMPONENT containing EQUIPMENT.

Table 222 — *EquipmentPlace – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*Place*	::>	EQUIPMENT PLACE inherits from PLACE.
«PK»	***id***	*EquipmentPlaceId*	0:1	Identifier of EQUIPMENT PLACE.
«cntd»	***equipment-Positions***	*EquipmentPosition*	0:*	EQUIPMENT POSITIONs within EQUIPMENT PLACE.
«cntd»	***placeEquipments***	*Equipment*	0:*	EQUIPMENTs within EQUIPMENT PLACE.

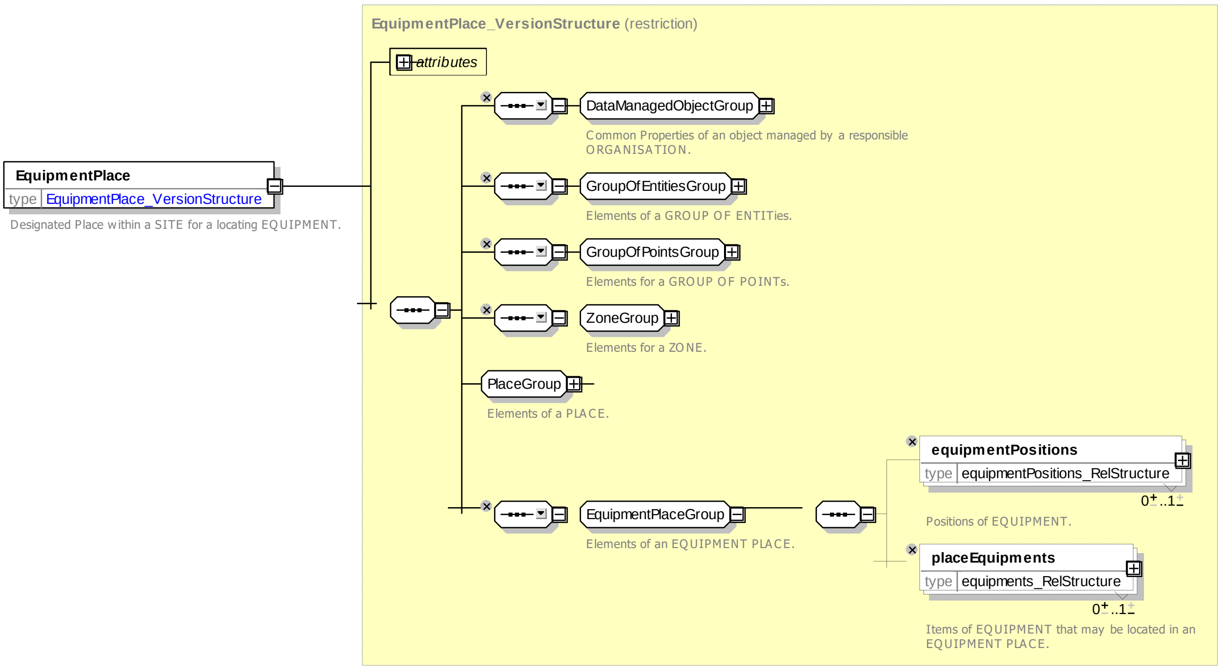


Figure 288 —*EquipmentPlace – XSD*

7.7.9.3.8 EquipmentPosition – Model Element

The precise position within EQUIPMENT PLACE where particular equipment is placed.

Table 223 — *EquipmentPosition – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	EQUIPMENT POSITION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*EquipmentPositionId*Type	0:1	Identifier of EQUIPMENT POSITION.
«FK»	***EquipmentRef***	*(EquipmentRef)*	1:1	Reference to EQUIPMENT for which this is the position.
	Description	*MultilingualString*	1:1	Description of EQUIPMENT POSITION.
	Location	*Location*	0:1	Location of EQUIPMENT.
«FK»	***ReferencePoint-Ref***	*(PointRef)*	0:1	Location of EQUIPMENT relative to a reference point. If absent, use top left corner of containing space. If present should be an entrance or other specific point within the space containing the EQUIPMENT.
	XOffset	*LengthType*	0:1	North South Horizontal offset from reference point.
	YOffset	*LengthType*	0:1	East West Horizontal offset from reference point.

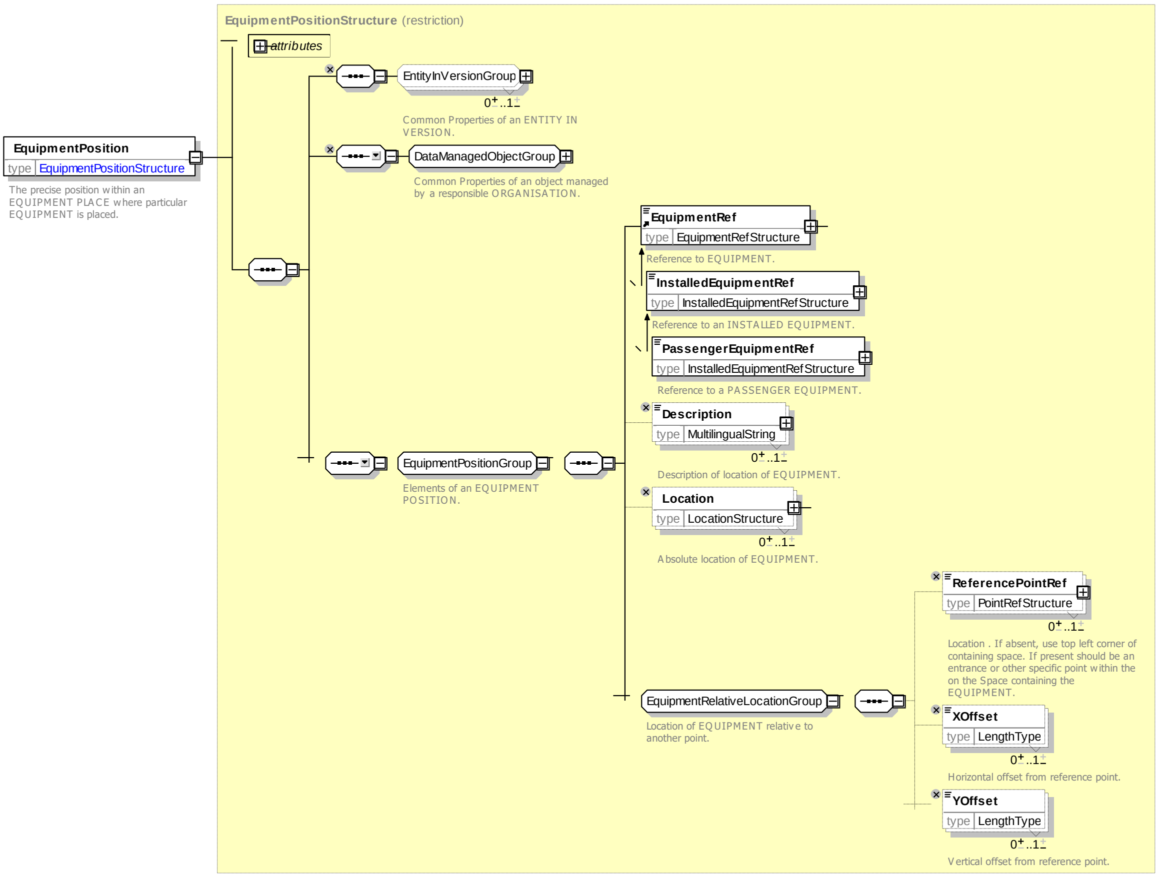


Figure 289 —*EquipmentPosition – XSD*

7.7.9.4 XML Examples of Equipment

7.7.9.4.1 Equipment – XML Fragment Example

The following example shows the placement of two signs.

EXAMPLE

```
 <equipmentPlaces>
    <EquipmentPlace      version="001"      created="2010-05-17T09:30:47Z"      modification="new"
id="ztb::bh0023_Rail-Q1_Sign1">
        <equipmentPositions>
            <EquipmentPosition version="any" id="ztb::bh0023_Rail-Q1_Sign1">
                <EquipmentRef version="any" ref="ztb:bh0023_Rail-Q1_Sign_01"/>
                <Description>Station  name  Sign  on  Platform  1,  10  metres  from  platform
entrance</Description>
                <ReferencePointRef ref="tbd:StopPlaceEntrance:bh0023_A1-E2_Sign_01"/>
                <XOffset>10</XOffset>
                <YOffset>1</YOffset>
            </EquipmentPosition>
            <EquipmentPosition version="any" id="ztb::bh0023_Rail-Q1_Sign_02">
                <EquipmentRef version="any" ref="ztb:bh0023_Rail-Q1_Sign_02"/>
                <Description>ExitSigtn on Platform 1, 5 metres from platform entrance</Description>
                <ReferencePointRef       version="any"       ref="tbd:StopPlaceEntrance:bh0023_A1-
E2_Sign_011"/>
                <XOffset>5</XOffset>
                <YOffset>0</YOffset>
            </EquipmentPosition>
        </equipmentPositions>
        <placeEquipments>
            <StopPlaceSign version="any" id="ztb:bh0023_Rail-Q1_Sign_01">
                <Height>1</Height>
                <Width>2</Width>
                <HeightFromFloor>1.5</HeightFromFloor>
```

```
                <PlaceName>Reid</PlaceName>
            </StopPlaceSign>
            <OtherSign version="any" id="ztb:bh0023_Rail-Q1_Sign_02">
                <Height>1</Height>
                <Width>2</Width>
                <HeightFromFloor>1.5</HeightFromFloor>
                <Content>Ausgang</Content>
            </OtherSign>
        </placeEquipments>
    </EquipmentPlace>
</equipmentPlaces>
```

7.7.10 Additional Organisations

7.7.10.1 ADDITIONAL ORGANISATIONs – Conceptual MODEL

The additional Organisation ADDITIONAL ORGANISATION model describes additional ORGANISATION types other than OPERATOR & AUTHORITY, but that are also related to the execution of part of the public tranport services. The model depicts them as different institutions, the OTHER ORGANISATION pictures the possible relationships that can be involved in various types of the execution of a pubic transport contract.

A TRAVEL AGENT takes reservations.

A MANAGEMENT AGENT operates on behalf of another organisation, for example to collect data.

A SERVICED ORGANISATION is an organisation for whom a transport service is provided, for example a school or works and for which the schedule may vary according to whether the organisation is open for business. This is described through the ORGANISATION DAY TYPE (inheriting from DAY TYPE) and SERVICE CALENDAR for a given OPERATING PERIOD, with DAY TYPE ASSIGNMENT.

An ORGANISATION can be reached at a POSTAL ADDRESS or is generally located at an ADDRESS, and can be made up of several ORGANISATIONAL PARTs.

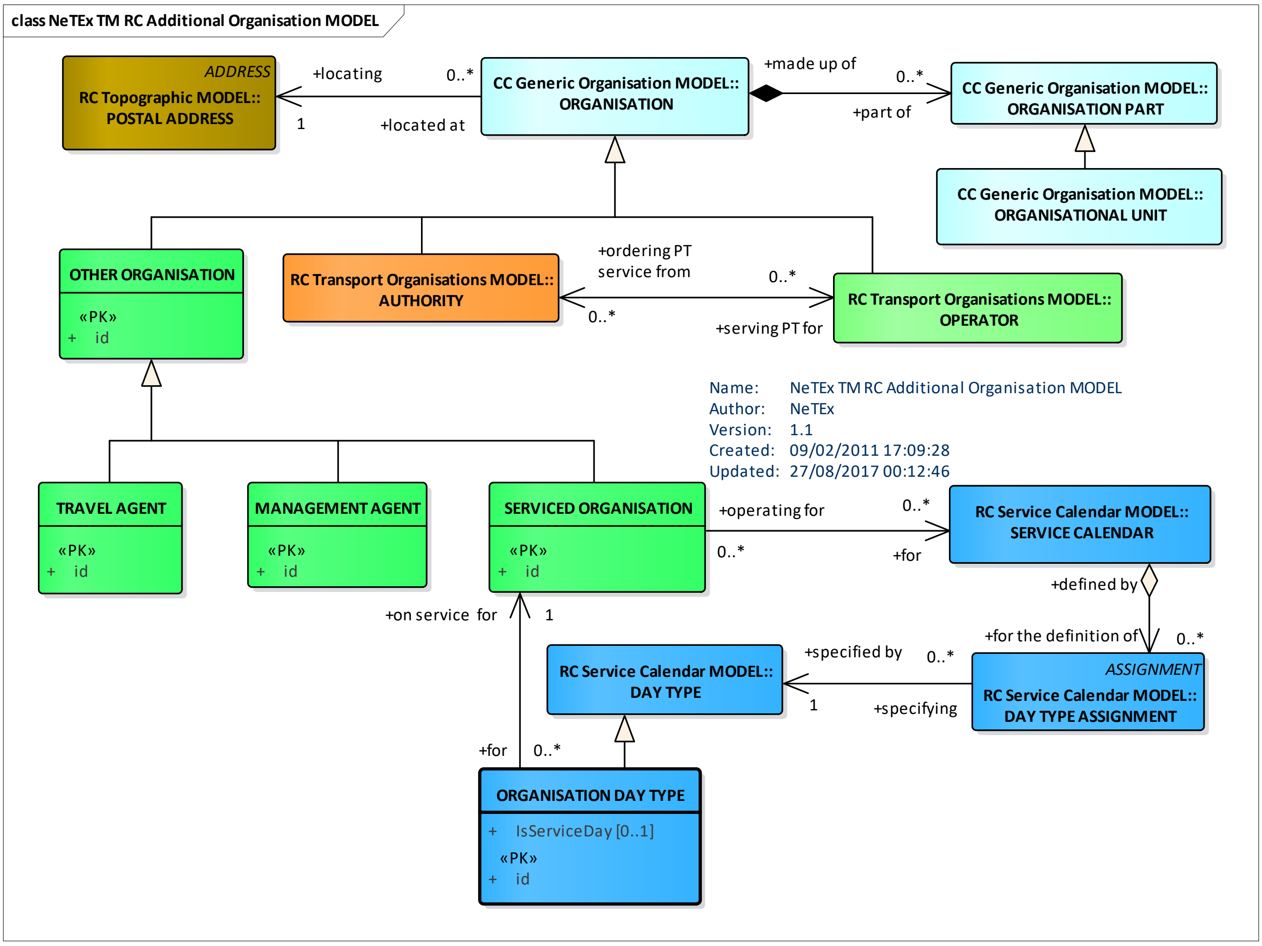


Figure 290 — Additional Organisations – Conceptual MODEL (UML)

7.7.10.2 Additional Organisations – Physical Model

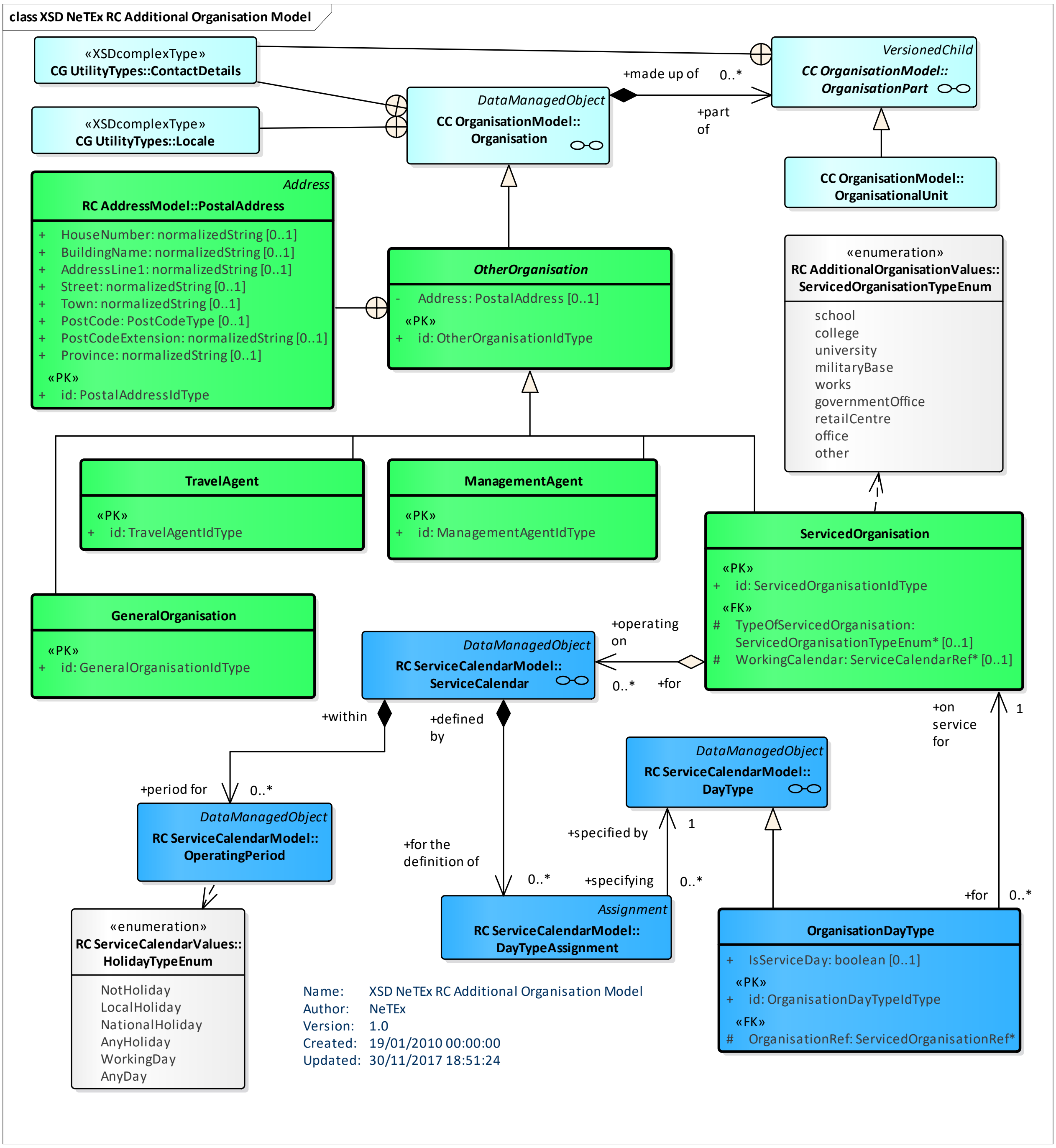


Figure 291 — Additional Organisations – Physical Model

7.7.10.3 Additional Organisations – Attributes and XSD

7.7.10.3.1 OtherOrganisation – Model Element

Common properties for an organisation other than an AUTHORITY or OPERATOR.

Table 224 — *OtherOrganisation – Element*

Classification	Name	Type	Cardinality	Description

::>	::>	*OtherOrganisation*	::>	OTHER ORGANISATION inherits from OTHER ORGANISATION.
«PK»	***id***	*OtherOrganisationIdType*	1:1	Identifier of OTHER ORGANISATION.
«cntd»	***Address***	*PostalAddress*	0:1	Postal ADDRESS of ORGANISATION.

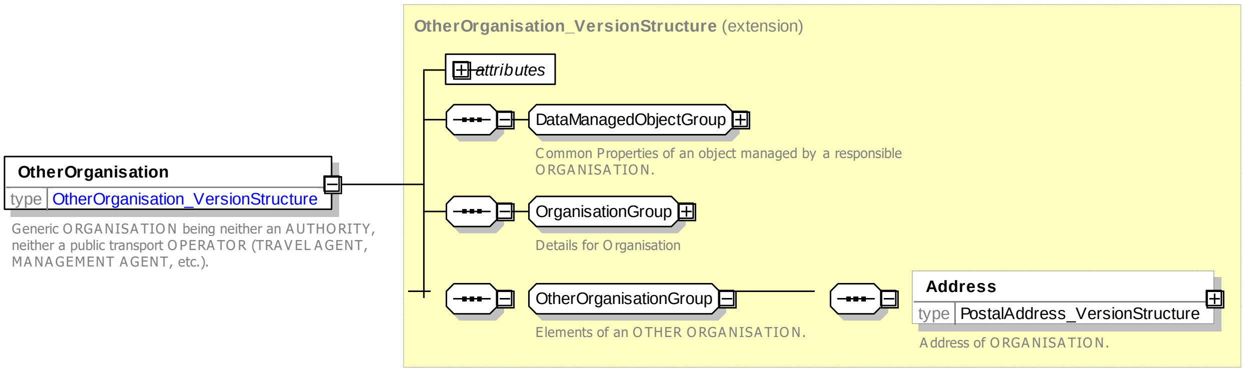


Figure 292 — *OtherOrganisation – XSD*

7.7.10.3.2 GeneralOrganisation – Model Element

A company providing public transport related or transport data services other than an AUTHORITY, OPERATOR, TRAVEL AGENT, etc.

Table 225 — *GeneralOrganisation – Element*

Classific ation	Name	Type	Cardinalit y	Description
::>	::>	*OtherOrganisation*	::>	GENERAL ORGANISATION inherits from OTHER ORGANISATION.
«PK»	***id***	*GeneralOrganisationIdType*	1:1	Identifier of OTHER ORGANISATION.

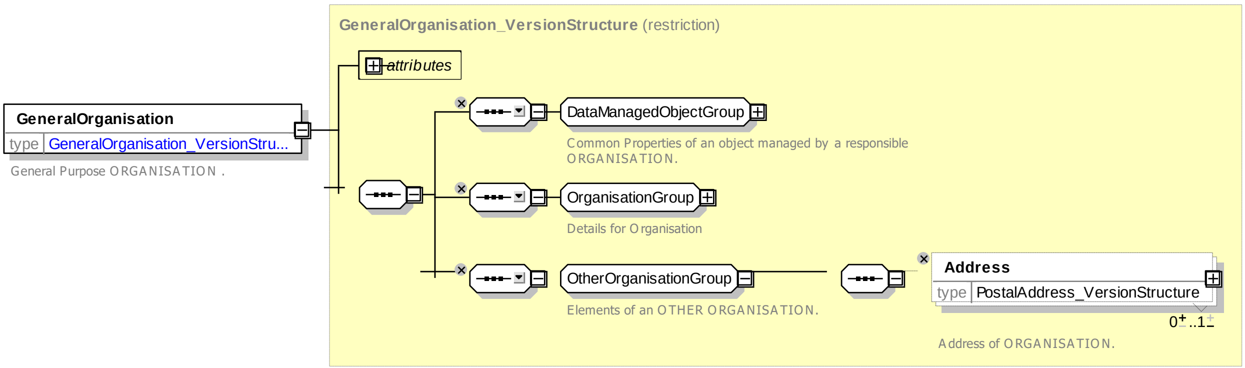


Figure 293 — *GeneralOrganisation – XSD*

7.7.10.3.3 ManagementAgent – Model Element

A company providing management services.

Table 226 — *ManagementAgent – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*OtherOrganisation*	::>	MANAGEMENT AGENT inherits from OTHER ORGANISATION.
«PK»	***id***	*ManagementAgentIdType*	1:1	Identifier of MANAGEMENT AGENT.

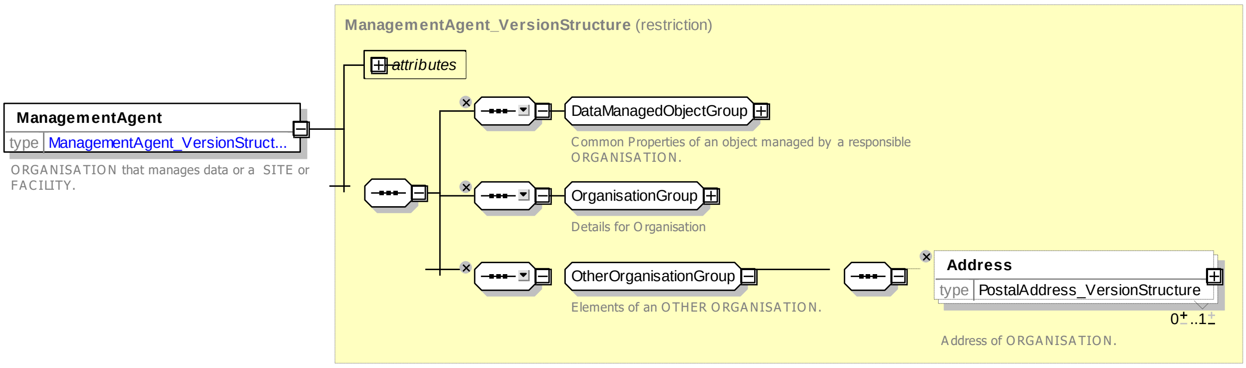


Figure 294 — *ManagementAgent – XSD*

7.7.10.3.4 TravelAgent – Model Element

A company providing travel agency services.

Table 227 — *TravelAgent – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*OtherOrganisation*	::>	TRAVEL AGENT inherits from OTHER ORGANISATION.
«PK»	***id***	*TravelAgentIdType*	1:1	Identifier of TRAVEL AGENT.

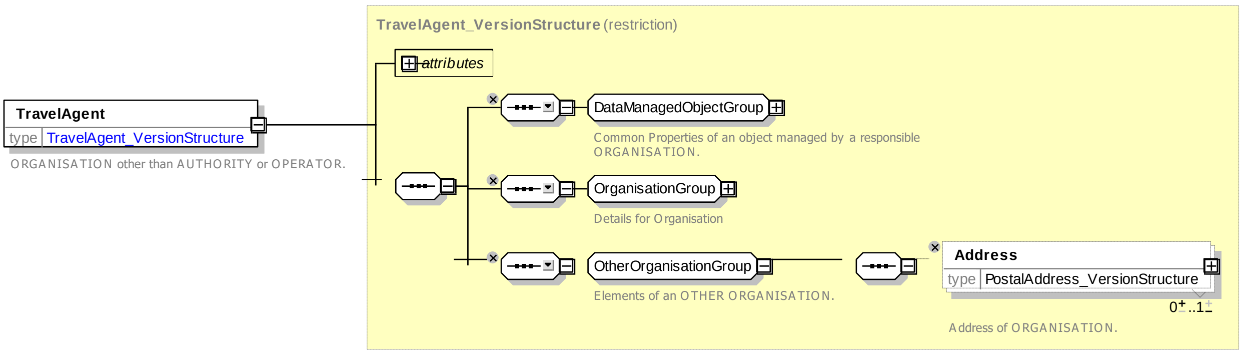


Figure 295 — *TravelAgent – XSD*

7.7.10.3.5 ServicedOrganisation – Model Element

A company providing various services.

Table 228 — *ServicedOrganisation – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*OtherOrganisation*	::>	SERVICED ORGANISATION inherits from OTHER ORGANISATION.
«PK»	***id***	*ServicedOrganisation-IdType*	1:1	Identifier of SERVICED ORGANISATION.
«FK»	***ServiceCalendarRef***	*ServiceCalendarRef*	0:1	Days SERVICED ORGANISATION is open.
«enum»	***ServiceOrganisationType***	*ServicedOrganisation-Enum*	0:1	Classification of SERVICED ORGANISATION. See allowed values below.

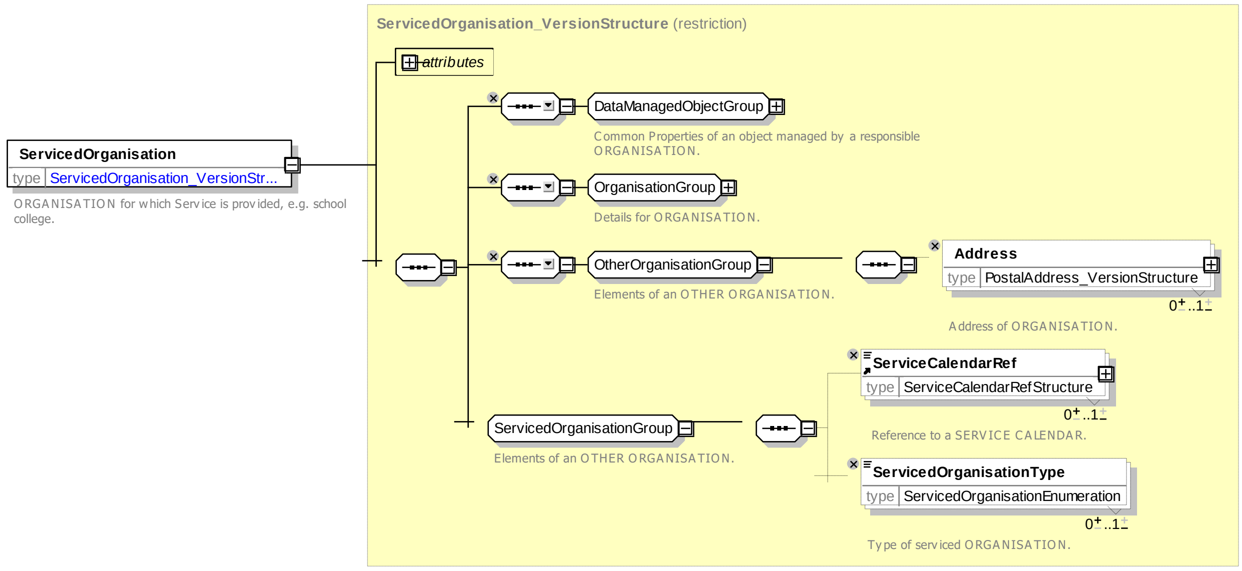


Figure 296 — *ServicedOrganisation – XSD*

7.7.10.3.5.1 ServicedOrganisationType – Allowed Values

Allowed values for TYPE OF SERVICED ORGANISATION (*ServicedOrganisationTypeEnumeration)*.

Table 229 — *ServicedOrganisationType – AllowedValues*

Value	Description
school	School.
college	Organisation is a college.
university	Organisation is a university.
militaryBase	Organisation is a military base.
works	Factory or works.
governmentOffice	Government office.
retailCentre	Retail centre.
office	Office.
other	Other

7.7.10.3.6 OrganisationDayType – Model Element

DAY TYPE that is defined in terms of operation or not operation of a referenced SERVICED ORGANISATION.

Table 230 — *OrganisationDayType – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DayType*	::>	ORGANISATION DAY TYPE inherits from DayType.
«PK»	***id***	*OrganisationDayType-IdType*	1:1	Identifier of DAY TYPE.
	IsServiceDay	*xsd:boolean*	1:1	Whether Transport Service is required on this day for ORGANISATION.
«FK»	**Serviced-OrganisationRef**	*ServicedOrganisationRef*	0:1	Reference to Parent ORGANISATION of SERVICED ORGANISATION.

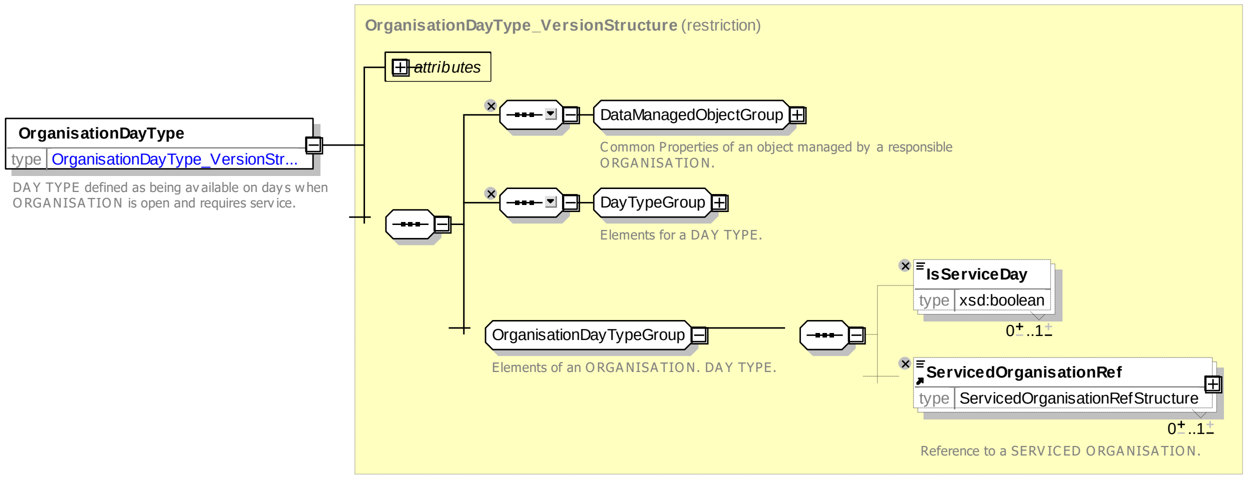


Figure 297 — *OrganisationDayType – XSD*

7.7.11 Vehicle Type

7.7.11.1 VEHICLE TYPE – Conceptual MODEL

The VEHICLE TYPE MODEL represents VEHICLES and their properties.

VEHICLES may be classified according to the vehicle scheduling requirements as to model and capacity and on board facilities (e.g. standard bus, double-deck, etc.). These same requirements may be attached to a SERVICE JOURNEY to indicate that it should be satisfied by a vehicle of that type.

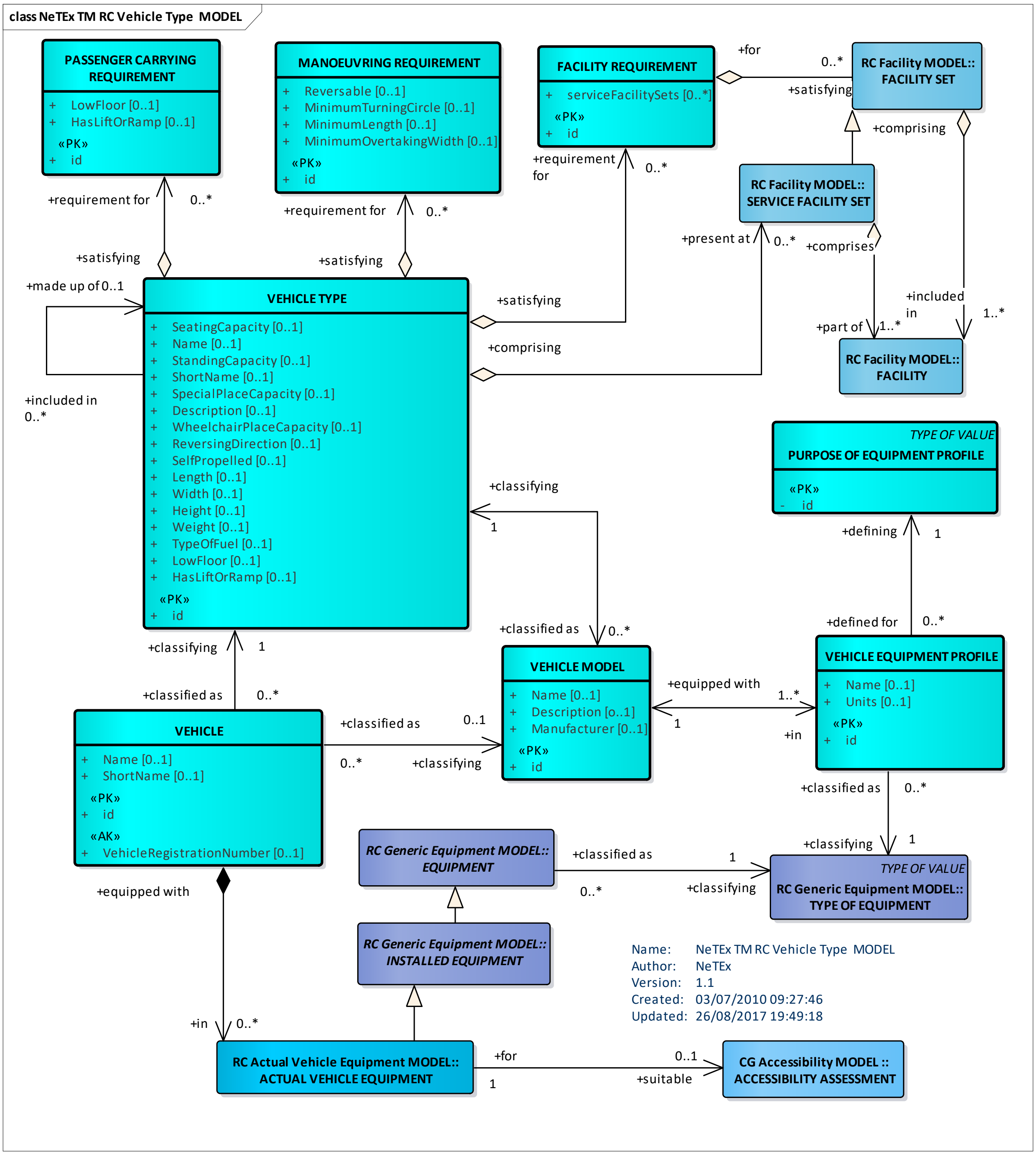


Figure 298 — *VEHICLE TYPE – Conceptual MODEL*

7.7.11.2 Vehicle Type – Physical Model

The following figure shows the detailed attributes of the VEHICLE & VEHICLE TYPE model

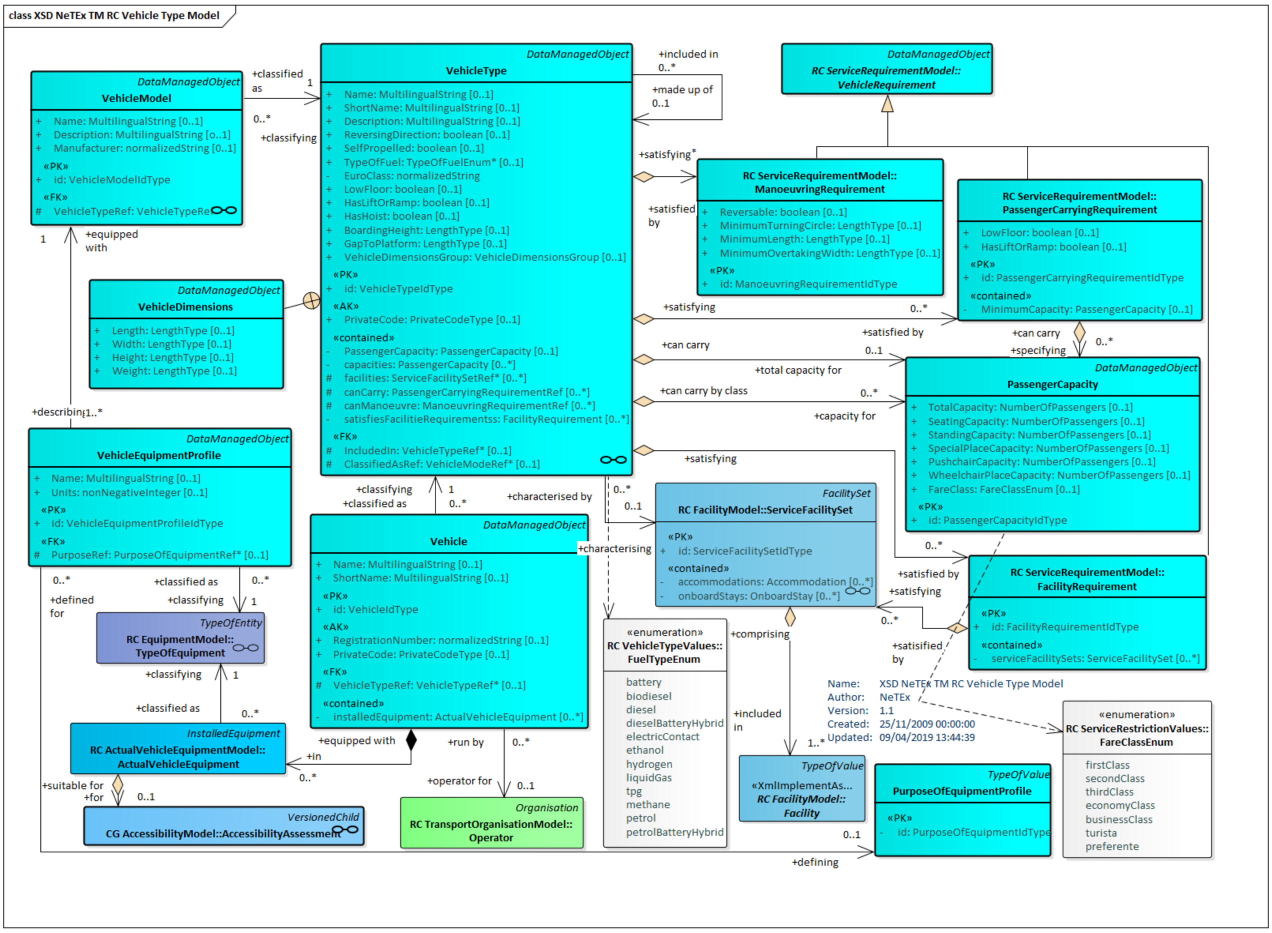


Figure 299 — Vehicle Type – Physical Model (UML)

7.7.11.3 Vehicle Type – Attributes and XSD

7.7.11.3.1 Vehicle – Model Element

A public transport vehicle used for carrying passengers.

Table 231 — *Vehicle – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	VEHICLE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*VehicleIdType*	1:1	Identifier of VEHICLE.
	Name	*MultilingualString*	0:1	Name of VEHICLE.
	ShortName	*MultilingualString*	0:1	Short name of VEHICLE.
«AK»	***RegistrationNumber***	*xsd:normalizedString*	0:1	Registration number of VEHICLE.
«AK»	***OperationalNumber***	*xsd:normalizedString*	0:1	Operational number of VEHICLE.

«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative identifier for VEHICLE.
«FK»	***OperatorRef***	*OperatorRef*	1:1	Reference to an OPERATOR.
«FK»	***VehicleTypeRef***	*VehicleTypeRef*	1:1	TYPE of VEHICLE.
«cntd»	***actualVehicle-Equipments***	*(EquipmentRef) | (Equipment)*		Actual vehicle equipment, i.e. EQUIPMENT for VEHICLE

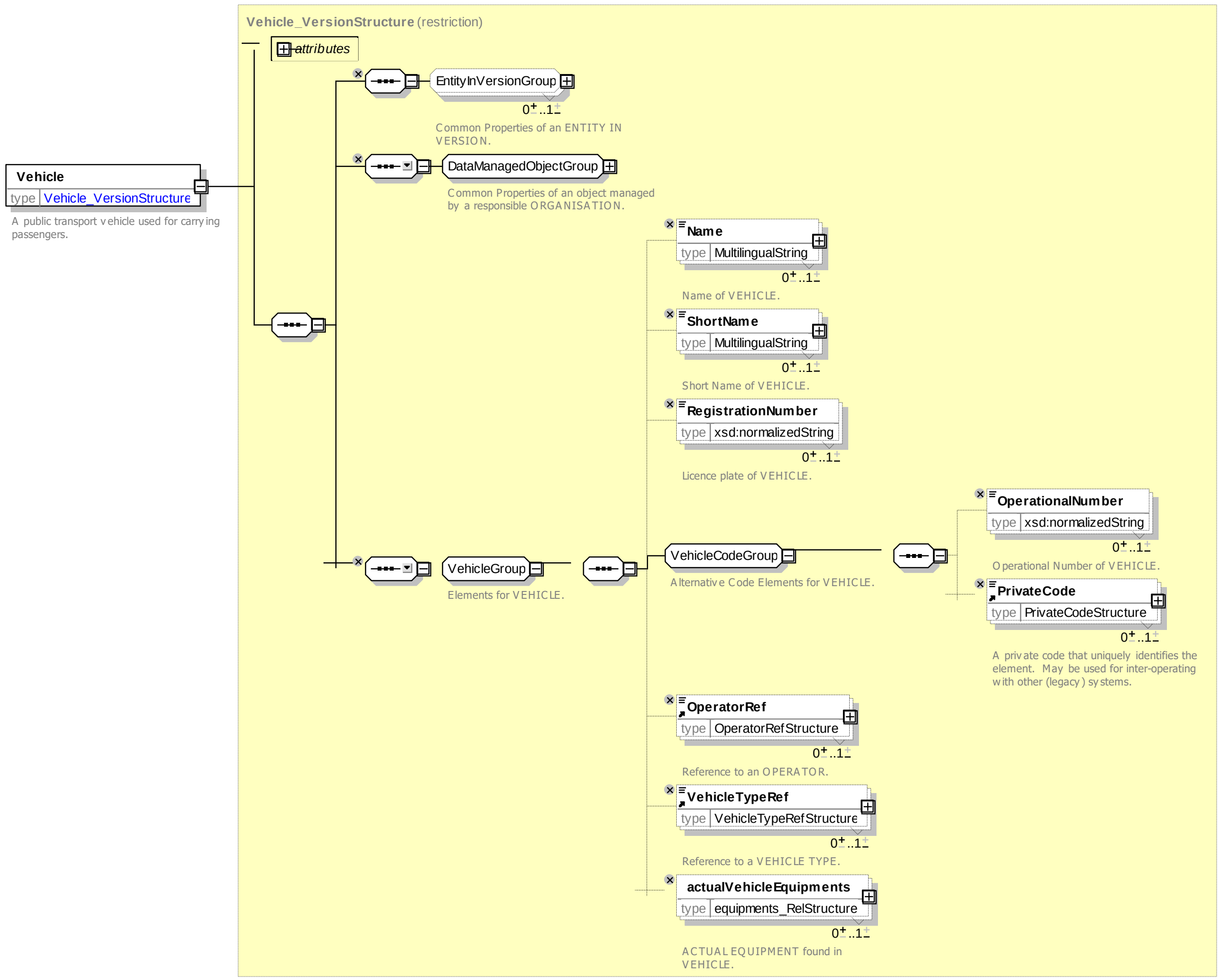


Figure 300 — *Vehicle – XSD*

7.7.11.3.2 VehicleType – Model Element

A classification of public transport vehicles according to the vehicle scheduling requirements in mode and capacity (e.g. standard bus, double-deck, etc.).

Table 232 — *VehicleType – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VEHICLE TYPE inherits from DATA MANAGED OBJECT.

«PK»	***id***	*VehicleTypeIdType*	1:1	Identifier of VEHICLE TYPE.
	Name	*MultilingualString*	0:1	Name of VEHICLE TYPE.
	ShortName	*MultilingualString*	0:1	Short Name of DAY TYPE.
	Description	*MultilingualString*	0:1	Description of VEHICLE TYPE.
«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative Identifier for VEHICLE TYPE.
XGRP	***VehicleType-PropertiesGroup***	***xmlGroup***	0:1	Elements describing the properties of a VEHICLE TYPE. See below.
«FK»	***IncludedIn***	*VehicleTypeRef*	0:1	Included in definition of VEHICLE TYPE.
«FK»	***ClassifiedAsRef***	*VehicleModelRef*	0:1	Classification of type as being of a particular VEHICLE MODEL.
«cntd»	***facilities***	*FacilitySet* | FacilitySetRef	0:*	Actual facilities on VEHICLE
«cntd»	***Vehicle-Requirements-Group***	***xmlGroup***	0:1	Elements describing requirements that the VEHICLE TYPE satisfies.

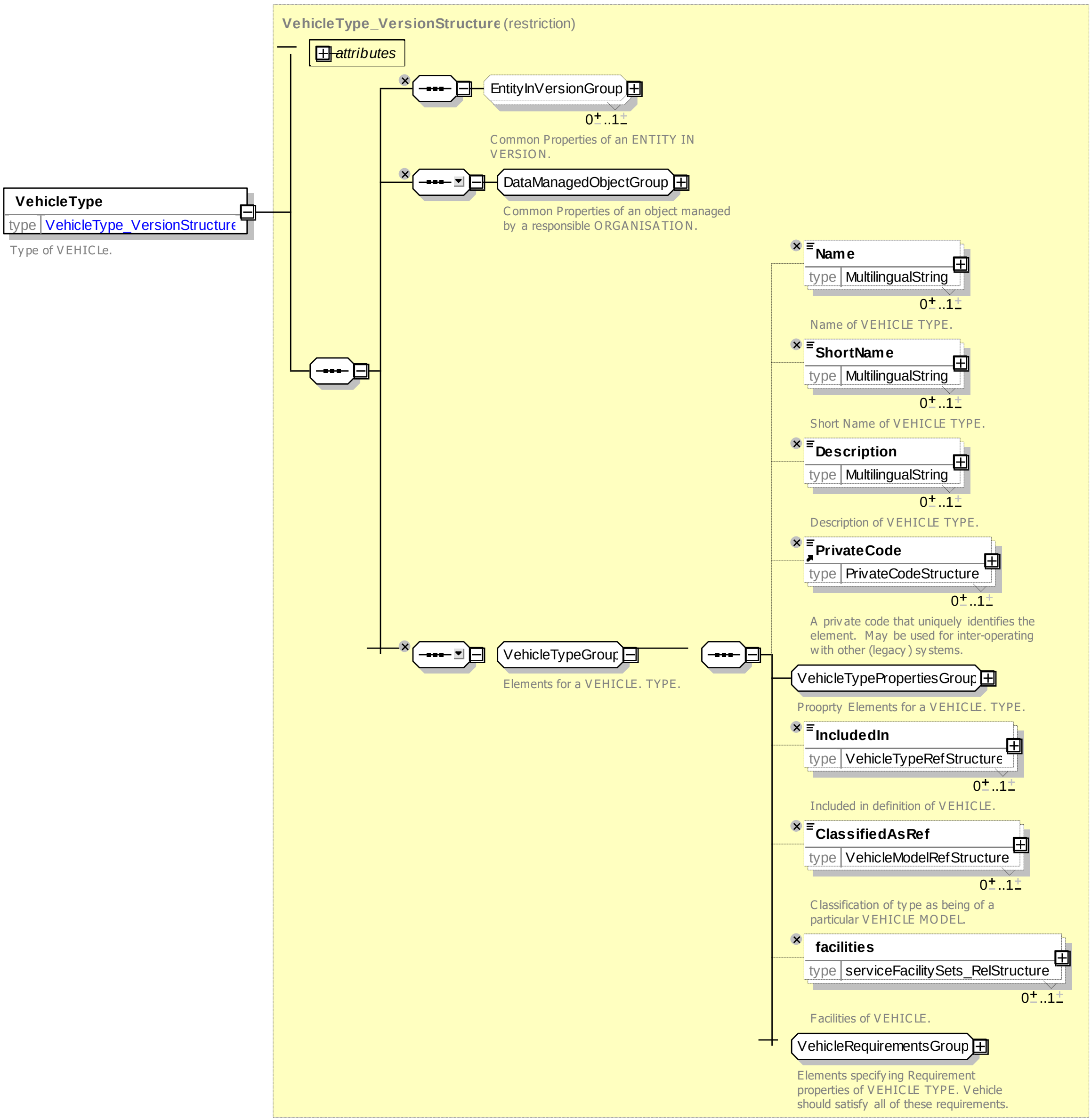


Figure 301 — *VehicleType – XSD*

7.7.11.3.2.1 VehicleTypePropertiesGroup – XML Group

Table 233 —*VehicleTypePropertiesGroup – Group*

Classification	Name	Type	Cardinality	Description
	Reversing-Direction	*boolean*	0:1	Whether VEHICLE TYPE has a reversing direction.
	SelfPropelled	*boolean*	0:1	Whether VEHICLE TYPE is self-propelled.
«enum»	***TypeOfFuel***	*TypeOfFuelEnum*	0:1	Type of Fuel of VEHICLE TYPE. See allowed values below.
	EuroClass	*xsd:normalizedString*	0:1	Euroclass of the vehicle type.

«cntd»	***Passenger-Capacity***	*PassengerCapacity*	0:1	Total passenger carrying capacity of VEHICLE TYPE.
«cntd»	***capacities***	*PassengerCapacity | PasengerCapacityRef*	0:*	Passenger carrying capacity of VEHICLE TYPE by FARE CLASS.
	LowFloor	*xsd:boolean*	0:1	Whether VEHICLE TYPE is low floor.
	HasLiftOrRamp	*xsd:boolean*	0:1	Whether VEHICLE TYPE has lift or ramp for wheelchair access.
	HasHoist	*xsd:boolean*	0:1	Whether vehicle has hoist for wheelchair access.
	BoardingHeight	*LengthType*	0:1	Maximum step height to board. + v1.1
	GapToPlatform	*LengthType*	0:1	Gap to platform. +v1.1
	Length	*LengthType*	0:1	Length of VEHICLE TYPE.
	Width	*LengthType*	0:1	The width of a VEHICLE of the type. +v1.1
	Height	*LengthType*	0:1	The height of a VEHICLE of the type. +v1.1
	Weight	*WeightType*	0:1	The weight of a VEHICLE of the type. +v1.1

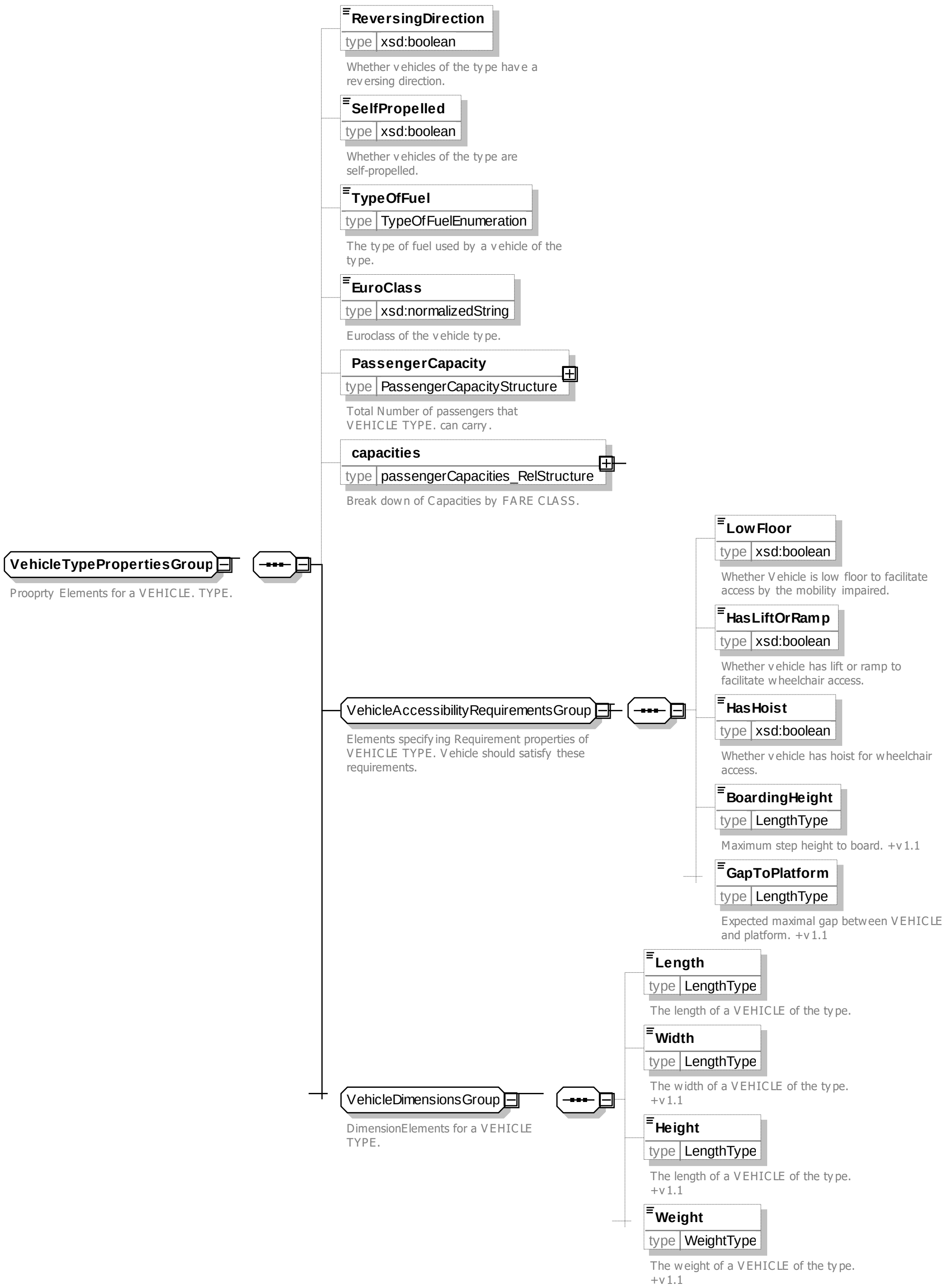


Figure 302 — *VehicleTypePropertiesGroup – XSD*

7.7.11.3.2.2 VehicleRequirementsGroup – XML Group

Elements describing requirements that the VEHICLE TYPE satisfies.

Table 234 — *VehicleRequirementsGroup – Group*

Classification	Name	Type	Cardinality	Description

«cntd»	***canCarry***	*PassengerCarrying-RequirementRef*	0:*	Passenger carrying requirements satisfied by VEHICLE TYPE.
«cntd»	***canManoeuvre***	*ManoeuvringRequirement Ref*	0:*	Manoeuvring requirements satisfied by VEHICLE TYPE.
«cntd»	***satisfiesFacilities***	*FacilityRequirement*	0:*	Facility requirements satisfied by VEHICLE TYPE.

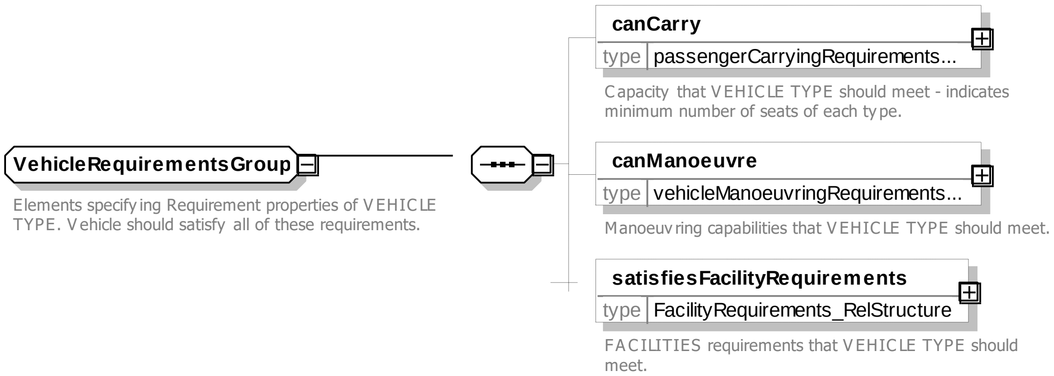


Figure 303 — *VehicleRequirementsGroup – XSD*

7.7.11.3.3 VehicleModel – Model Element

A classification of public transport vehicles of the same VEHICLE TYPE, e.g. according to equipment specifications or model generation.

Table 235 — *VehicleModel– Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VEHICLE MODEL inherits from DATA MANAGED OBJECT.
«PK»	***id***	*VehicleModelIdType*	1:1	Identifier of VEHICLE MODEL.
	Name	*MultilingualString*	0:1	Name of VEHICLE MODEL.
	Description	*MultilingualString*	o:1	Description of VEHICLE MODEL.
	Manufacturer	*xsd:normalizedString*	0:1	Manufacturer of VEHICLE MODEL.
«FK»	***VehicleTypeRef***	*VehicleTypeRef*	1:1	Reference to a VEHICLE TYPE of VEHICLE MODEL.

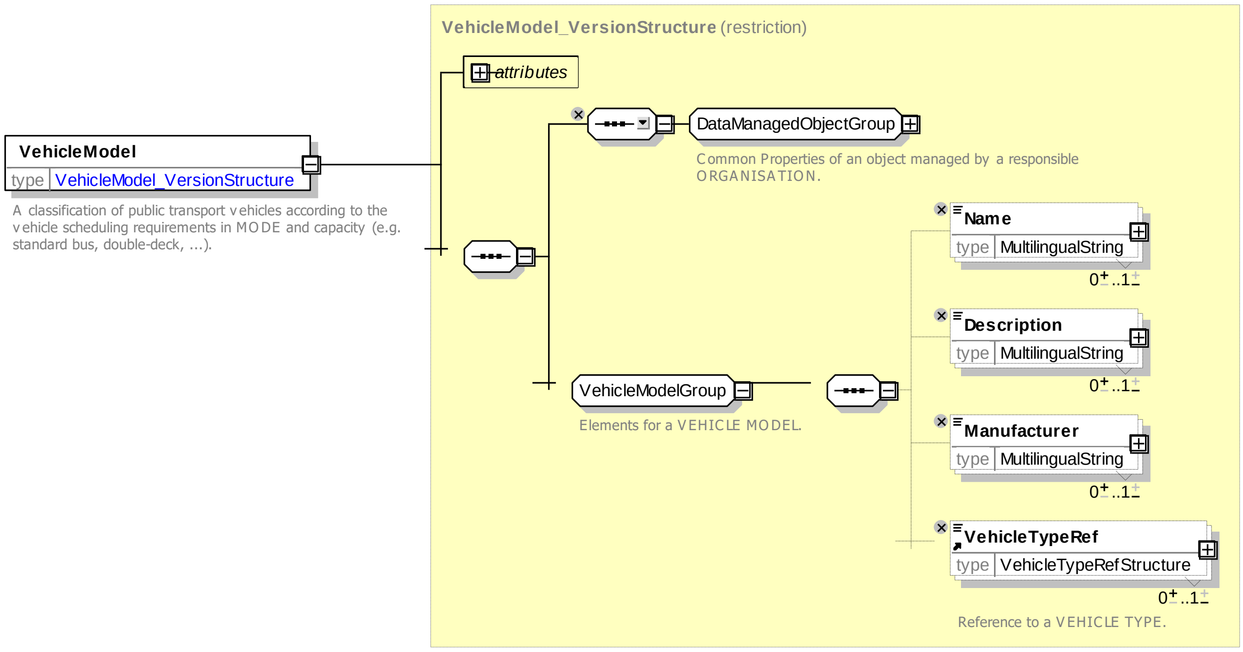


Figure 304 — *VehicleModel – XSD*

7.7.11.3.4 VehicleEquipmentProfile – Model Element

Each instantiation of this entity gives the number of items of one TYPE OF EQUIPMENT a VEHICLE MODEL should contain for a given PURPOSE OF EQUIPMENT PROFILE. The set of instantiations for one VEHICLE MODEL and one purpose gives one complete 'profile'.

Table 236 — *VehicleEquipmentProfile – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VEHICLE EQUIPMENT PROFILE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*VehicleEquipment-ProfileIdType*	1:1	Identifier of VEHICLE EQUIPMENT PROFILE.
	Name	*MultilingualString*	0:1	Text for VEHICLE EQUIPMENT PROFILE.
«FK»	***EquipmentRef***	*(EquipmentRef)*	0:1	Reference to a VEHICLE EQUIPMENT.
	Units	*xsd:nonNegativeInteger*	0:1	Units (number of instances of equipment), for VEHICLE EQUIPMENT PROFILE.
	Manufacturer	*xsd:normalizedString*	0:1	Manufacturer of VEHICLE EQUIPMENT.
«FK»	***PurposeOfEquipmentProfileRef***	*PurposeOfEquipment-ProfileRef*	0:1	Purpose of VEHICLE EQUIPMENT PROFILE.

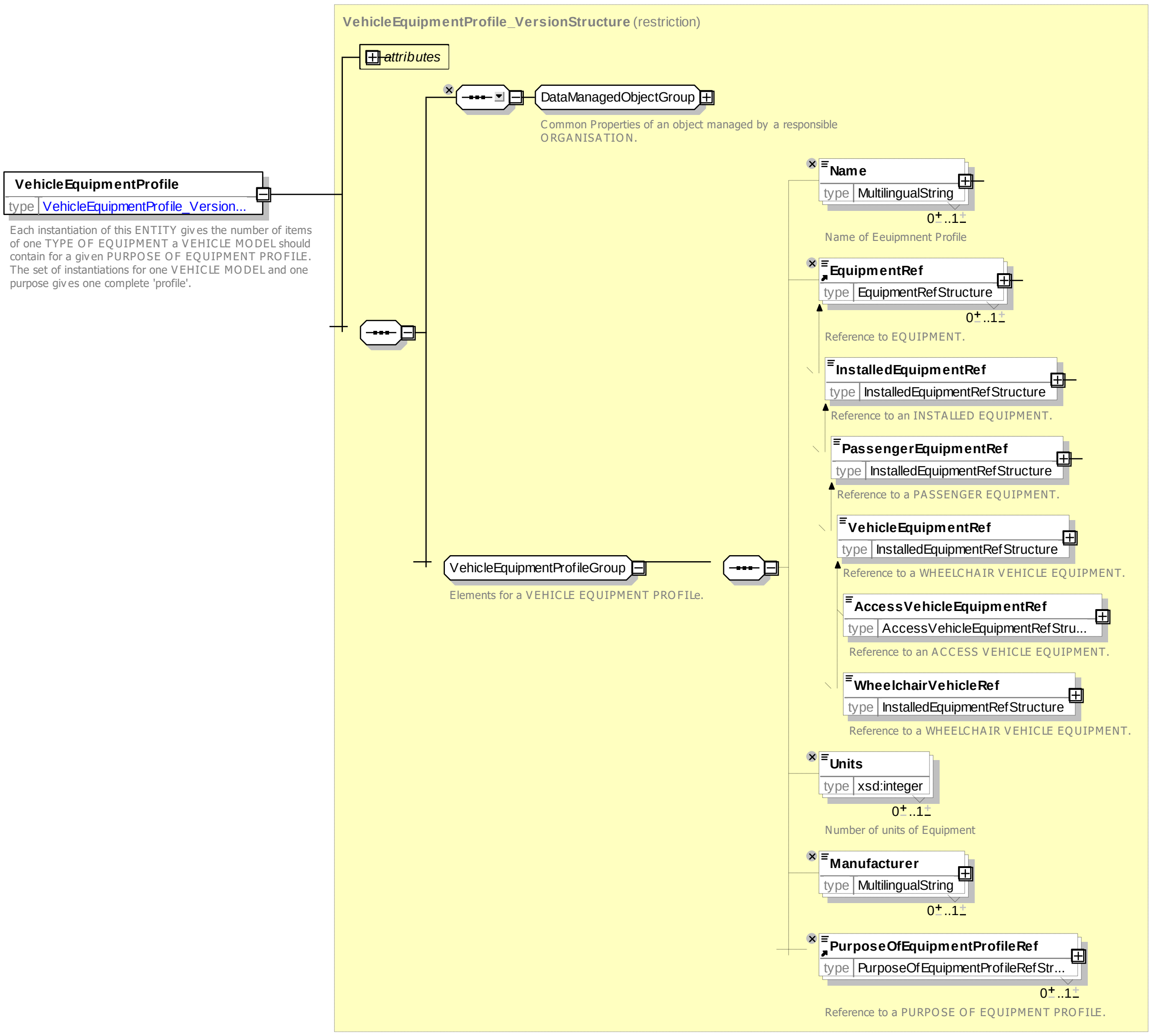


Figure 305 — *VehicleEquipmentProfile – XSD*

7.7.11.3.5 PassengerCapacity – Model Element

A classification of requirements for a public transport vehicle according to the passenger carrying capabilities of the vehicle.

Table 237 — *PassengerCapacity – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	PASSENGER CAPACITY inherits from DATA MANAGED OBJECT.
«PK»	***id***	*PassengerCapacity-IdType*	1:1	Identifier of PASSENGER CAPACITY.

«enum»	***FareClass***	*FareClassEnum*	0:1	FARE CLASS that this capacity applies to. The default is '*any*'. See ServiceRestrictions Model for allowed values.
	TotalCapacity	*NumberOfPassengers*	0:1	Total Capacity for a VEHICLE of given VEHICLE TYPE.
	SeatingCapacity	*NumberOfPassengers*	0:1	Maximum number of seated passengers that can be carried by a VEHICLE of this type.
	StandingCapacity	*NumberOfPassengers*	0:1	Maximum number of standing passengers that can be carried by a VEHICLE of this type.
	SpecialPlaceCapacity	*NumberOfPassengers*	0:1	Maximum number of passengers needing special places that can be carried by a VEHICLE of given type.
	PushchairCapacity	*NumberOfPassengers*	0:1	Maximum number of pushchairs needing special places that can be carried by a VEHICLE of given type.
	WheelchairPlace-Capacity	*NumberOfPassengers*	0:1	Maximum number of passengers needing wheelchair places that can be carried by a VEHICLE of given type.

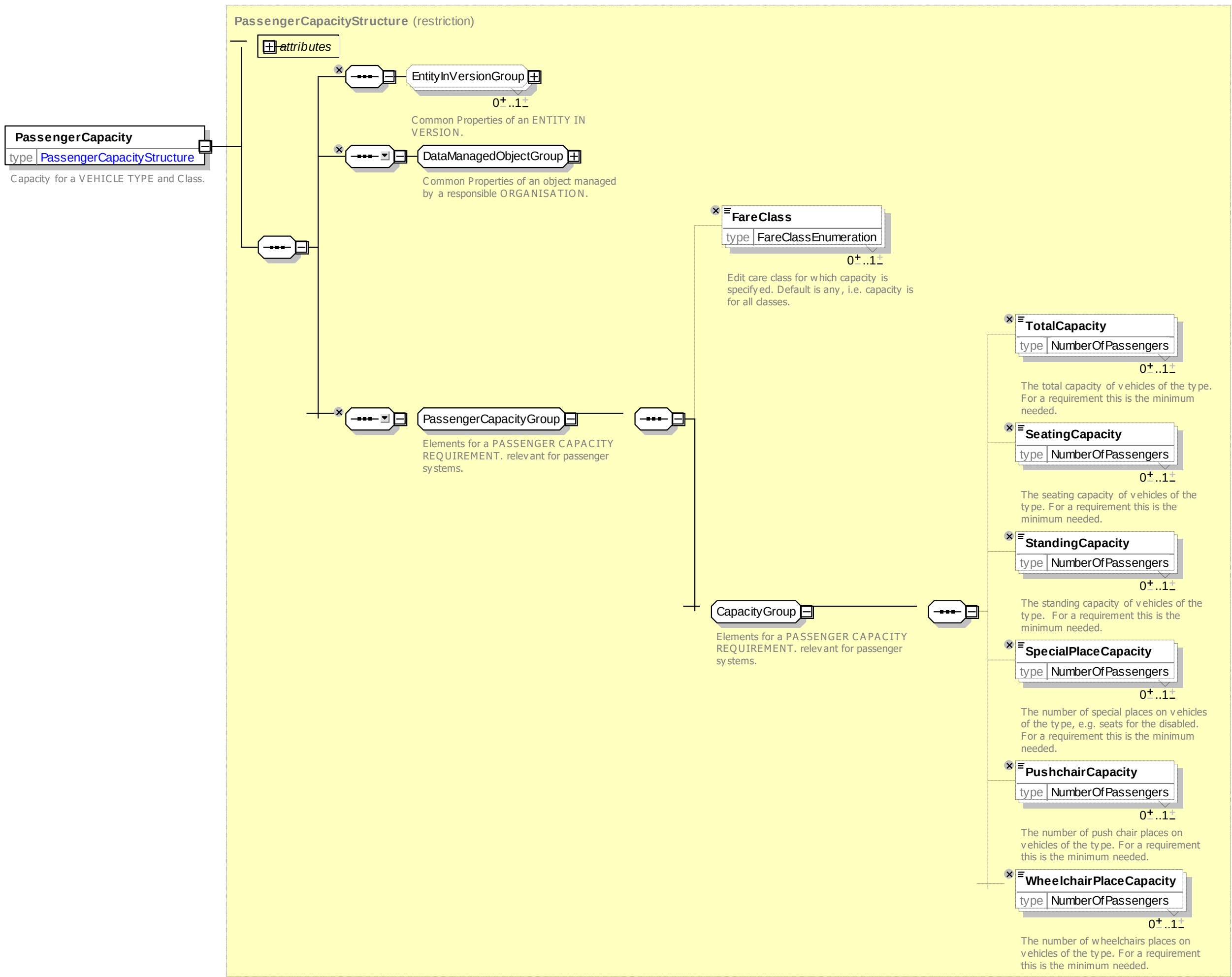


Figure 306 — *PassengerCapacity – XSD*

7.7.11.3.6 PurposeOfEquipmentProfile – Model Element

A functional purpose which requires a certain set of equipment of different types put together in a VEHICLE EQUIPMENT PROFILE or STOP POINT EQUIPMENT PROFILE.

Table 238 — *PurposeOfEquipmentProfile – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	PURPOSE OF EQUIPMENT PROFILE inherits from TYPE OF VALUE.
«PK»	***id***	*PurposeOfEquipmentIdType*	1:1	Identifier of PURPOSE OF EQUIPMENT PROFILE.

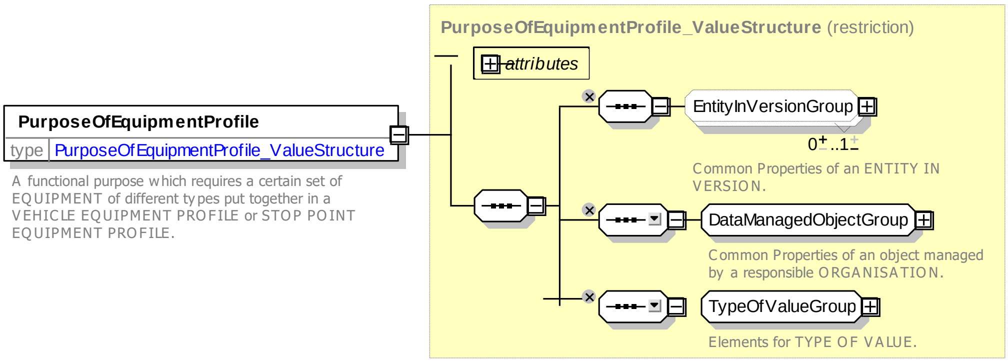


Figure 307 — *PurposeOfEquipmentProfile – XSD*

7.7.11.3.7 TypeOfFuel – Model Element

A classification of equipment items to be installed at stop points or on-board vehicles, for instance.

TYPE OF FUEL is implemented with an enumerated value.

7.7.11.3.7.1 FuelType – Allowed Values

Allowed values for ***FuelType*** (*FuelTypeEnum*).

Table 239 — FuelType – Allowed Values

Value	Description
battery	Battery Power
biodiesel	Biodiesel
diesel	Diesel fuel
dieselBatteryHybrid	Diesel & Battery Hybrid
electricContact	Electric requiring contact with rail or wires
ethanol	Ethanol fuel
hydrogen	Hydrogen
liquidGas	Liquid gas
tpg	TPG (Thermochemical Power Group)
methane	Methane
petrol	Petrol
petrolBatteryHybrid	Petrol and battery hybrid

7.7.11.3.8 VehicleRequirement – Model Element

A classification of requirements for a public transport vehicle to meet a service.

Table 240 — *VehicleRequirement – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	VEHICLE REQUIREMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*VehicleRequirementIdType*	1:1	Identifier of VEHICLE REQUIREMENT.
	Name	*MultilingualString*	0:1	Name of VEHICLE REQUIREMENT.

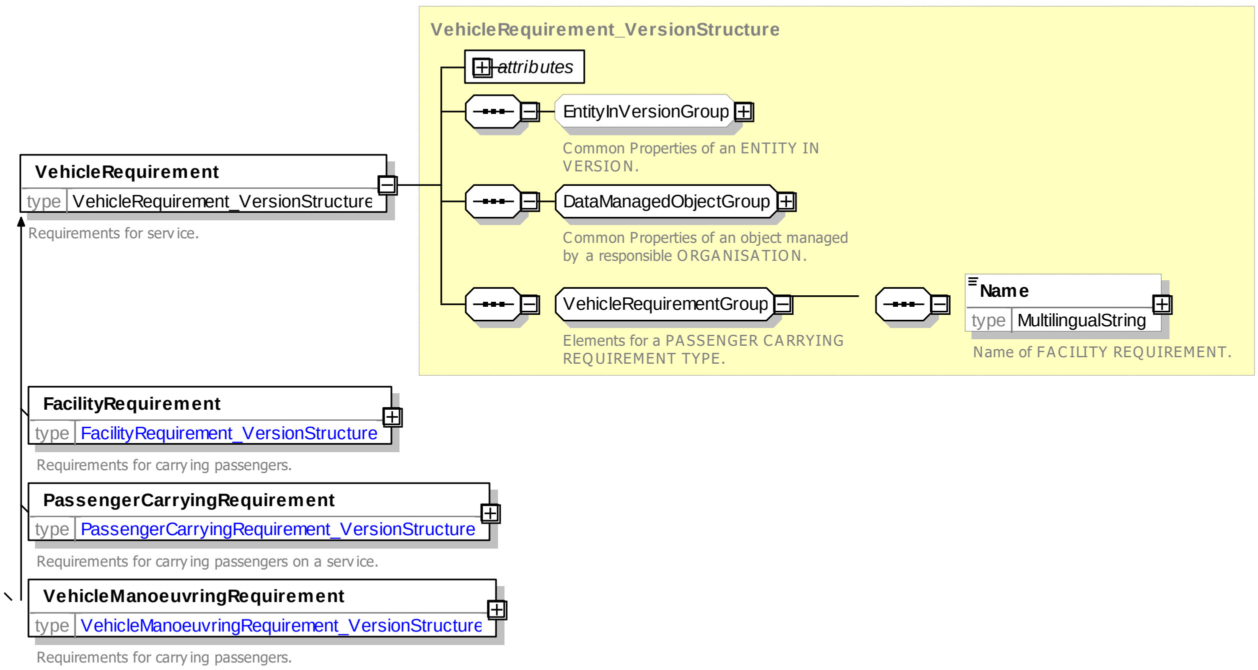


Figure 308 — *VehicleRequirement – XSD*

7.7.11.3.9 VehicleManoeuvringRequirement – Model Element

A classification of requirements for a public transport VEHICLE according to the manoeuvring capabilities of the vehicle.

Table 241 — *VehicleManoeuvringRequirement – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VehicleRequirement*	::>	VEHICLE MANOEUVRING REQUIREMENT inherits from VEHICLE REQUIREMENT.
«PK»	***id***	*Manoeuvring-RequirementIdType*	1:1	Identifier of VEHICLE MANOEUVRING REQUIREMENT.

	Reversible	*xsd:boolean*	0:1	Whether VEHICLE must be Reversible.
	MinimumTurningCircle	*LengthType*	0:1	Minimum turning circle needed to turn a VEHICLE.
	MinimumLength	*LengthType*	0:1	Minimum length needed to accommodate VEHICLE.
	MinimumOvertaking-Width	*LengthType*	0:1	Minimum width needed by VEHICLE TYPE to overtake.

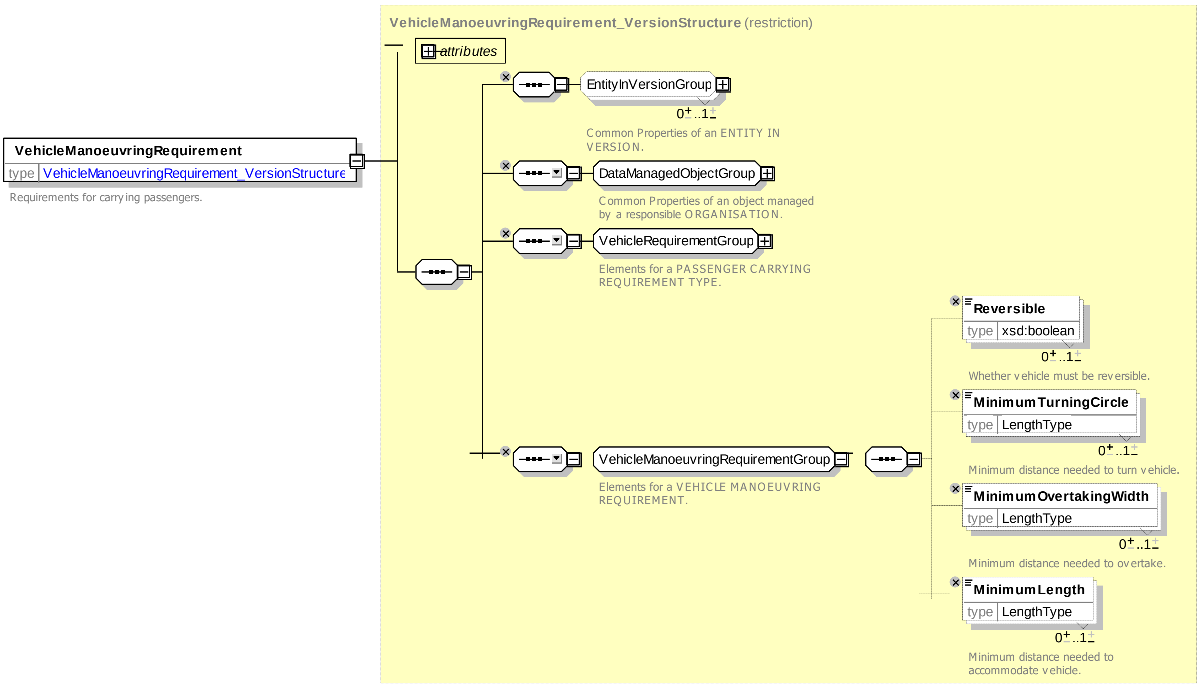


Figure 309 — *VehicleManouevringRequirement – XSD*

7.7.11.3.10 PassengerCarryingRequirement

A classification of requirements for a public transport vehicle according to the passenger carrying capabilities of the vehicle.

Table 242 — *PassengerCarryingRequirement – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VehicleRequirement*	::>	PASSENGER CARRYING REQUIREMENT inherits from VEHICLE REQUIREMENT.
«PK»	***id***	*PassengerCarrying-RequirementIdType*	1:1	Identifier of PASSENGER CARRYING REQUIREMENT.
«cntd»	***MinimumCapacity***	*PassengerCapacity*	1:1	Minimum number of passengers that needs to be provided.
	LowFloor	*xsd:boolean*	0:1	Whether VEHICLE needs to be low floor.

	HasLiftOrRamp	*xsd:boolean*	0:1	Whether VEHICLE needs to have lift or ramp for wheelchair access.

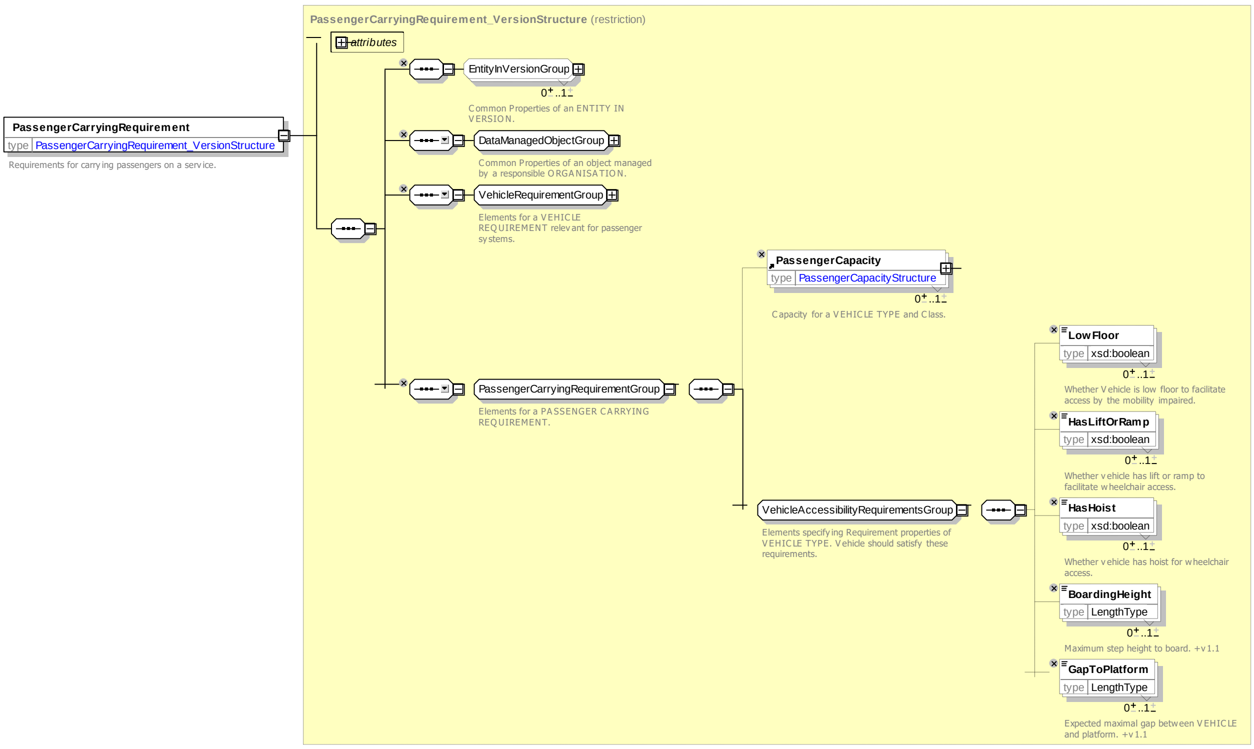


Figure 310 — *PassengerCarryingRequirement – XSD*

7.7.11.3.11 FacilityRequirement – Model Element

A classification of public transport vehicles according to the facilities available on the VEHICLE.

Table 243 — *FacilityRequirement – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VehicleRequirement*	::>	FACILITY REQUIREMENT inherits from VEHICLE REQUIREMENT.
«PK»	***id***	*FacilityRequirementIdType*	1:1	Identifier of a FACILITY REQUIREMENT.
«cntd»	***facilitySets***	*ServiceFacilitySet*	0:*	FACILITIes required on service.

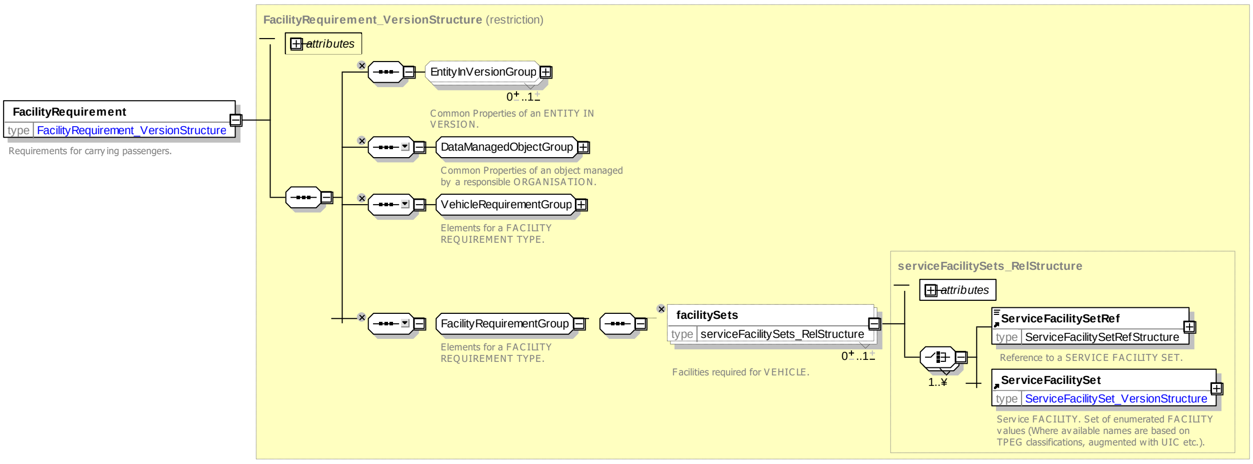


Figure 311 — *FacilityRequirement – XSD*

7.7.12 Actual Vehicle Equipment

7.7.12.1 ACTUAL VEHICLE EQUIPMENT – Conceptual MODEL

The ACTUAL VEHICLE EQUIPMENT specifies the type of EQUIPMENT to use in a given vehicle.

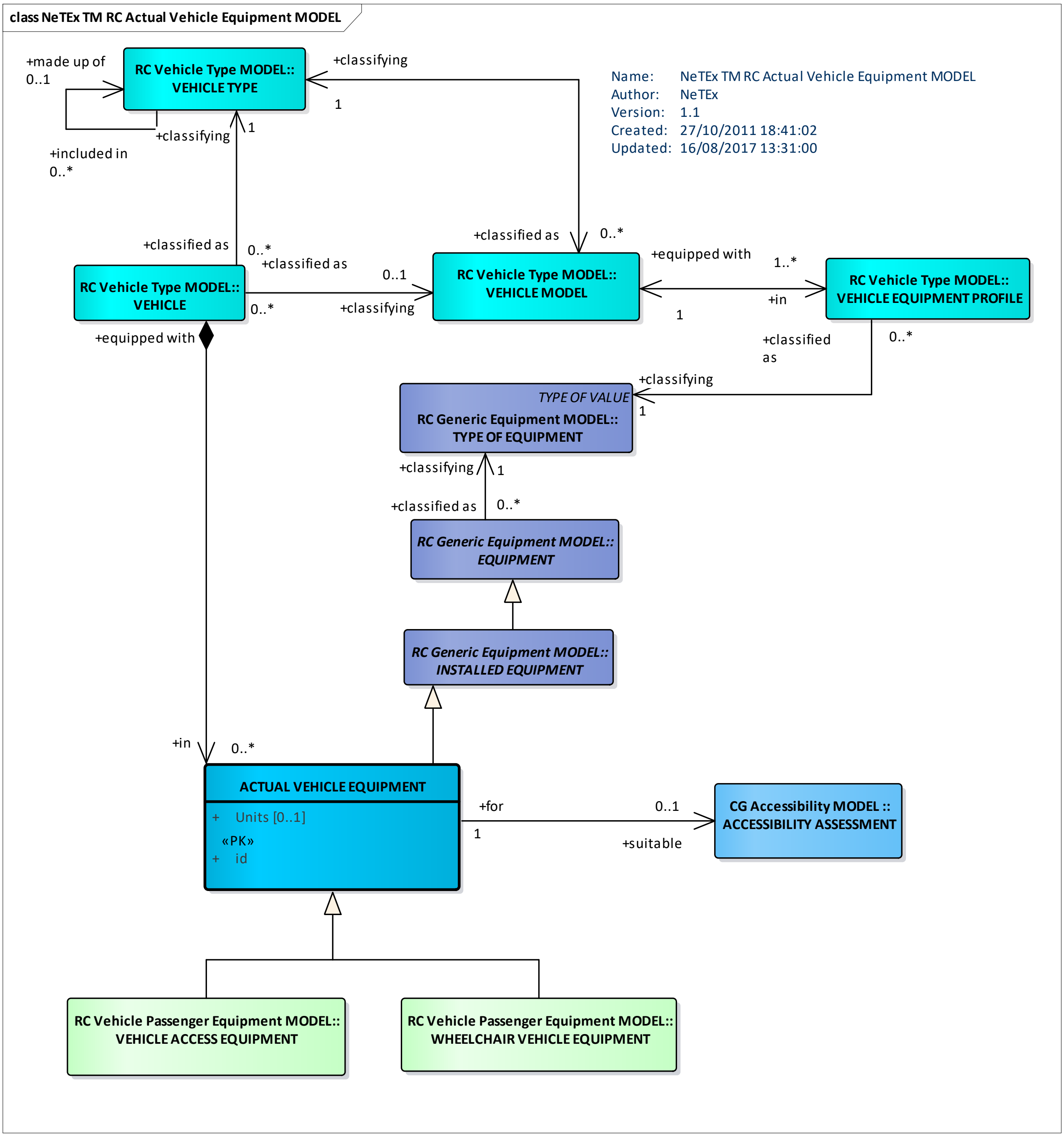


Figure 312 — Actual Vehicle Equipment – Conceptual MODEL (UML)

7.7.12.2 Actual Vehicle Equipment – Physical Model

The following figure shows the Physical classes.

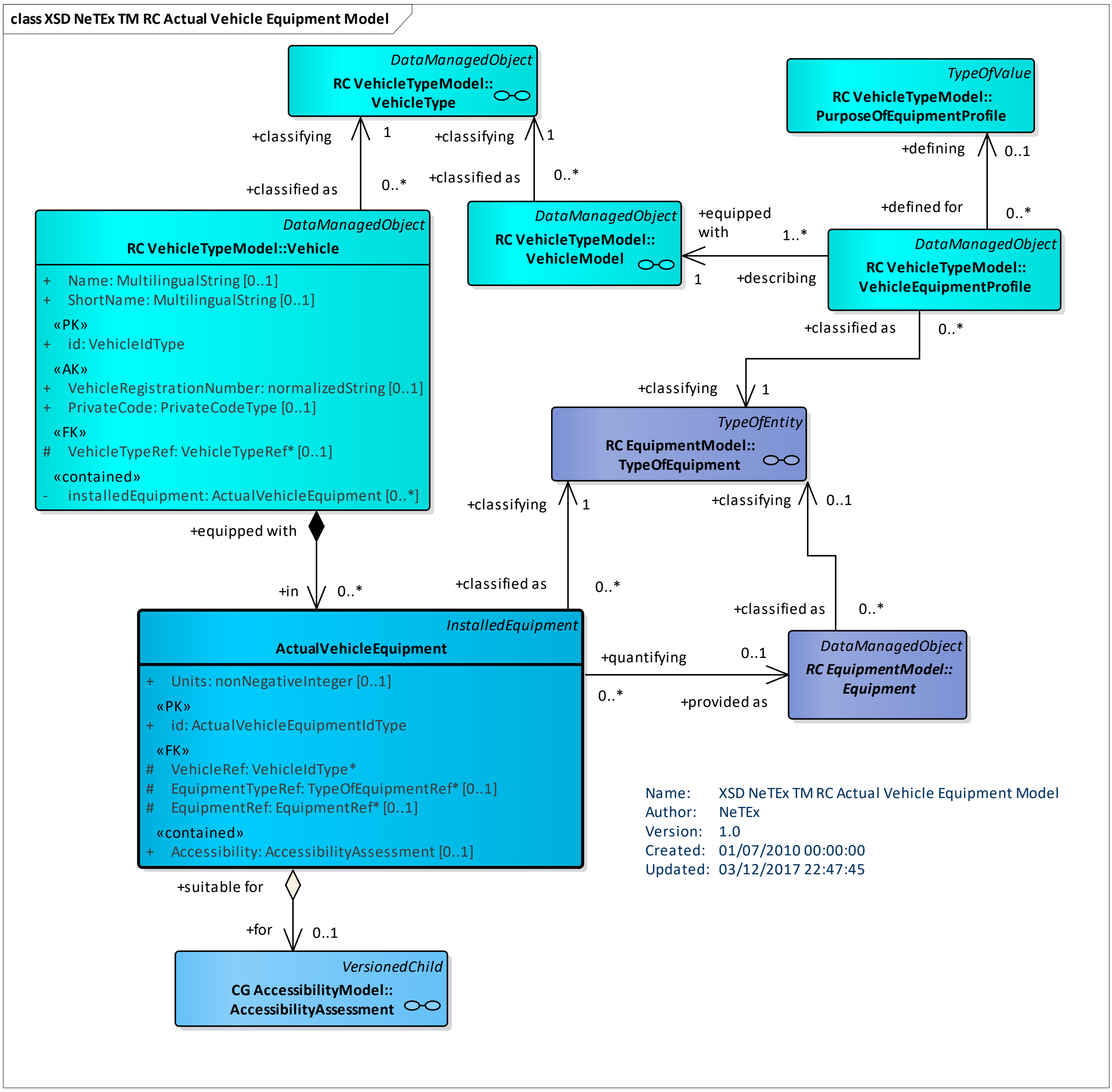


Figure 313 — Actual Vehicle Equipment – Physical Model (UML)

7.7.12.3 Actual Vehicle Equipment – Attributes and XSD

7.7.12.3.1 ActualVehicleEquipment – Model Element

An item of equipment of a particular type actually available in an individual VEHICLE.

Table 244 — *ActualVehicleEquipment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*InstalledEquipment*	::>	ACTUAL VEHICLE EQUIPMMENT inherits from PASSENGER EQUIPMENT.

«PK»	***id***	*ActualVehicleEquipmentIdType*	1:1	Identifier of ACTUAL VEHICLE EQUIPMENT.
	Units	*xsd:nonNegativeInteger*	0:1	Number of instances of ACTUAL VEHICLE EQUIPMENT there are on vehicle.
«FK»	***VehicleTypeRef***	*VehicleTypeRef*	01:1	VEHICLE TYPE for which this is the ACTUAL VEHICLE EQUIPMENT.
«cntd»	***Accessibility-Assessment***	*AccessibilityAssessment*	0:1	ACCESSIBILITY ASSESSMENT of ACTUAL VEHICLE EQUIPMENT.

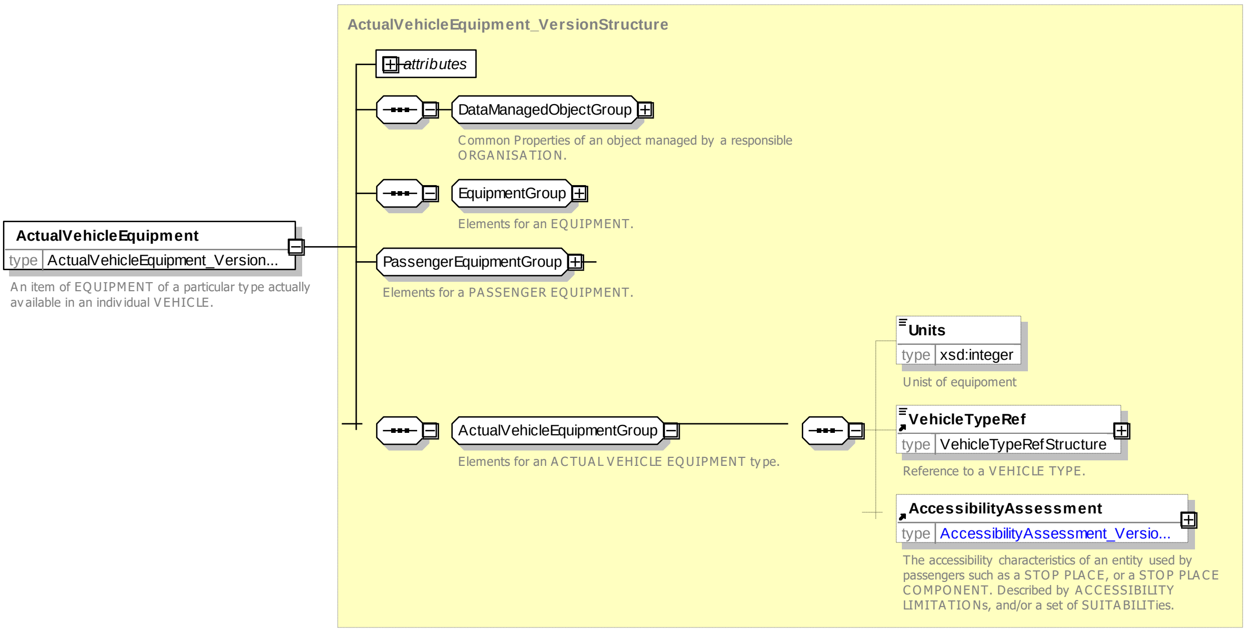


Figure 314 — *ActualVehicleEquipment – XSD*

7.7.13 Vehicle Passenger Equipment

7.7.13.1 VEHICLE PASSENGER EQUIPMENT – Conceptual MODEL

The VEHICLE PASSENGER EQUIPMENT describes boarding properties of a VEHICLE. WHEELCHAIR VEHICLE EQUIPMENT describes on-board capacity for wheelchairs.

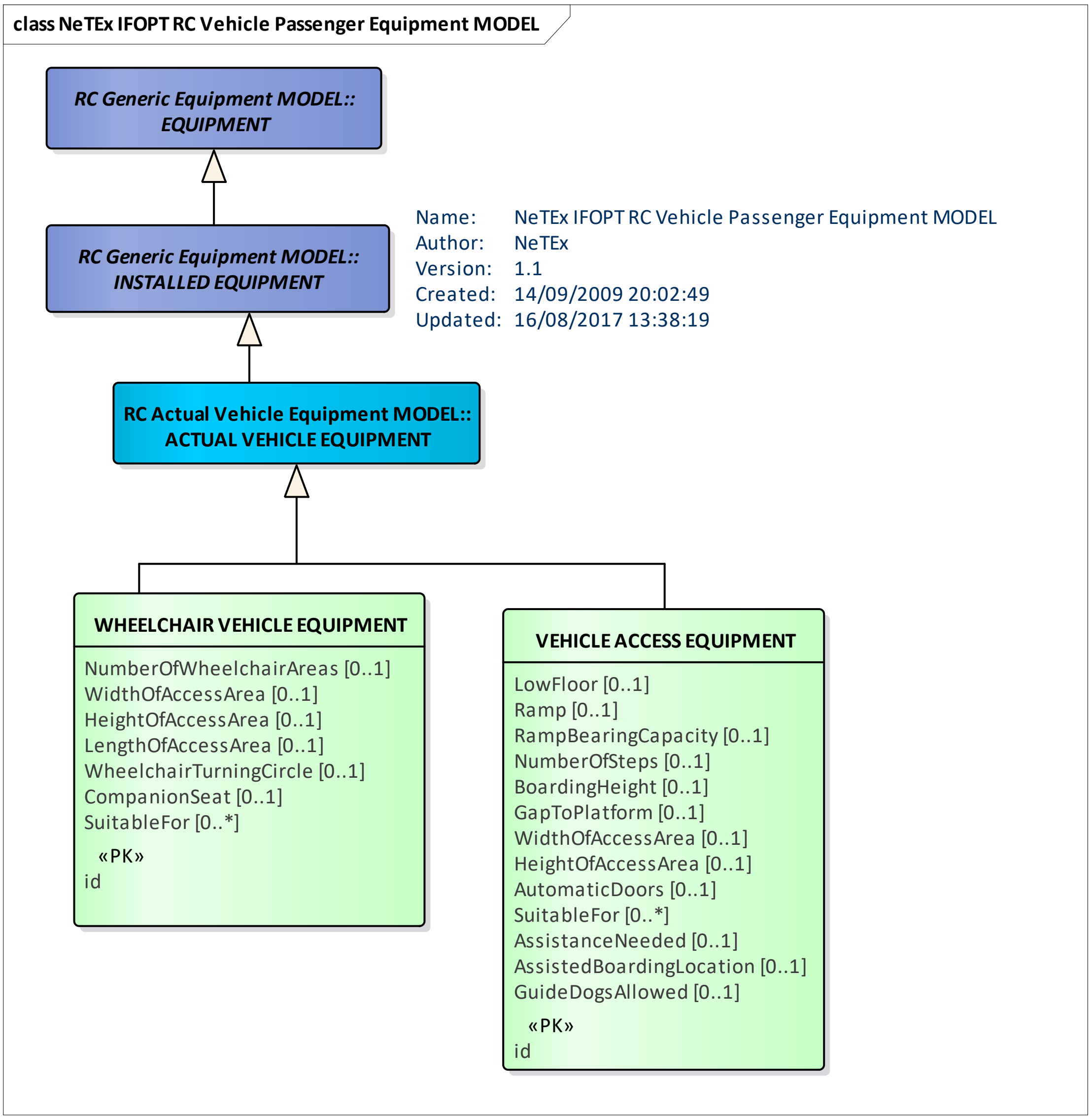


Figure 315 — Vehicle Passenger Equipment – Conceptual MODEL (UML)

7.7.13.2 Vehicle Passenger Equipment – Physical Model

The following figure shows the corresponding physical classes.

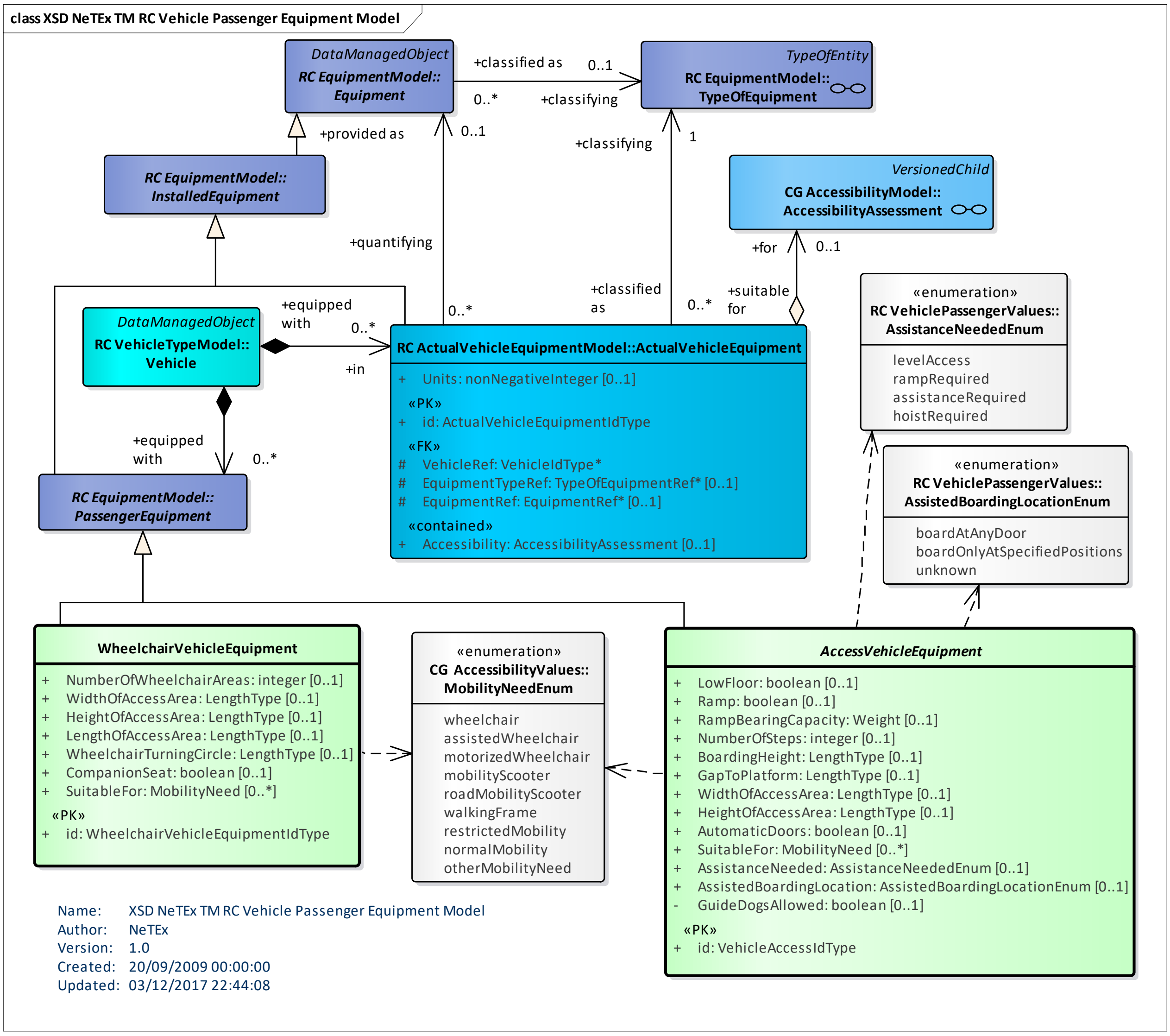


Figure 316 — Vehicle Passenger Equipment – Physical Model (UML)

7.7.13.3 Vehicle Passenger Equipment – Attributes and XSD

7.7.13.3.1 AccessVehicleEquipment – Model Element

Specialization of VEHICLE EQUIPMENT for ACCESS providing information such as low floor, ramp, access area dimensions, etc.

Table 245 — *VehicleAccessEquipment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*ActualVehicle-Equipment*</u>	::>	VEHICLE ACCESS EQUIPMENT inherits from ACTUAL VEHICLE EQUIPMENT.
«PK»	***id***	*VehicleAccess-IdType*	1:1	Identifier of ACCESS VEHICLE EQUIPMENT.
	LowFloor	*xsd:boolean*	0:1	Whether there is a low floor.

	Ramp	*xsd:boolean*	0:1	Whether there is a RAMP.
	RampBearingCapacity	*WeightType*	0:1	Bearing capacity of RAMP.
	NumberOfSteps	*xsd:integer*	0:1	Number of steps for access.
	BoardingHeight	*LengthType*	0:1	Boarding height.
	GapToPlatform	*LengthType*	0:1	Normal gap to platform at most stations.
	WidthOfAccessArea	*LengthType*	0:1	Width of Wheelchair access.
	HeightOfAccessArea	*LengthType*	0:1	Height of Wheelchair access.
	AutomaticDoors	*xsd:boolean*	0:1	Whether there are automatic doors.

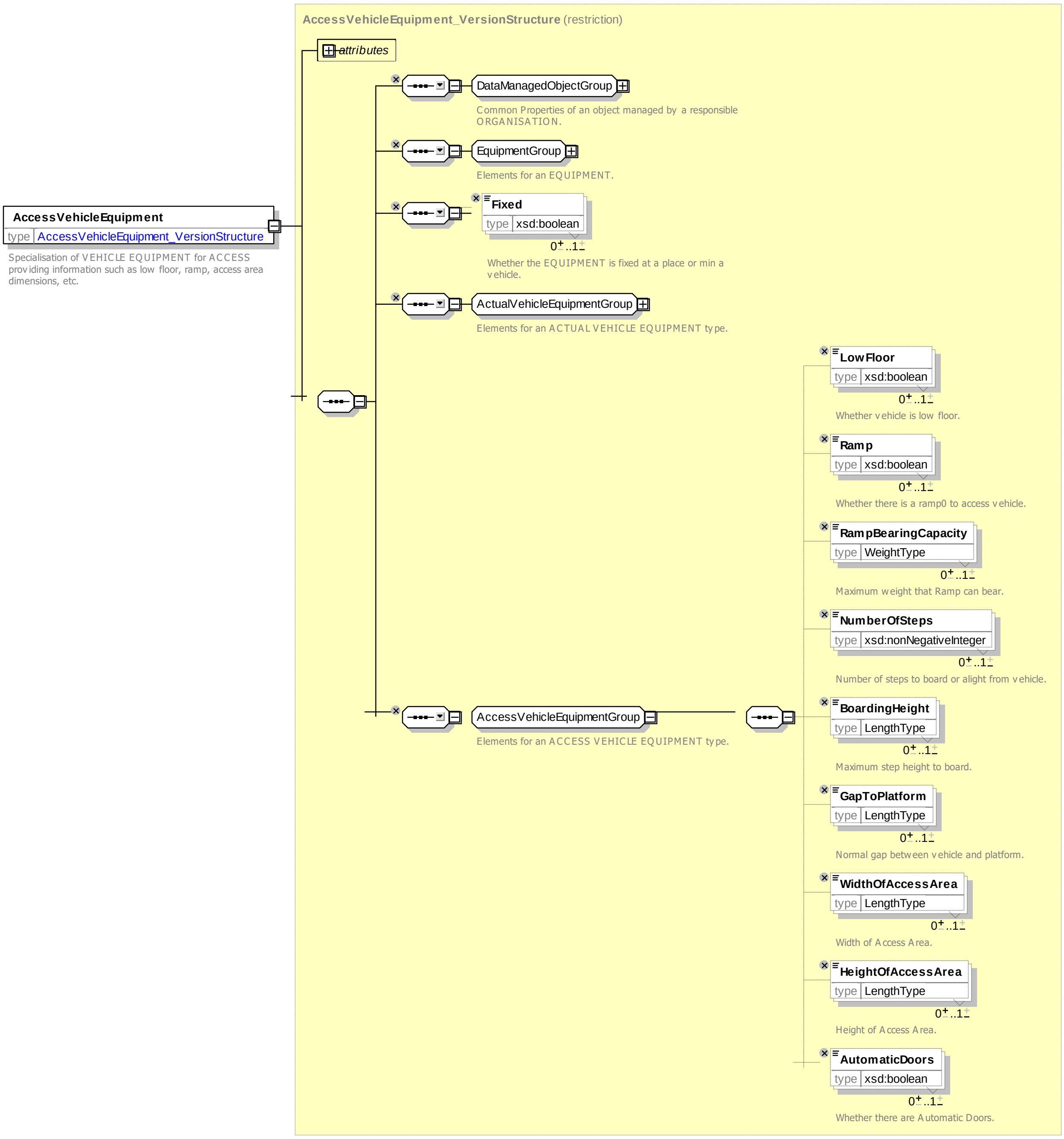


Figure 317 —*AccessVehicleEquipment – XSD*

7.7.13.3.2 WheelchairVehicleEquipment – Model Element

Specialization of VEHICLE EQUIPMENT for wheelchair accessibility on board a VEHICLE providing information such as the number of wheel chair areas and the access dimensions.

Table 246 — *VehicleAccessEquipment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*ActualVehicle-Equipment*	::>	WHEELCHAIR VEHICLE EQUIPMENT inherits from ACTUAL VEHICLE EQUIPMENT.

«PK»	***id***	*VehicleWheelchair-IdType*	1:1	Identifier of WHEELCHAIR VEHICLE EQUIPMENT.
	HasWheelchairSpaces	*xsd:boolean*	0:1	Whether there are wheelchair spaces.
	NumberOfWheelchairAreas	*xsd:integer*	0:1	Number of wheelchair spaces.
	WidthOfAccessArea	*LengthType*	0:1	Width of wheelchair space.
	LengthOfAccessArea	*LengthType*	0:1	Length of wheelchair space.
	HeightOfAccessArea	*LengthType*	0:1	Height of wheelchair space.
	WheelchairTurning-Circle	*LengthType*	0:1	Wheelchair turning circle in space (radius).
	CompanionSeat	*xsd:boolean*	0:1	Whether there is a companion seat.

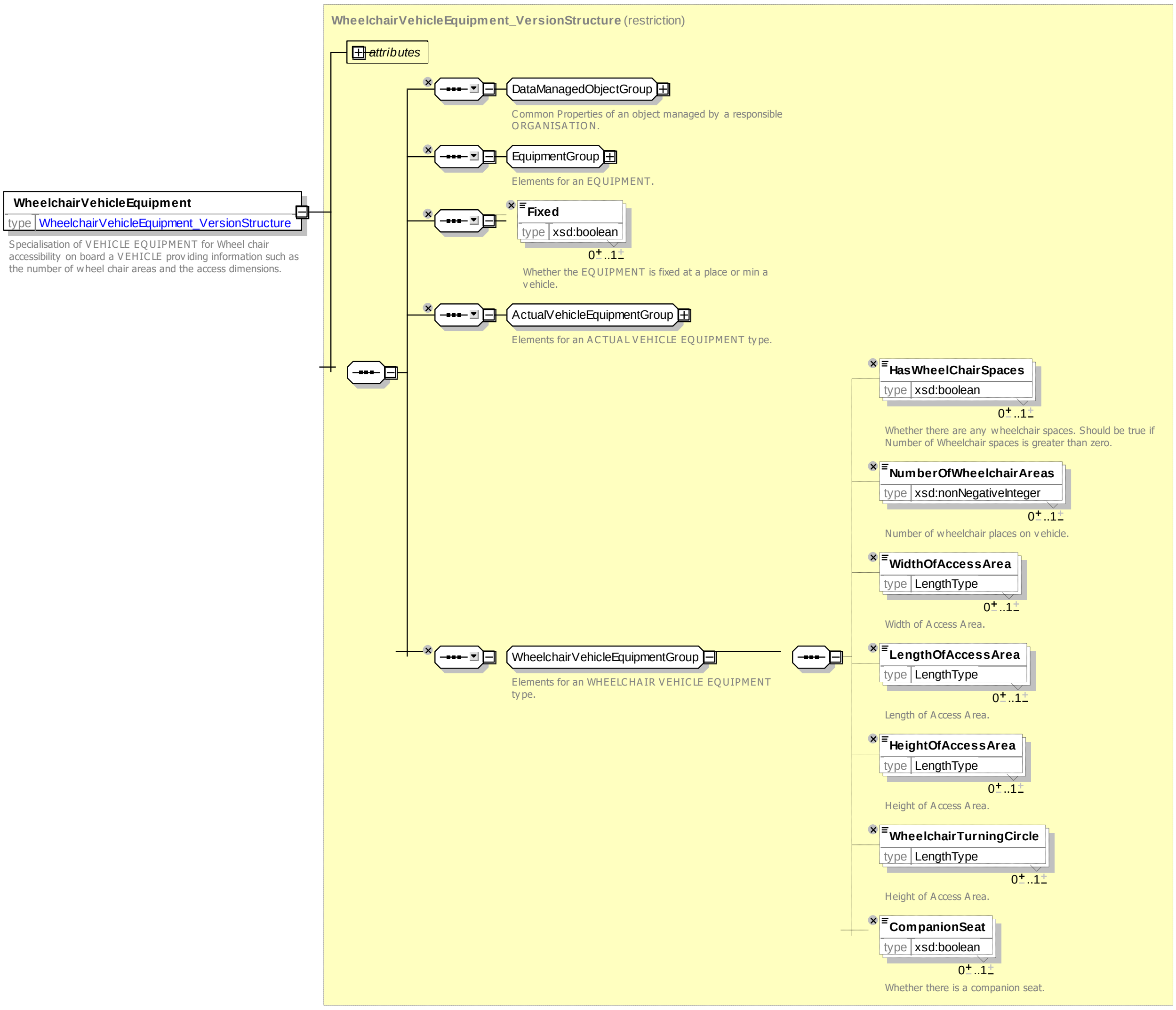


Figure 318 — *WheelchairVehicleEquipment – XSD*

7.7.14 Facility

7.7.14.1 FACILITY – Conceptual MODEL

The Facility Model provides named facilities that can be associated with SERVICE JOURNEYS, SITE ELEMENTs and other entities.

A FACILITY provides just a simple name of a capability. Detailed properties may be stated for some types of facilities by a corresponding EQUIPMENT type.

FACILITies are combined into FACILITY SETs – reusable standard combinations of facilities.

A SERVICE FACILITY SET describes a set of FACILITIES for use on a SERVICE. It can include information about the ACCOMMODATION on board. A SITE FACILITY SET describes a set of FACILITIES available at a SITE.

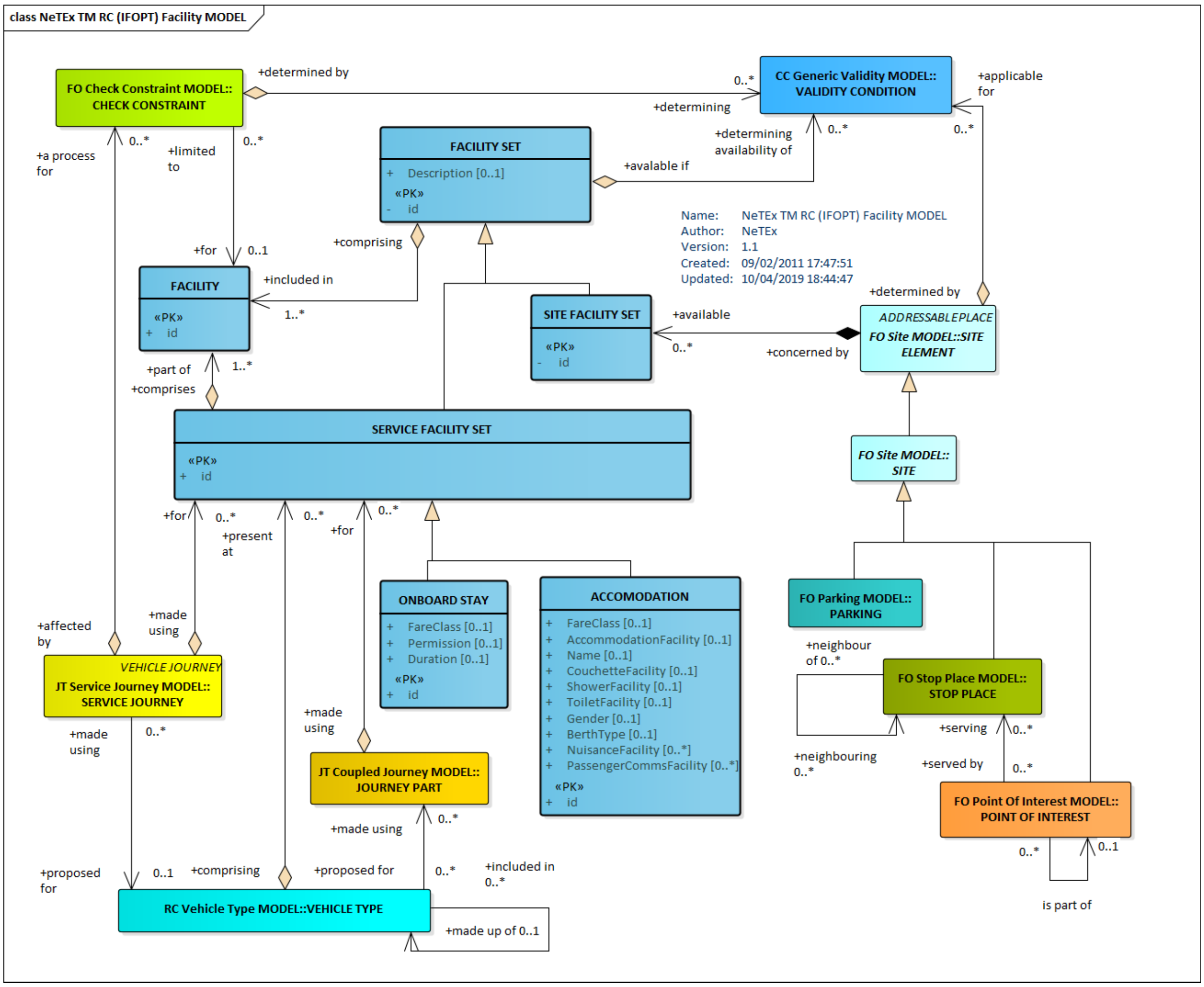


Figure 319 — Facility – Conceptual MODEL (UML)

7.7.14.2 Facility – Physical Model

NeTEx uses enumerations to define standardised sets of Facility values. The values are implemented as list of enumerated values.

7.7.14.2.1 Facility Set – Physical Model

The following figure shows the basic Physical model for FACILITY SETs. The allowed values for different facilities are shown in subsequent diagrams.

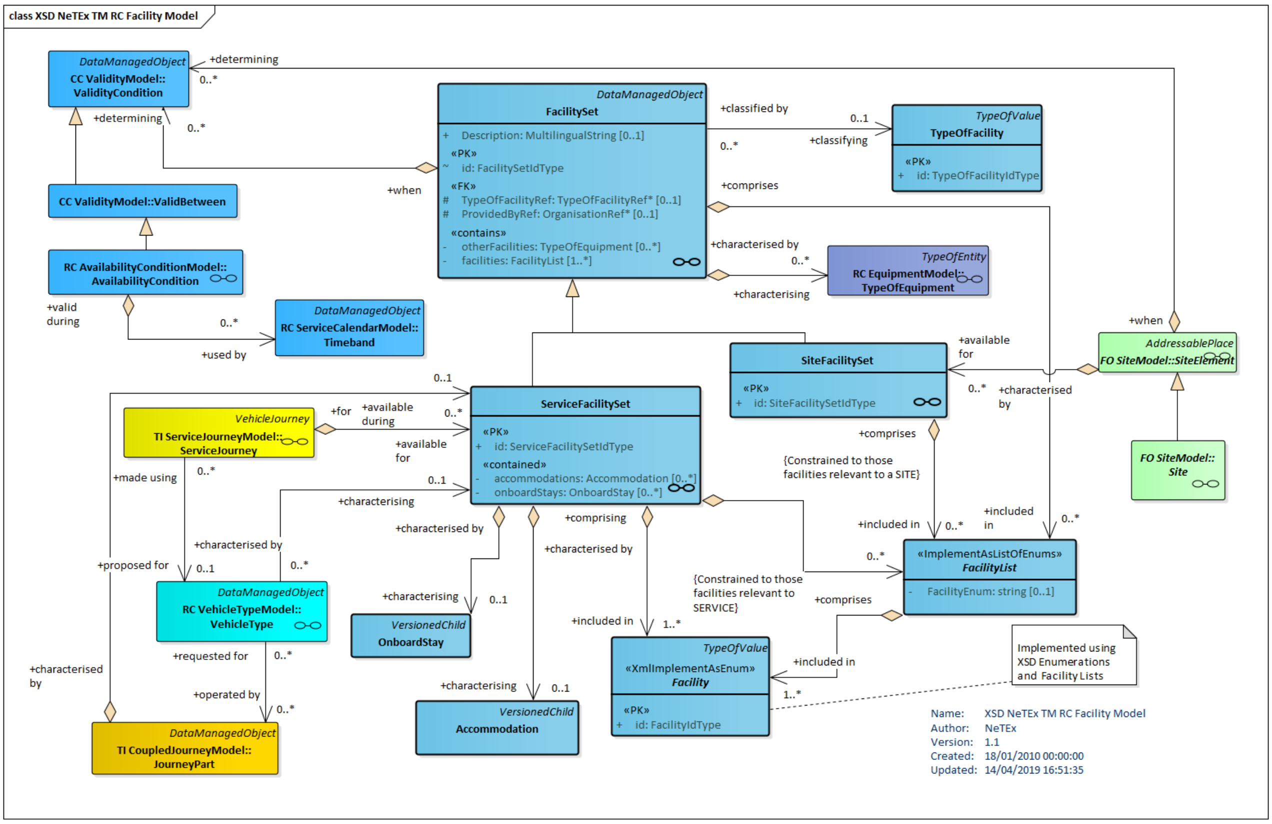


Figure 320 — Facility – Physical Model (UML)

7.7.14.2.2 Use of Facilities – Physical Model

The following diagram shows how FACILITies can be associated with SITE and JOURNEY entities.

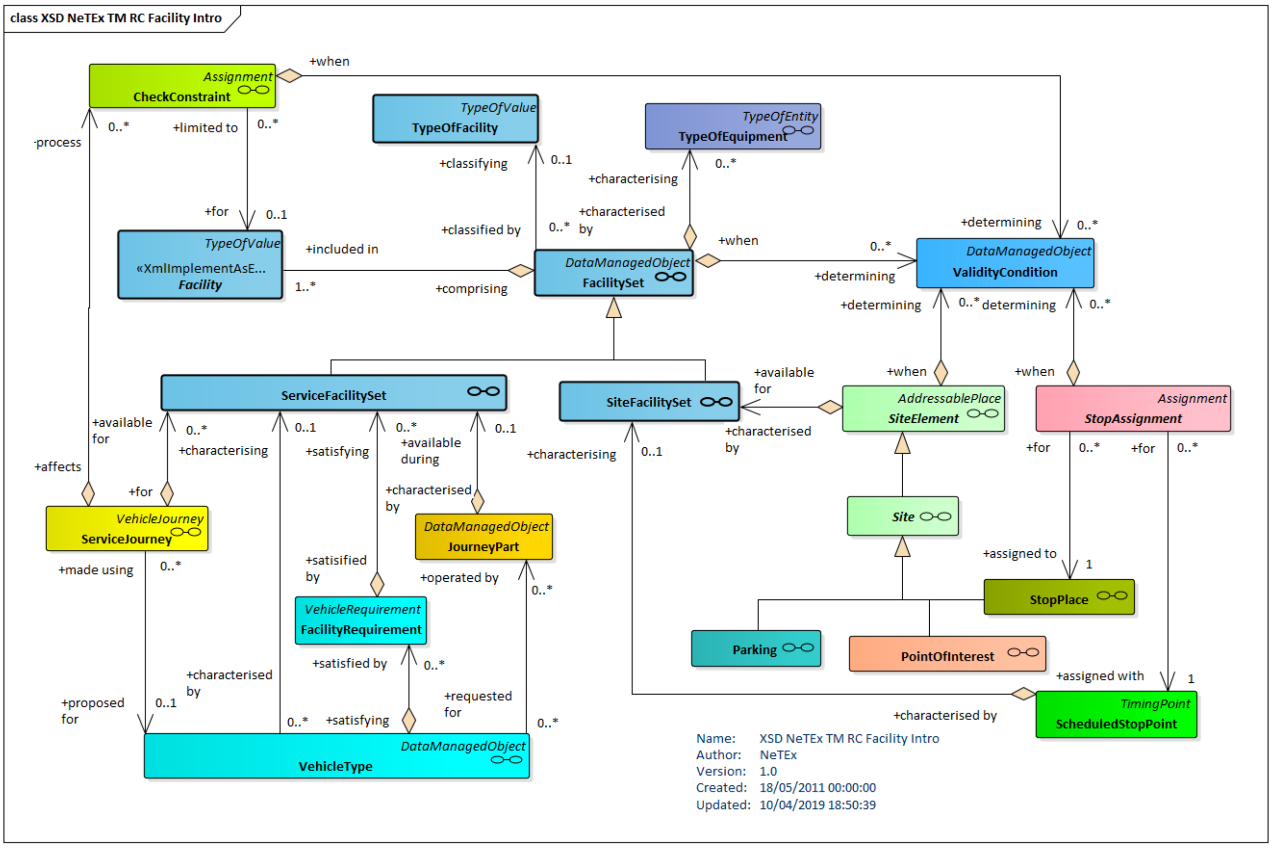


Figure 321 — Use of Facilities – Physical Model (UML)

7.7.14.2.3 Common Facility Values – Physical Model

The following diagram shows the common FACILITies that are allowed in a FACILITY SET for both SITEs and SERVICEs.

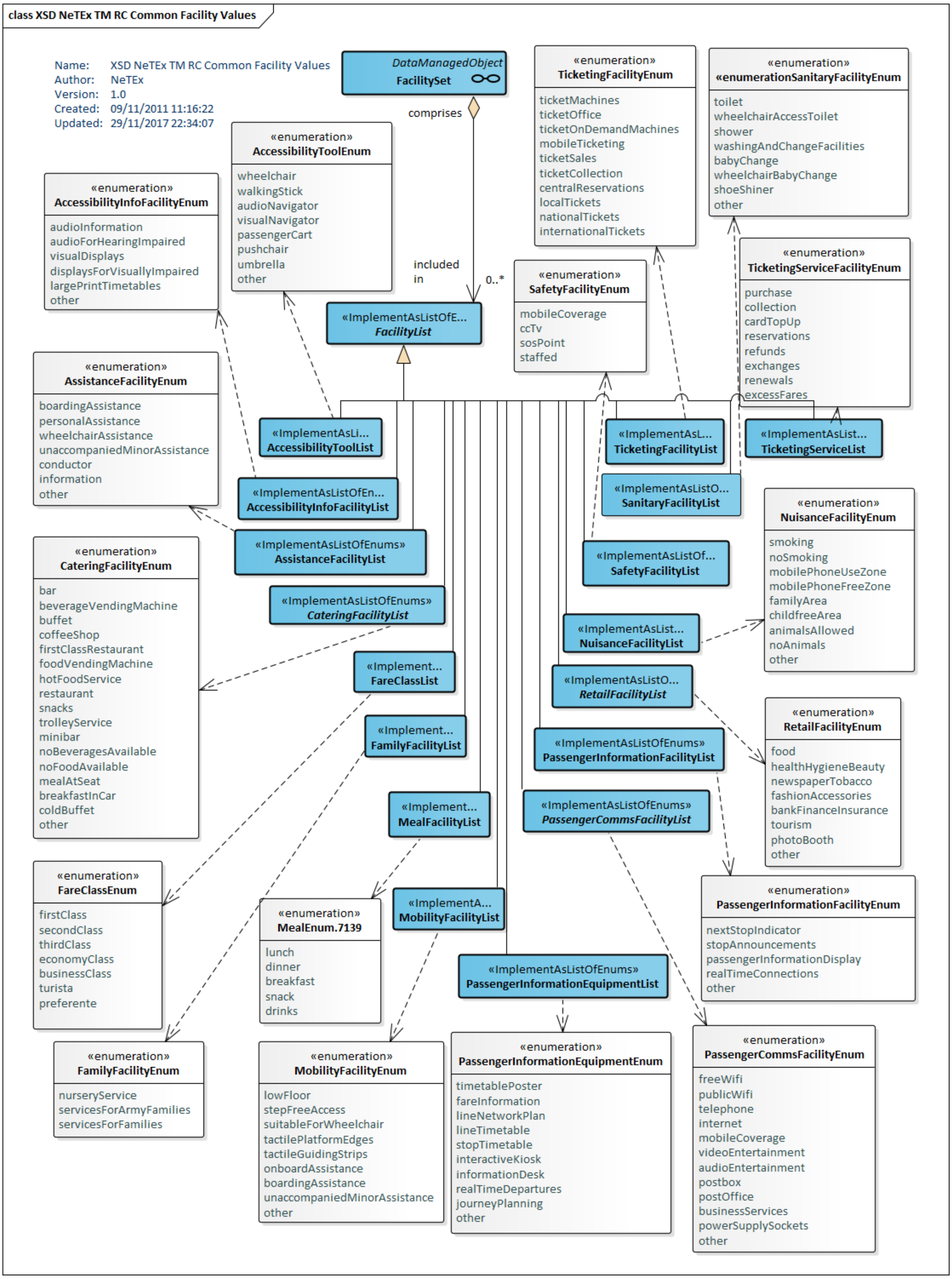


Figure 322 — Common Facility Values– Physical Model (UML)

7.7.14.2.4 Site Facility Values – Physical Model

The following diagram shows the Facilities that are allowed only in a SITE FACILITY SET

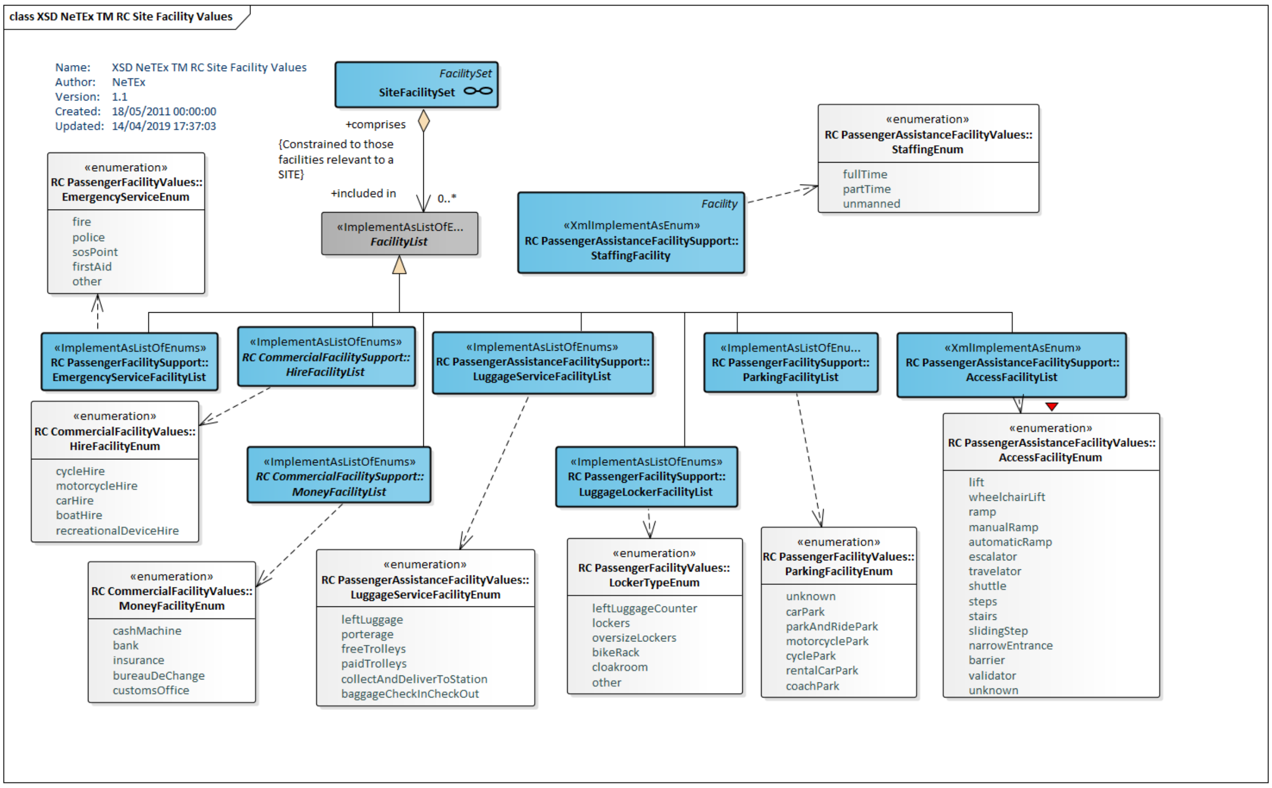


Figure 323 — Site Facility Values – Physical Model (UML)

7.7.14.2.5 Service Facility Values– Physical Model

The following diagram shows the Facilities that are allowed only in a SERVICE FACILITY SET (along with the common facilities).

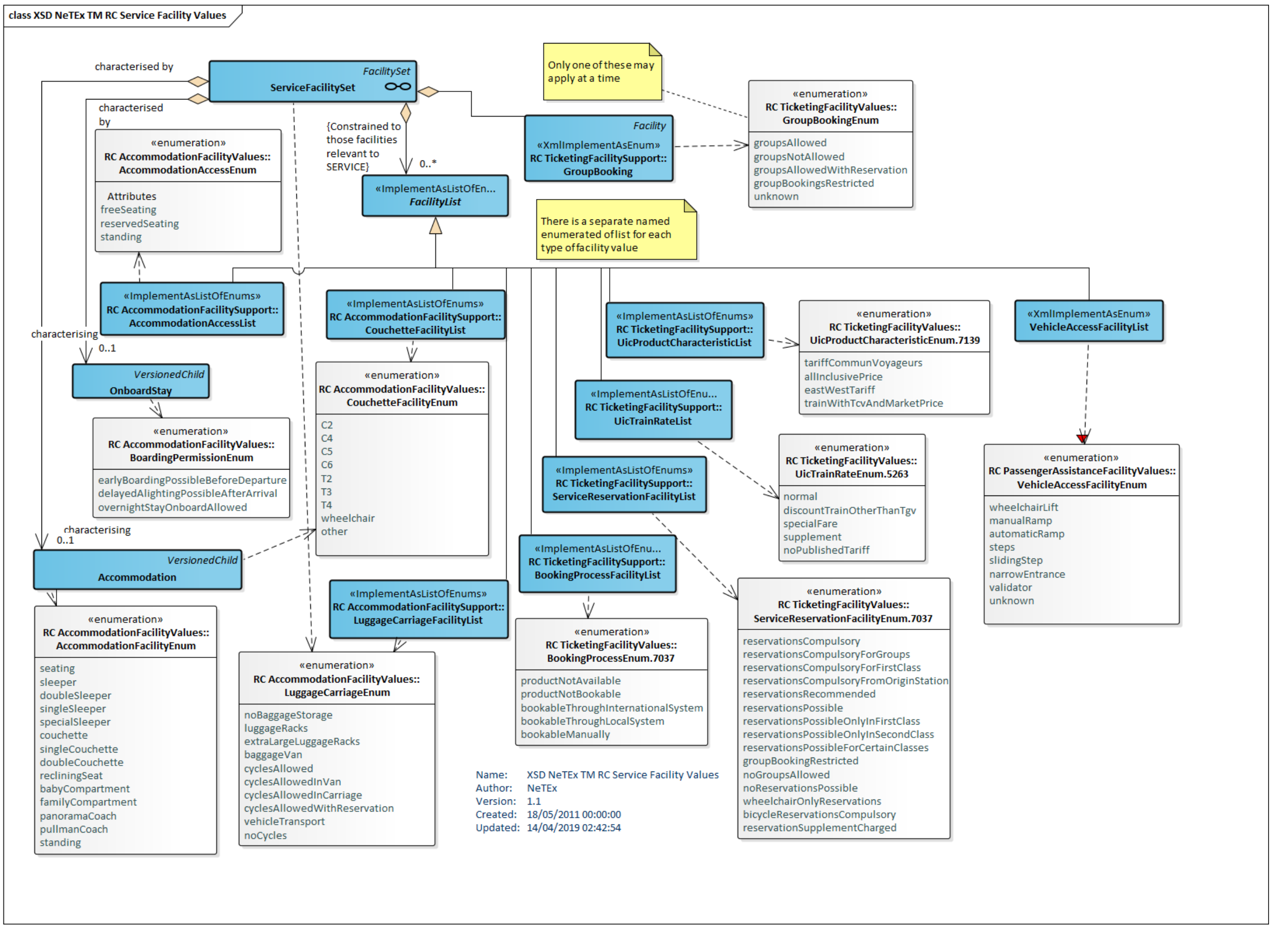


Figure 324 — Service Facility Values – Physical Model (UML)

7.7.14.2.6 Accommodation – Physical Model

A SERVICE FACILITY SET may in addition have ACCOMMODATION elements to describe the type of onboard accommodation available and an ON BOARD STAY to specify conditions for early and late boarding.

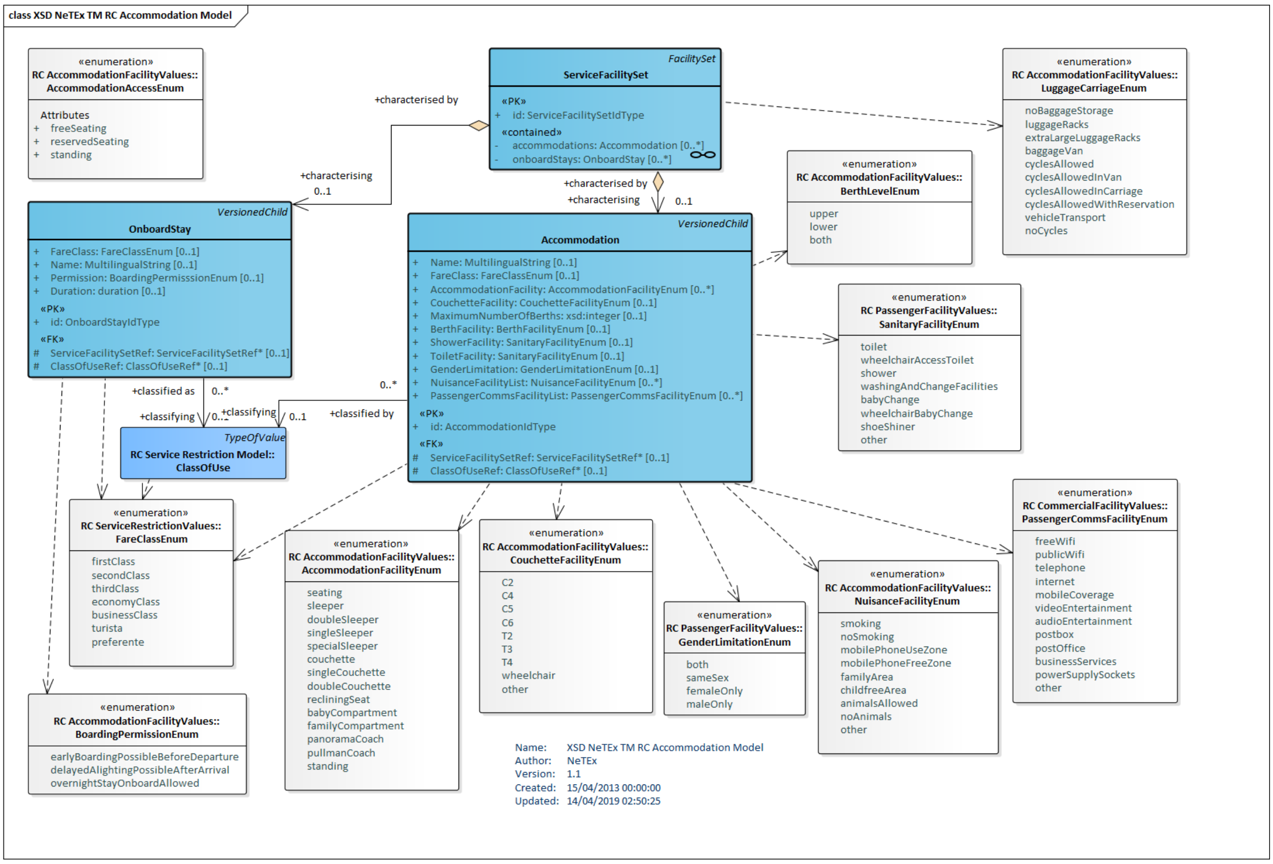


Figure 325 — On board Accommodation – Physical Model (UML)

7.7.14.3 Facility – Attributes and XSD

7.7.14.3.1 FacilitySet – Model Element

A set of FACILITIES that may be associated with an ENTITY and subject to a specific VALIDITY CONDITION. Values with a SET are logically ANDed together. For example, 'FareClass = firstClass' and 'CateringFacility = Restaurant'.

Table 247 — *FacilitySet – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	FACILITY SET inherits from DATA MANAGED OBJECT.
«PK»	***id***	*FacilitySetIdType*	1:1	Identifier of FACILITY SET.
«FK»	***ProvidedByRef***	*OrganisationRef*	1:0	ORGANISATION responsible for providing facilities in FACILITY SET.
	Description	*MultilingualString*	0:1	Description of FACILITY SET.
«FK»	***TypeOfFacilityRef***	*TypeOfFacilityRef*	0:1	Classification of FACILITY SET.
«cntd»	***otherFacilities***	*TypeOfEquipment* | *TypeOfEquipmentRef*	0:*	Arbitrary user defined FACILITY types in FACILITY SET, defined as TYPES OF EQUIPMENT.
XGRP	***CommonFacility-Group***	***xmlGroup***	0:*	FACILITIEs, defined as lists of enumerated values of fixed types. These are common to any FACILITY SET. There are additional facilities that are specific to the SERVICE FACILITY SET and SITE FACILITY SET.

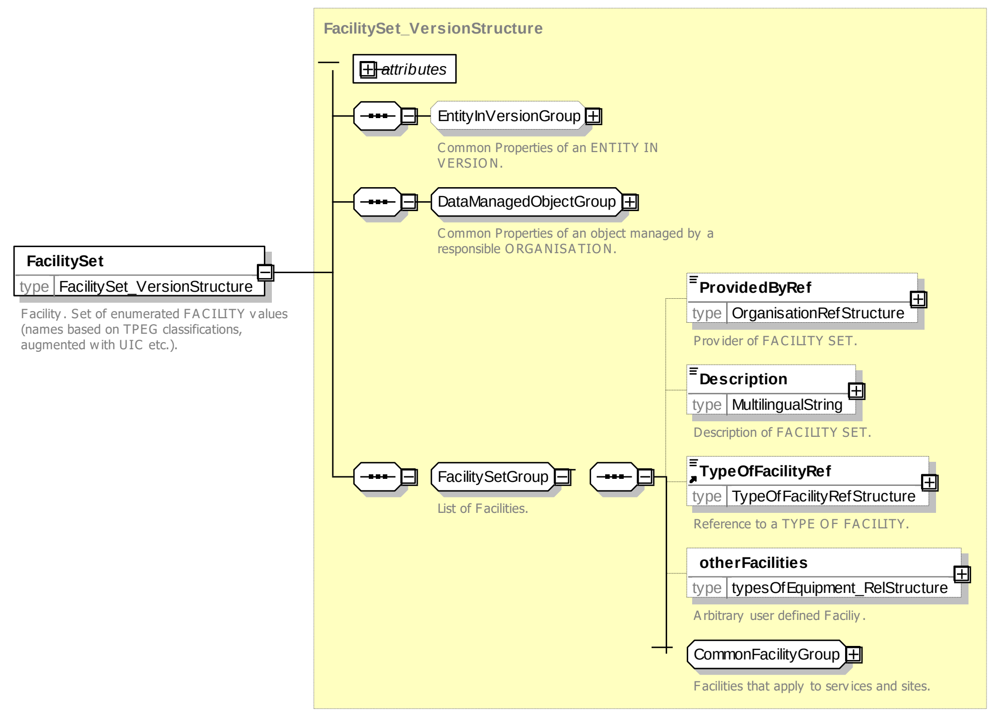


Figure 326 —*FacilitySet – XSD*

7.7.14.3.1.1 CommonFacilityGroup – XML Group

Different lists of named FACILITies that may be included in a FACILITY SET.

Table 248 — *CommonFacilitySet – Group*

Classification	Name	Type	Cardinality	Description
«enum»	***Accessibility-InfoFacilityList***	*AccessibilityInfoFacility-Enum*	0:*	ACCESSIBILITY INFO FACILITies. See allowed values below.
«enum»	***AssistanceFacility-List***	*AssistanceFacilityEnum*	0:*	ASSISTANCE FACILITies. See allowed values below.
«enum»	***AccessibilityToolList***	*AccessibilityToolEnum*	0:*	ACCESSIBILITY TOOLs. See allowed values below.
«enum»	***CarServiceFacility-List***	*CarServiceFacilityEnum*	0:*	CAR SERVICE FACILITies. See allowed values below.
«enum»	***CateringFacilityList***	*CateringFacilityEnum*	0:*	CATERING FACILITies. See allowed values below.
«enum»	***FamilyFacilityList***	*FamilyFacilityEnum*	0:*	FAMILY FACILITies. See allowed values below.
«enum»	***FareClasses***	*FareClasses*	0:1	FARE CLASS. See allowed values below.
«enum»	***GenderLimitation***	*GenderLimitation*	0:1	GENDER LIMITATION. See allowed values below.
«enum»	***MealFacilityList***	*MealFacilityEnum*	0:*	MEAL FACILITies. See allowed values below.
«enum»	***MedicalFacilityList***	*MedicalFacilityEnum*	0:*	MEDICAL FACILITies. See allowed values below.
«enum»	***MobilityFacilityList***	*MobilityFacilityEnum*	0:*	MOBILITY FACILITies. See allowed values below.
«enum»	***NuisanceFacilityList***	*NuisanceFacilityEnum*	0:*	NUISANCE FACILITies. See allowed values below.
«enum»	***PassengerComms-FacilityList***	*PassengerComms-FacilityEnum*	0:*	PASSENGER COMMS FACILITies. See allowed values below.
«enum»	***Passenger-Information-EquipmentList***	*PassengerInformation-EquipmentEnum*	0:*	PASSENGER INFORMATION EQUIPMENT See allowed values below.
«enum»	***Passenger-Information-FacilityList***	*PassengerInformation-FacilityEnum*	0:*	PASSENGER INFORMATION FACILITies. See allowed values below.
«enum»	***RetailFacilityList***	*RetailFacilityEnum*	0:*	RETAIL FACILITies. See allowed values below.

«enum»	***SafetyFacilityList***	*SafetyFacilityEnum*	0:*	SAFETY FACILITies. See allowed values below.
«enum»	***SanitaryFacilityList***	*SanitaryFacilityEnum*	0:*	SANITARY FACILITies. See allowed values below.
«enum»	***TicketingFacilityList***	*TicketingFacilityEnum*	0:*	TICKETING FACILITies. See allowed values below.
«enum»	***TicketingService-FacilityList***	*TicketingService-FacilityEnum*	0:*	TICKETING SERVICE FACILITies. See allowed values below.

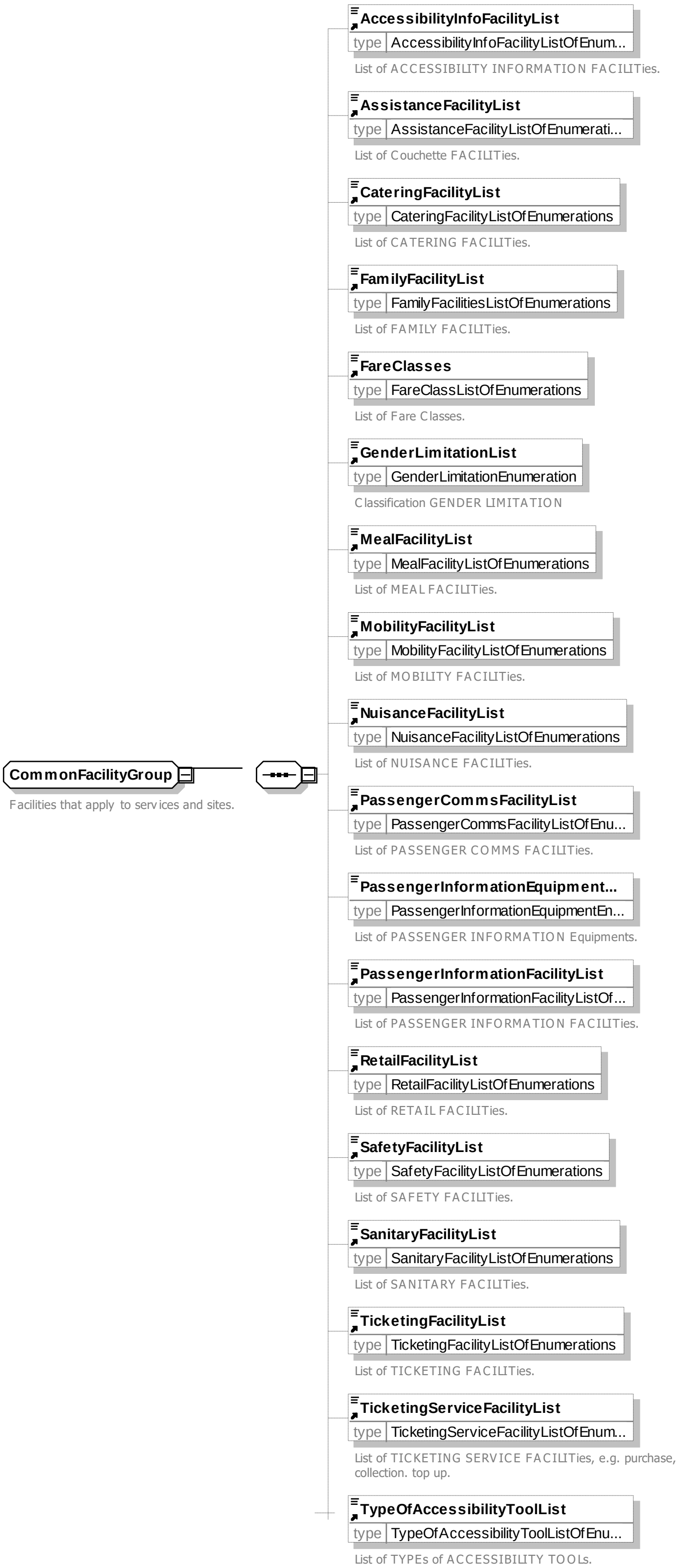


Figure 327 —*CommonFacilityGroup – XSD*

7.7.14.3.2 SiteFacilitySet – Model Element

Set of FACILITies available for a SITE or SITE ELEMENT.

Table 249 — *SiteFacilitySet – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*FacilitySet*	::>	SITE FACILITY SET inherits from FACILITY SET.
«PK»	***id***	*SiteFacilitySetIdType*	1:1	Identifier of SITE FACILITY SET.
XGRP	***SiteFacilityGroup***	***xmlGroup***	0:*	SITE FACILITies in SITE FACILITY SET defined as enumerated lists of fixed values. There are specific to the SITE FACILITY SET.

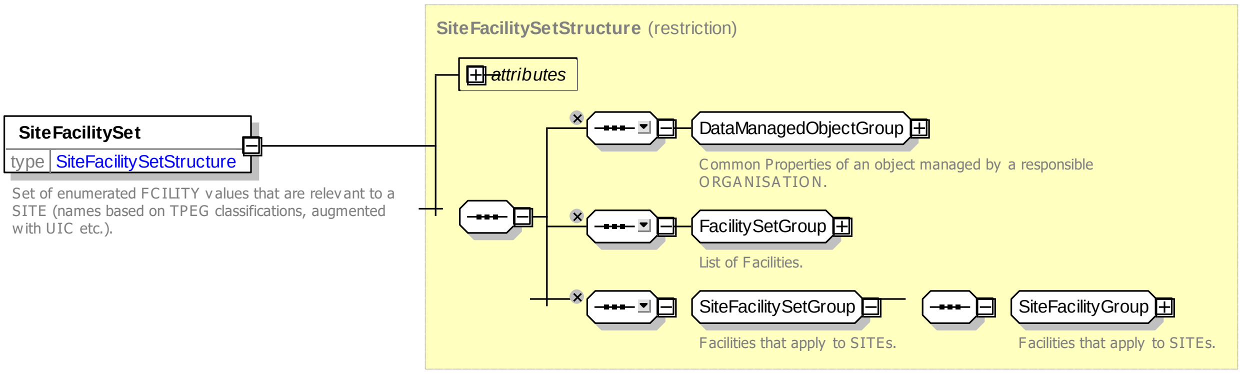


Figure 328 — *SiteFacilitySet – XSD*

7.7.14.3.2.1 SiteFacilityGroup – XML Group

The ***SiteFacilityGroup*** provides list of enumerations characterising facilities that apply to a SITE.

Table 250 — *SiteFacilitySet – Group*

Classification	Name	Type	Cardinality	Description
«enum»	***AccessFacilityList***	*AccessFacilityEnum*	0:*	ACCESS FACILITY. See allowed values below. +v1.1
«enum»	***EmergencyService-List***	*EmergencyServiceEnum*	0:*	EMERGENCY SERVICE. See allowed values below.
«enum»	***HireFacilityList***	*HireFacilityEnum*	0:*	HIRE FACILITY. See allowed values below.
«enum»	***LuggageLocker-FacilityList***	*LuggageLockerFacility-Enum*	0:*	LUGGAGE LOCKER FACILITY.See allowed values below.
«enum»	***LuggageService-FacilityList***	*LuggageServiceFacility-Enum*	0:*	LUGGAGE SERVICE FACILITY.See allowed values below.
«enum»	***MoneyFacilityList***	*MoneyFacilityEnum*	0:*	MONEY FACILITY. See allowed values below.
«enum»	***ParkingFacilityList***	*ParkingFacilityEnum*	0:*	PARKING FACILITY. See allowed values below.

«enum»	***Staffing***	*StaffingEnum*	0:1	STAFFING. See allowed values below.
«enum»	***ServiceReservation-FacilityList***	*ServiceReservation-FacilityEnum*	0:*	SERVICE RESERVATION FACILITY. See allowed values below.
«enum»	***UicProduct-CharacteristicList***	*UicProduct-CharacteristicEnum*	0:*	UIC PRODUCT CHARACTERISTIC. See allowed values below.

Figure 329 —*SiteFacilityGroup – XSD*

7.7.14.3.3 ServiceFacilitySet – Model Element

Set of FACILITies available for a SERVICE JOURNEY or a JOURNEY PART. The set may be available only for a specific VEHICLE TYPE within the SERVICE (e.g. carriage equipped with low floor)

Table 251 — *ServiceFacilitySet – Element*

Classifi cation	Name	Type	Cardin ality	Description
::>	::>	*FacilitySet*	::>	SERVICE FACILITY SET inherits from FACILITY SET.
«PK»	***id***	*ServiceFacilitySetIdType*	1:1	Identifier of SERVICE FACILITY SET.
XGRP	***ServiceFacility-Group***	***xmlGroup***	0:1	Facilities specific to SERVCIE FACILITY SET. See below.
«cntd»	***accommodations***	*Accommodation* | *AccommodationRef*	0*	ACCOMMODATIONs in SERVICE FACILITY SET. See below.
«cntd»	***onboardStays***	*OnboardStay*	0:*	ONBOARD STAYs in SERVICE FACILITY SET. See below.

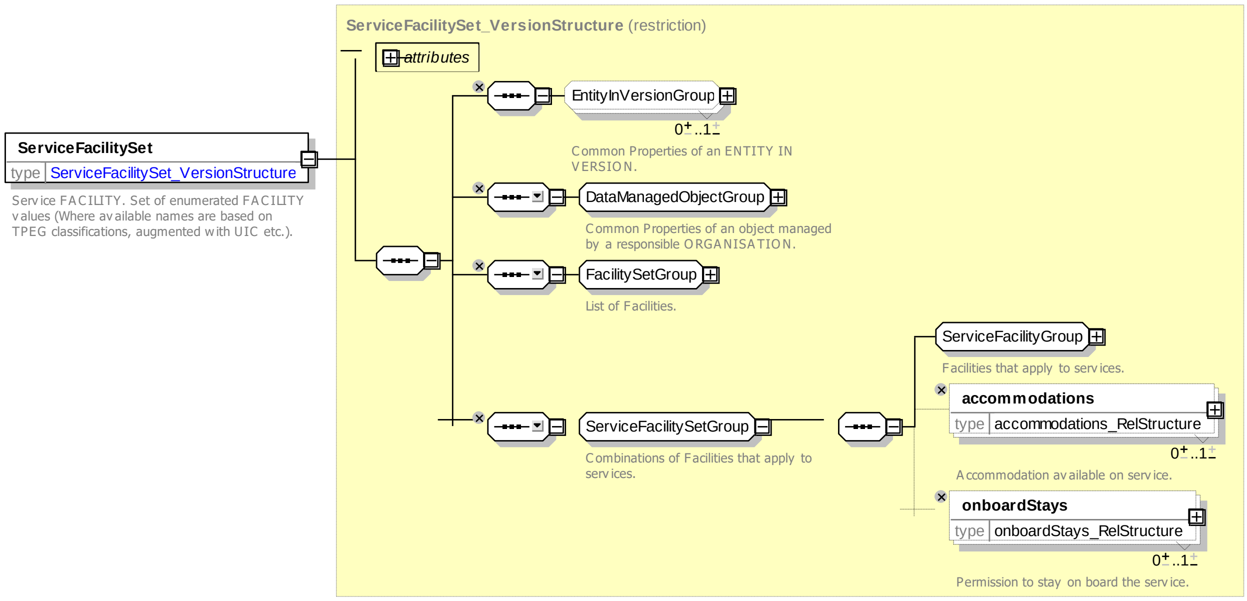


Figure 330 — *ServiceFacilitySet – XSD*

7.7.14.3.3.1 ServiceFacilityGroup – XML Group

Different lists of named FACILITIes that may be included in the SERVICE FACILITY SET.

Table 252 — *ServiceFacilitySet – Group*

Classification	Name	Type	Cardinality	Description
«enum»	***VehicleAccess-FacilityList***	*VehicleAccessFacility-Enum*	0:*	VEHICLE ACCESS FACILITY. See allowed values below.
«enum»	***Accommodation-AccessList***	*Accommodation-AccessEnum*	0:*	ACCOMMODATION ACCESS. See allowed values below.
«enum»	***Accommodation-FacilityList***	*AccommodationFacility-Enum*	0:*	ACCOMMODATION FACILITY. See allowed values below.
«enum»	***BoardingPermission***	*BoardingPermissionEnum*	0:1	BOARDING PERMISSION. See allowed values below.
«enum»	***BookingProcess-FacilityList***	*BookingProcessFacilityEnum*	0:*	BOOKING PROCESS FACILITY. See allowed values below.
«enum»	***CouchetteFacilityList***	*CouchetteFacilityEnum*	0:*	COUCHETTE FACILITY. See allowed values below.
«enum»	***GroupBooking-Facility***	*GroupBooking-FacilityEnum*	0:1	GROUP BOOKING FACILITY. See allowed values below.
«enum»	***LuggageCarriage-FacilityList***	*LuggageCarriage-FacilityEnum*	0:*	LUGGAGE CARRIAGE FACILITY. See allowed values below.
«enum»	***ServiceReservation-FacilityList***	*ServiceReservation-FacilityEnum*	0:*	SERVICE RESERVATION FACILITY. See allowed values below.

«enum»	***UicProduct-CharacteristicList***	*UicProduct-CharacteristicEnum*	0:1	UIC PRODUCT CHARACTERISTIC. See allowed values below.

Figure 331 —*ServiceFacilityGroup – XSD*

7.7.14.3.4 Accommodation – Model Element

A combination of accommodation characteristics available on a service, e.g. "*First Class Couchette with Shower and 2 bunks"*

Table 253 — *Accommodation – Element*

Classific ation	Name	Type	Cardin ality	Description
::>	::>	<u>*VersionedChild*</u>	::>	ACCOMMODATION inherits from VERSIONED CHILD.
«PK»	***id***	*AccomodationIdType*	1:1	Identifier of ACCOMMODATION.
	Name	*AccomodationIdType*	0:1	Name of ACCOMMODATION.

«FK»	***ServiceFacilitySet Ref***	*ServiceFacilitySetRef*	0:1	Parent SERVICE FACILITY SET for ACCOMMODATION.. +v1.1
«enum»	***FareClass***	*FareClassEnum*	0:1	FARE CLASS of ACCOMMODATION. See Service Restrictions model for allowed values.
«FK»	***ClassOfUseRef***	*ClassOfUseRef*	0:1	CLASS OF USE of ACCOMMODATION.. +v1.1
«enum»	***Accommodation-Facility***	*AccommodationFacility Enum*	0:1	Type of accommodation Facility in ACCOMMODATION. See allowed values below.
«enum»	***CouchetteFacility***	*CouchetteFacilityEnum*	0:1	Couchette Facility in ACCOMMODATION. See allowed values below.
	MaximumNumber OfBerths	*xsd:integer*	0:1	Maximum number of berths in ACCOMMODATION.
«enum»	***BerthFacility***	*BerthFacilityEnum*	0:1	Berth Facility in ACCOMMODATION. See allowed values below.
«enum»	***ShowerFacility***	*SanitaryFacilityEnum*	0:1	Shower Facility in ACCOMMODATION. See allowed values below.
«enum»	***ToiletFacility***	*SanitaryFacilityEnum*	0:1	Toilet Facility in ACCOMMODATION. See allowed values below.
«enum»	***GenderLimitation***	*GenderLimitationEnum*	0:1	Gender Limitation for ACCOMMODATION. E.g. Women only. See facility values.
«enum»	***NuisanceFacility***	*NuisanceFacilityEnum*	0:1	Nuisacne Facility in ACCOMMODATION. See allowed values below.

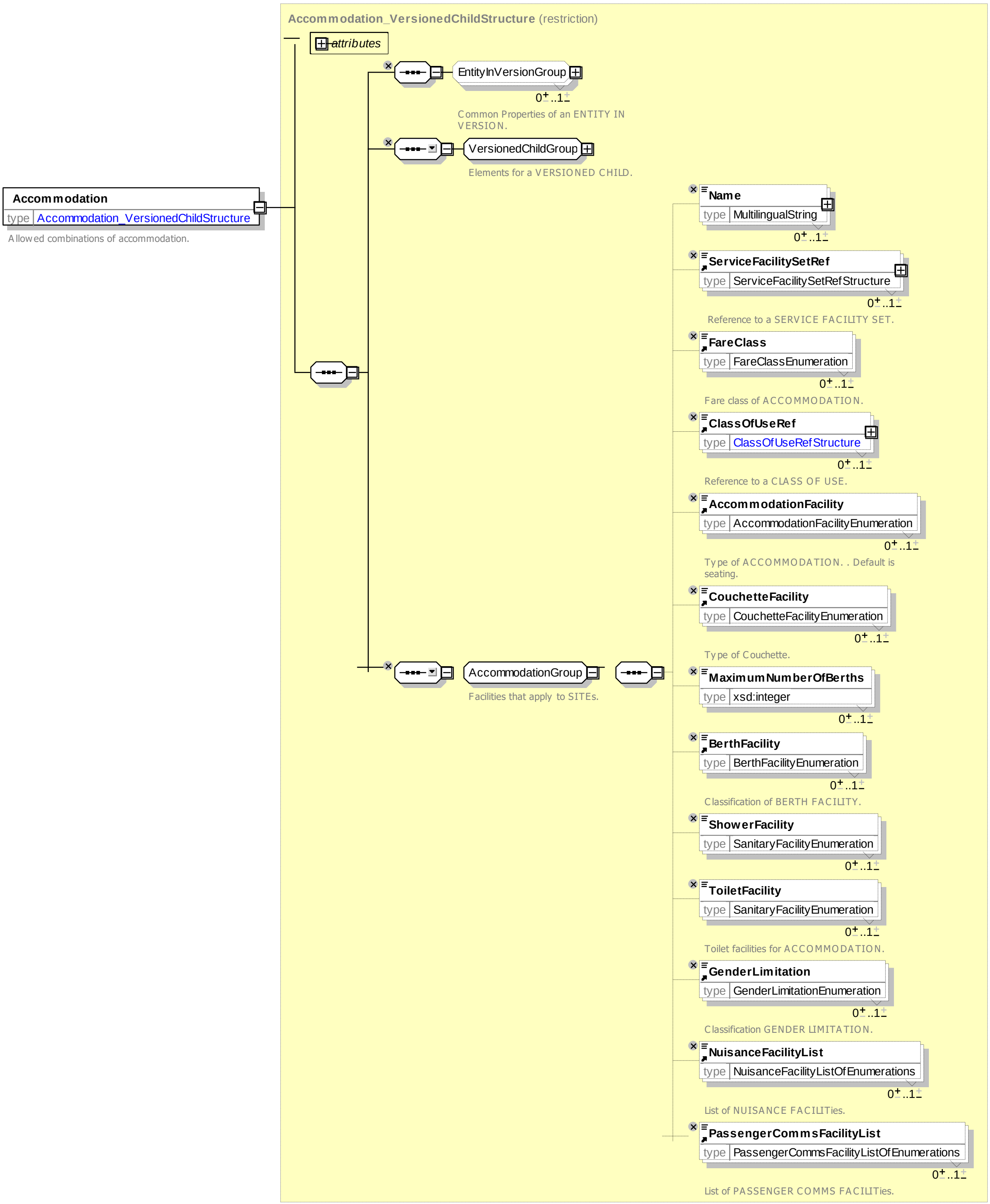


Figure 332 — *Accommodation – XSD*

7.7.14.3.5 OnboardStay – Model Element

Permission to board early before the journey or stay on board after the journey.

Table 254 — *OnboardStay – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	ON BOARD STAY inherits from VERSIONED CHILD.
«PK»	***id***	*OnBoardStayIdType*	1:1	Identifier of ON BOARD STAY. +v1.1
	Name	*MultilingualString*	0:1	Name of ON BOARD STAY. + v1.1.
«FK»	***ServiceFacility-SetRef***	*ServiceFacilitySetRef*	0:1	Parent SERVICE FACILITY SET for ACCOMMODATION.. +v1.1
«enum»	***FareClass***	*FareClassEnum*	0:1	FARE CLASS to which BOARDING PERMISSION applies.
«FK»	***ClassOfUseRef***	*ClassOfUseRef*	0:1	CLASS OF USE of ON BOARD STAY.. +v1.1
«enum»	***Permission***	*BoardingPermission-Enum*	0:1	Nature of BOARDING PERMISSION.
	Duration	*xsd:duration*	0:1	Duration of BOARDING PERMISSION.

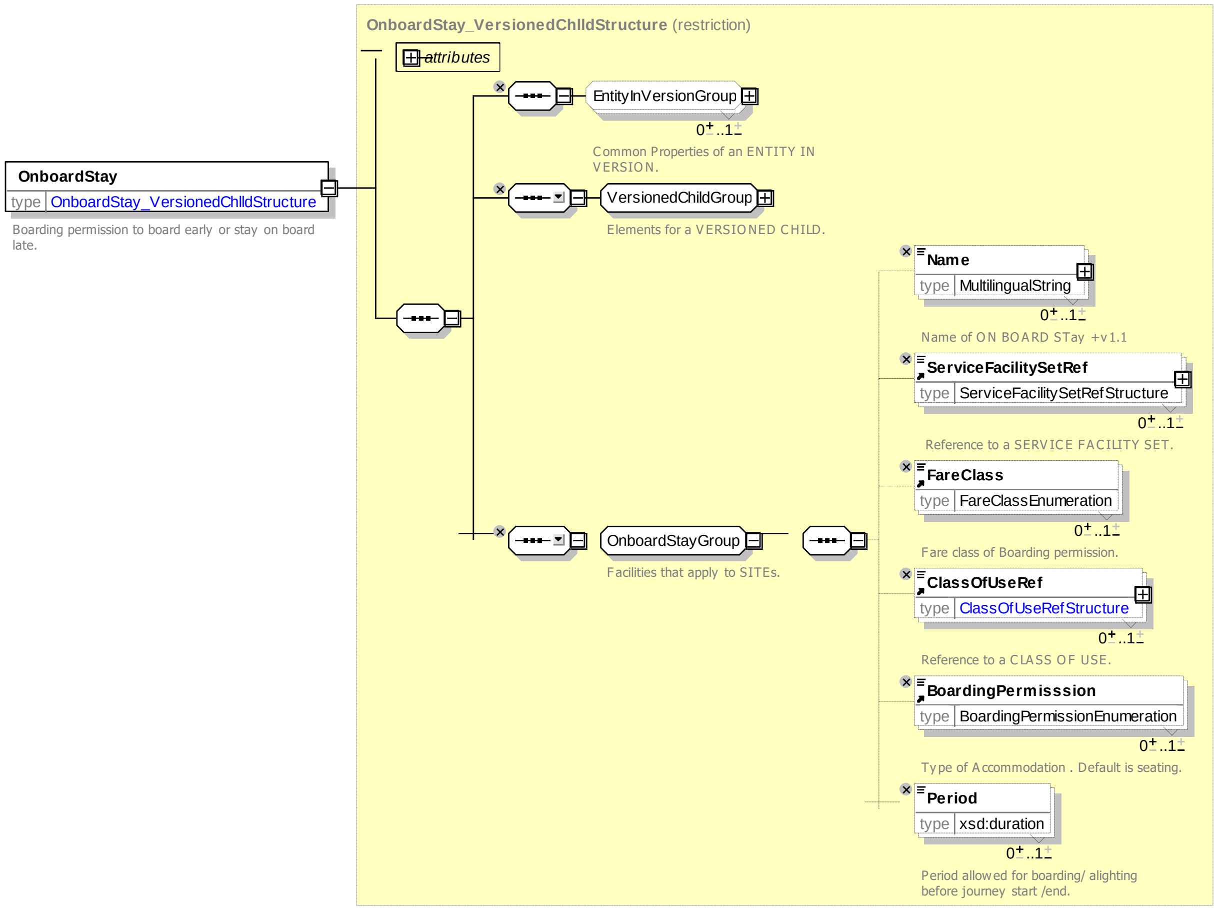


Figure 333 — *OnboardStay – XSD*

7.7.14.3.5.1 BoardingPermissionType – Allowed Values

Allowed values for Boarding Permission (*BoardingPermissionEnum*).

Table 255 — *BoardingPermissionType – AllowedValues*

Value	Description
earlyBoardingPossibleBeforeDeparture	Early boarding possible before departure.
delayedAlightingPossibleAfterArrival	Delayed alighting possible.
overnightStayOnboardAllowed	Overnight stay allowed.

7.7.14.3.6 FacilityList – Model Element

List of FACILITies: there are different lists of enumerations for each type of facilit.y.

Implemented by a List of Enumerations.

7.7.14.3.7 Facility – Model Element

A named amenity available to the public at a SITE or on a SERVICE. A facility has no further properties other than a name. An EQUIPMENT or LOCAL SERVICE is used to describe the further properties provided as part of particular facility.

Table 256 — *Facility – Element*

Classification	Name	Type	Cardinality	Description
«PK»	***id***	*FacilityIdType*	1:1	Identifier for FACILITY.

Facilities are normally implemented by enumerations.

7.7.14.4 Accommodation Facility Support Values

For each of the ACCOMMODATION FACILITies there is an enumeration that specifies the allowed values and a list type that allows the selection of multiple instances of the value.

Table 257 — *Accommodation Facilities – Value Lists*

List Of Values	TypeOfLocalService	Description
AccommodationFacilityList	***AccommodationFacility***	Type of Accommodation Facility.
AccommodationAccessList	***AccommodationAccess***	Type of Accommodation Access.
n/a	***BoardingPermission***	Type of Boarding Permission.
CouchetteFacilityList	***CouchetteeFacility***	Type of Couchette.
LuggageCarriageFacilityList	***LuggageCarriageFacility***	Type of Luggage Carriage service.
NuisanceFacilityList	***TypeOfNuisancee***	Type of Nuisance.

7.7.14.4.1 Accommodation Facilities – Physical Model

The following figure shows the allowed values for the various ACCOMMODATION FACILITY lists.

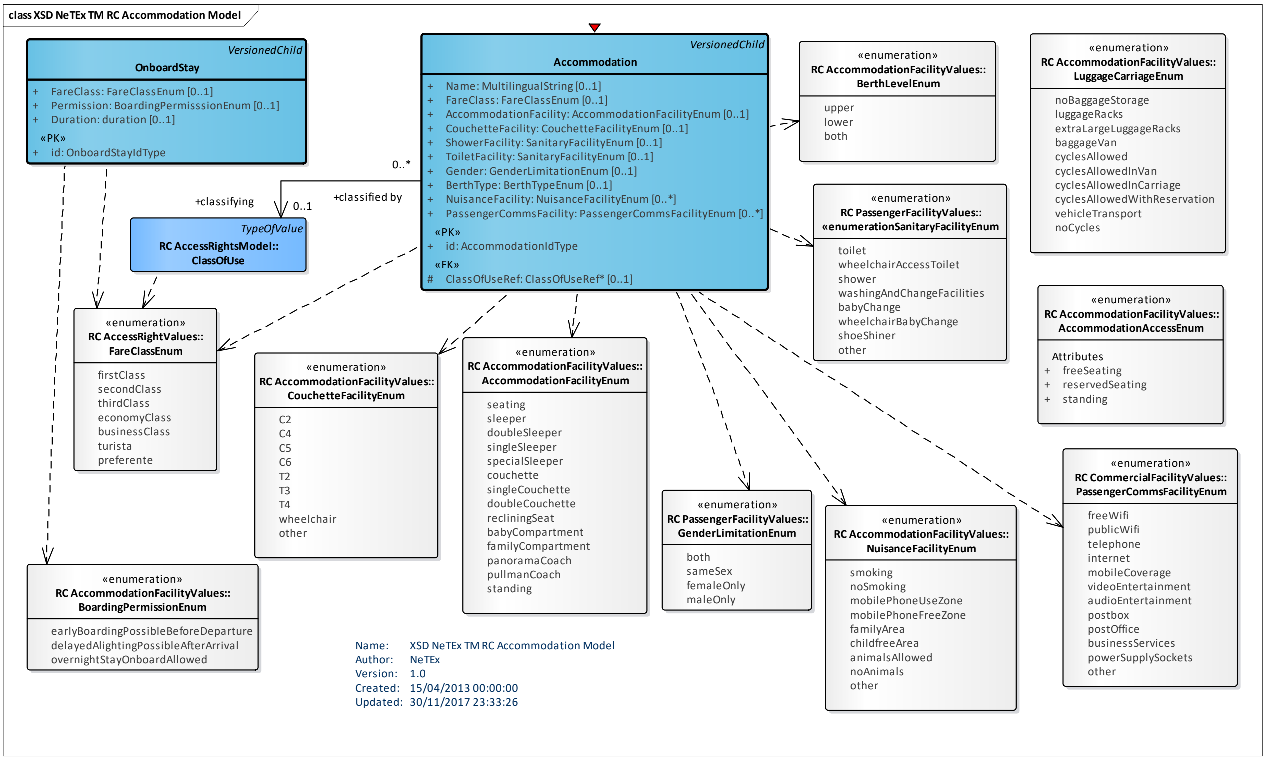


Figure 334 —*Accommodation Facility Values – Physical Model (UML)*

7.7.14.4.2 Accommodation Facilities – XSD Values

7.7.14.4.2.1 AccommodationAccessType – Allowed Values

Allowed values for type of ACCOMMODATION ACCESS FACILITY (*AccommodationAccessEnum*).

Table 258 — *AccommodationAccessType – AllowedValues*

Value	Description
freeSeating	Free seating.
reservedSeating	Reserved seating.
standing	Standing.

7.7.14.4.2.2 AccommodationFacilityType – Allowed Values

Allowed values for type of ACCOMMODATION FACILITY (*AccommodationFacilityEnumeration*).

Table 259 — AccommodationFacilityType – *AllowedValues*

Value	Description
seating	Seating.
sleeper	Sleeper accommodation.
doubleSleeper	Double Sleeper accommodation.
singleSleeper	Single Sleeper accommodation.
specialSleeper	Special Sleeper accommodation.

couchette	Couchette.
recliningSeat	Reclining seat.
babyCompartment	Baby Compartment.
familyCompartment	Family Compartment.
panoramaCoach	Panorama coach.
pullmanCoach	Pullman coach.
standing	Standing accommodation.

7.7.14.4.2.3 **CouchetteFacilityType – Allowed Values**

Allowed values for COUCHETTE FACILITY (*CouchetteFacilityEnumeration*).

Table 260 — CouchetteFacilityType – *AllowedValues*

Value	Description
C2	Couchette for 2.
C4	Couchette for 4.
C5	Couchette for 5.
C6	Couchette for 6.
T2	T2 Couchette.
T3	T3 Couchette.
T4	T4 Couchette.
wheelchair	Wheelchair couchette.
other	Other type of couchette.

7.7.14.4.2.4 **LuggageCarriageType – Allowed Values**

Allowed values for type of LUGGAGE CARRIAGE FACILITY (*LuggageCarriageEnumeration)*.

Table 261 — LuggageCarriageType – *AllowedValues*

Value	Description
noBaggageStorage	No baggage storage.
luggageRacks	Luggage racks.
extraLargeLuggageRacks	Extra large luggage storage.
baggageVan	Baggage van.
cyclesAllowed	Cycles allowed.
cyclesAllowedInVan	Cycles allowed in van.
cyclesAllowedInCarriage	Cycles allowed in carriage.
cyclesAllowedWith-Reservation	Cycles allowed with reservation.
noCycles	No cycles allowed.

7.7.14.4.2.5 **NuisanceFacilityType – Allowed Values**

Allowed values for type of NUISANCE FACILITY (*NuisanceFacilityEnumeration*).

Table 262 —NuisanceFacilityType – *AllowedValues*

Value	Description
smoking	Smoking zone.
noSmoking	No Smoking. zone.
mobilePhone-UseZone	Mobile phone zone.
mobilePhone-FreeZone	Quiet area.
famillyArea	Family area.
childfreeArea	Child free area.
other	Other area.

7.7.14.5 Commercial Facility Support Values

For each of the COMMERCIAL FACILITIes there is an enumeration that specifes the allowed values and a list type that allows the selection of multiple instances of the value.

Table 263 — *Commercial Facilities – Value Lists*

List Of Values	TypeOfLocalService	Description
PassengerCommsFacilityList	***TypeOfCommunicationService***	Type of Communication service.
MoneyFacilityList	***TypeOfMoneyService***	Type of Money service.
HireFacilityList	***TypeOfHireService***	Type of Hire service.
RetailFacilityList	***TypeOfRetailService***	Type of Retail service.
CateringFacilityList	***TypeOfCateringService***	Type of Catering service.

7.7.14.5.1 Commercial Facilities – Physical Model

The following figure shows the allowed values for the various COMMERCIAL FACILITY lists.

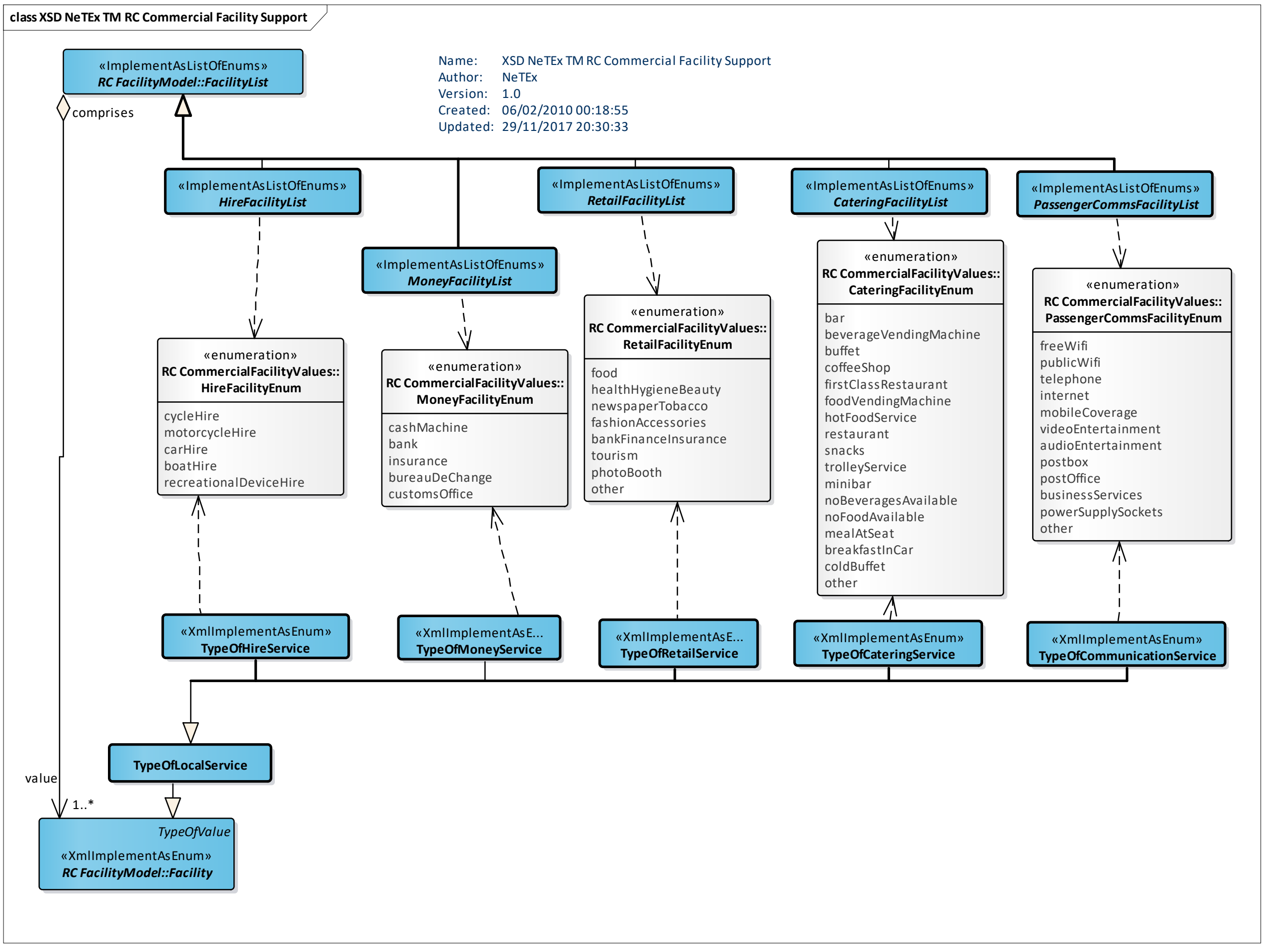


Figure 335 —*Commercial Facility Values – Physical Model (UML)*

7.7.14.5.2 Commercial Facilities – XSD Values

7.7.14.5.2.1 PassengerCommsFacilityType – Allowed Values

Allowed values for type of PASSENGER COMMS Facility (*PassengerCommsFacilityEnumeration*).

Table 264 — PassengerCommsFacilityType – *AllowedValues*

Value	Description
freeWifi	Free wifi.
publicWifi	Public wifi, may be charged.
telephone	Telephone.
internet	Internet access.
mobileCoverage	Mobile phone coverage.
videoEntertainment	Video entertainment.
audioEntertainment	Audio entertainment.
postbox	Post box.
postOffice	Post office.
businessServices	Business service.
powerSupplySockets	Power supply sockets.

7.7.14.5.2.2 HireFacilityType – Allowed Values

Allowed values for type of HIRE FACILITY (*HireFacilityEnumeration*).

Table 265 — HireFacilityType – *AllowedValues*

Value	Description
cycleHire	Cycle hire.
motorcycleHire	Motorcycle hire.
carHire	Car hire.
boatHire	Boat hire.
recreationalDeviceHire	Recreational device hire.

7.7.14.5.2.3 MoneyFacilityType – Allowed Values

Allowed values for type of MONEY FACILITY (*MoneyFacilityEnumeration*).

Table 266 — MoneyFacilityType – *AllowedValues*

Value	Description
cashMachine	Cash machine.
bank	Bank.
insurance	Insurance.
bureauDeChange	Bureau de change.

7.7.14.5.2.4 RetailFacilityType – Allowed Values

Allowed values for type of RETAIL FACILITY (*RetailFacilityEnumeration*).

Table 267 — RetailFacilityType – *AllowedValues*

Value	Description
food	Food retail outlets.
healthHygieneBeauty	Health and hygiene retail outlets.
newspaperTobacco	Newspaper and tobacco retail outlets.
fashionAccessories	Fashion and accessory retail outlets.
bankFinanceInsurance	Banks.
tourism	Tourist shops.
photoBooth	Photo booth.
other	Other retail outlets.

7.7.14.5.2.5 **CateringFacilityType – Allowed Values**

Allowed values for type of CATERING FACILITY (*CateringFacilityEnumeration*).

Table 268 — CateringFacilityType – *AllowedValues*

Value	Description
bar	Bar.
beverageVendingMachine	Beverage vending machines.
buffet	Buffet.
coffeeShop	Coffee shop.
firstClassRestaurant	First class restaurant.
foodVendingMachine	Food vending machines.
hotFoodService	Hot food available.
restaurant	Restaurant.
snacks	Snacks.
trolleyService	Trolley service.
minibar	Mini bar.
noBeveragesAvailable	No beverage available.
noFoodAvailable	No food available.
mealAtSeat	Meal at seat.
breakfastInCar	Breakfast car.
coldBuffet	Cold buffet.
other	Other refreshment facility.

7.7.14.6 Passenger Assistance Facility Support Values

For most of the types of PASSENGER ASSISTANCE FACILITies there is an enumeration that specifies the allowed values and a list type that allows the selection of multiple instances of the value.

Table 269 — *Passenger Assistance Facilities – Value Lists*

List Of Values	*TypeOfLocalService*	*Description*
AccessFacilityList	***AccessFacility***	SITE ACCESS FACILITY. +v1.1
AccessibilityInfoFacilityList	***AccessibilityInfoFacility***	A type of ACCESSIBILITY INFO FACILITY.
AccessibilityToolList	***AccessibilityTool***	A type of ACCESSIBILITY TOOL.
AssistanceFacilityList	***AssistanceFacility***	A type of ASSISTANCE FACILITY.
n/a	***EmergencyService***	A type of EMERGENCY SERVICE.
FamilyFacilityList	***FamilyFacility***	A type of FAMILY FACILITY.
LuggageServiceFacilityList	***TypeOfLuggageService***	A type of TYPE OF LUGGAGE SERVICE.
MobilityFacilityList	***MobilityFacility***	A type of MOBILITY FACILITY.
SafetyFacilityList	***SafetyFacility***	A type of SAFETY FACILITY.
n/a	***StaffingFacility***	A type of STAFFING FACILITY.

7.7.14.6.1 Passenger Assistence Facilities – Physical Model

The following figure shows the allowed values for the various PASSENGER ASSISTENCE FACILITY lists.

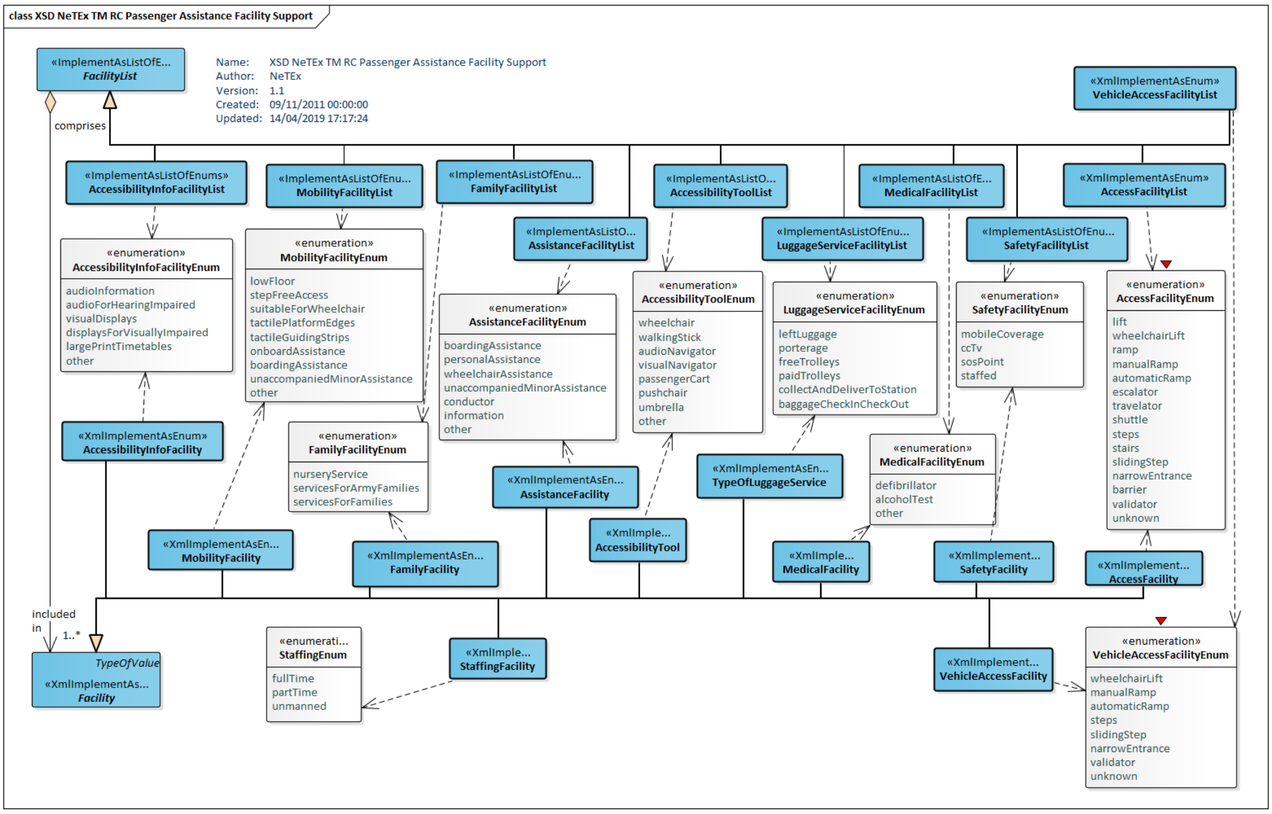


Figure 336 —*Passenger Assistance Facility Values – Physical Model (UML)*

7.7.14.6.2 Passenger Assistence Facilities – XSD Values

7.7.14.6.2.1 AccessibilityInfoFacilityType – Allowed Values

Allowed values for type of ACCESSIBILITY INFO (*AccessibilityInfoFacilityEnumeration*).

Table 270 — AccessibilityInfoFacilityType – *AllowedValues*

Value	Description
audioInformation	Audio information.
audioForHearingImpaired	Special Audio information for those who are hearing impaired.
visualDisplays	Visual displays of information.
displaysForVisually-Impaired	Specially enhanced visual information for those who are visually impaired.
largePrintTimetables	Large print timetables.
other	Other INFO FACILITY.

7.7.14.6.2.2 AccessibilityToolType – Allowed Values

Allowed values for Accessibility Tooltype of ACCESSIBILITY TOOL FACILITY (*AccessibilityToolEnumeration*).

Table 271 — AccessibilityToolType – *AllowedValues*

Value	Description
wheelchair	Wheelchairs available for passenger use.

walkingStick	Walking sticks available for passenger use.	*visualNavigator*	Audio navigators devices available.
audioNavigator	Audio navigators available for passenger use.	*other*	Other devices available.

7.7.14.6.2.3 AssistanceFacilityType – Allowed Values

Allowed values for type of ASSISTANCE SERVICE FACILITY (*AssistanceServiceFacilityEnumeration)*,

Table 272 — AssistanceFacilityType – *AllowedValues*

Value	Description		
boardingAssistance	Boarding assistance.	*unaccompanied-MinorAssistance*	Assistance for unaccompanied minors.
personalAssistance	Personal assistance.	*conductor*	There is a conductor.
wheelchairAssistance	Wheelchair assistance.	*information*	Information facility is available.
		other	Other information.

7.7.14.6.2.4 FamilyFacilityType – Allowed Values

Allowed values for type of FAMILY FACILITY (*FamilyFacilityEnumeration)*.

Table 273 — FamilyFacilityType – *AllowedValues*

Value	Description		
nurseryService	Nursery Service.	*servicesForArmyFamilies*	Services for Army families.
		servicesForFamilies	Services for families.

7.7.14.6.2.5 LuggageServiceFacilityType – Allowed Values

Allowed values for type of LUGGAGE SERVICE (*LuggageServiceFacilityEnumeration)*.

Table 274 — LuggageServiceFacilityType – *AllowedValues*

Value	Description		
leftLuggage	Left luggage.	*freeTrolleys*	Free luggage trolleys.
porterage	Porterage.	*paidTrolleys*	Luggage trolleys require payment to use.
		collectAndDeliver-ToStation	Luggage Collection and delivery service.

7.7.14.6.2.6 MobilityFacilityType – Allowed Values

Allowed values for type of MOBILITY FACILITY (*MobilityFacilityEnumeration*).

Table 275 — MobilityFacilityType – *AllowedValues*

Value	Description		
lowFloor	Low floor.	*stepFreeAccess*	Step free access.
		suitableForWheelchair	Suitable for wheelchairs.

tactilePlatformEdges	Platform has tactile edges to aid the visually impaired.
tactileGuidingStrips	There are tactile strips to aid the visually impaired.
onboardAssistance	There is onboard assistance.
boardingAssistance	There is boarding assistance.
unaccompaniedMinor-Assistance	There is assistance for unaccompanied minors.
other	Other type of assistance.

7.7.14.6.2.7 SafetyFacilityType – Allowed Values

Allowed values for type of SAFETY FACILITY (*SafetyFacilityEnumeration)*.

Table 276 — SafetyFacilityType – *AllowedValues*

Value	Description
mobileCoverage	There is mobile phone coverage.
ccTv	There is close circuit television camera surveillance.
sosPoint	There is an SOS point.
staffed	There are staff.

7.7.14.6.2.8 Staffing – Allowed Values

Allowed values for type of STAFFING (*StaffingEnumeration)*.

Table 277 — Staffing – *AllowedValues*

Value	Description
fullTime	Full Time staffing.
partTime	Part Time staffing.
unmanned	Unmanned.

7.7.14.7 Passenger Facility Support Values

For most of the types of PASSENGER FACILITies there is an enumeration that specifies the allowed values and a list type that allows the selection of multiple instances of the value.

Table 278 — *Passenger Facilities – Value Lists*

List Of Values	TypeOfLocalService	Description
EmergencyServiceFacilityList	***EmergencyService***	Type of EMERGENCY SERVICE.
n/a	***GenderLimitation***	Restriction on use to a specified gender.
LuggageLockerFacilityList	***LuggageLockerFacility***	A LUGGAGE LOCKER FACILITY.
n/a	***TypeOfMeetingPoint***	Type of MEETING POINT.
MealFacilityList	***MealFacility***	Type of MEAL FACILITY.
ParkingFacilityList	***ParkingFacility***	Type of PARKING FACILITY.
PassengerInformationEquipment List	***PassengerInformationFacility***	A type of PASSENGER INFORMATION FACILITY.

PassengerInformationFacilityList	***TypeOfPassengerInformation-Equipment***	Type of PASSENGER INFORMATION EQUIPMENT.
SanitaryFacilityList	***SanitaryFacility***	Type of SANITARY FACILITY.

7.7.14.7.1 Passenger Facilities – Physical Model

The following figure shows the allowed values for the various PASSENGER FACILITY lists.

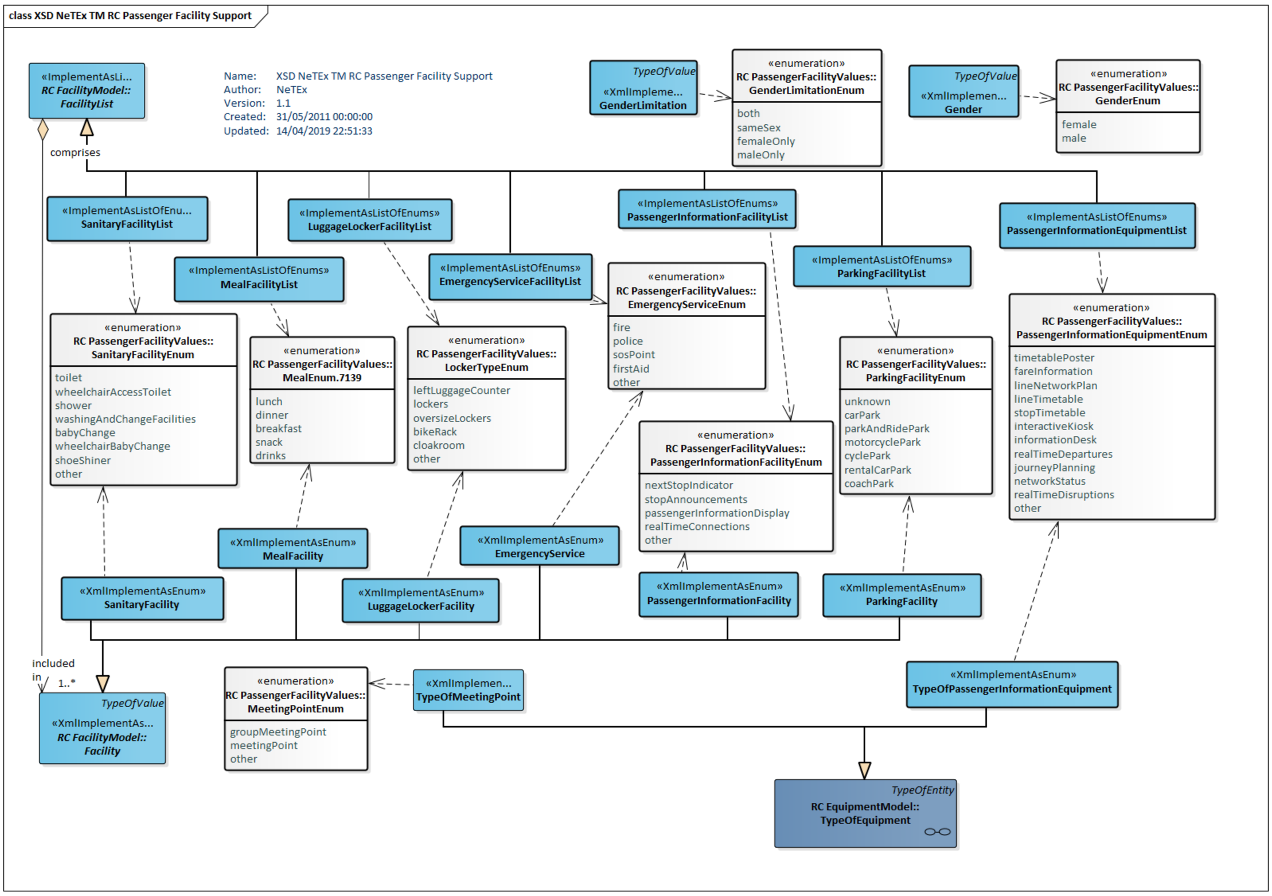


Figure 337 —*Passenger Facility Values – Physical Model (UML)*

7.7.14.7.2 Passenger Facilities – XSD Values

7.7.14.7.2.1 EmergencyServiceType – Allowed Values

Allowed values for type of EMERGENCY SERVICE (*EmergencyServiceEnumeration)*.

Table 279 — EmergencyServiceType – *AllowedValues*

Value	Description
fire	Fire brigade emergency services.
police	Police services.
sosPoint	SOS Point.
firstAid	First aid services.
other	Other emergency service.

7.7.14.7.2.2 GenderLimitation – Allowed Values

Allowed values for Gender Limitation (*GenderLimitationEnumeration*).

Table 280 — GenderLimitation – *AllowedValues*

Value	Description
both	Both genders.
maleOnly	Male only.
femaleOnly	Female only.

7.7.14.7.2.3 LuggageLockerType – Allowed Values

Allowed values for type of LUGGAGE LOCKER (*LuggageLockerEnumeration*).

Table 281 — LuggageLockerType – *AllowedValues*

Value	Description
leftLuggageCounter	Left Luggage Counter.
lockers	Lockers.
oversizeLockers	Oversize lockers.
bikeRack	Bike rack.
cloakroom	Cloakroom.

7.7.14.7.2.4 MeetingPointType – Allowed Values

Allowed values for type of MEETING POINT FACILITY (*MeetingPointEnumeration*).

Table 282 — MeetingPointType – *AllowedValues*

Value	Description
groupMeetingPoint	Group meeting point.
meetingPoint	Meeting point.
other	Other meeting point.

7.7.14.7.2.5 MealType – Allowed Values

Allowed values for type of MEAL - Based on TAP 7139 (*MealEnumeration.7139*).

Table 283 — MealType – *AllowedValues*

Value	Description
lunch	Lunch.
dinner	Dinner.
breakfast	Breakfast.
snack	Snack.

7.7.14.7.2.6 ParkingFacilityType – Allowed Values

Allowed values for type of PARKING FACILITY (*ParkingFacilityEnumeration*).

Table 284 — ParkingFacilityType – *AllowedValues*

Value	Description
unknown	Unknown type of parking .
carPark	Car park.
parkAndRidePark	Park and ride car park.
motorcyclePark	Motorcycle car park.
cyclePark	Cycle car park.

rentalCarPark	Rental car park.
coachPark	Coach park.

7.7.14.7.2.7 PassengerInformationEquipmentType – Allowed Values

Allowed values for type of PASSENGER INFORMATION Facility (*ParkingFacilityEnumeration*).

Table 285 — PassengerInformationEquipmentType – *AllowedValues*

Value	Description
timetablePoster	Timetable as poster.
fareInformation	Fare table.
lineNetworkPlan	Plan of line network.
lineTimetable	Timetable for whole line.
stopTimetable	Timetable just for a stop.
interactiveKiosk	Interactive kiosk.
informationDesk	Information desk.
realTimeDepartures	Real-time departures display.
journeyPlanning	Journey planning facility.
networkStatus	Network status. +v1.1
realTimeDisruption.	Real-tiem dsiruption info. +v1.1
other	Other information facility.

7.7.14.7.2.8 PassengerInformationFacilityType – Allowed Values

Allowed values for type of PASSENGER INFORMATION FACILITY (*PassengerInformationFacilityEnum*).

Table 286 — PassengerInformationFacilityType – *AllowedValues*

Value	Description
nextStopIndicator	Next stop Indicator.
stopAnnouncements	Stop announcements are made.
passengerInformation-Display	Display of passenger information.
realTimeConnections	Real time connection are shown.
other	other.

7.7.14.7.2.9 TypeOfSanitaryFacilityType – Allowed Values

Allowed values for type of SANITARY FACILITY (*TypeOfSanitaryFacilityEnumeration*).

Table 287 — TypeOfSanitaryFacilityType – *AllowedValues*

Value	Description
toilet	Lavatories.
wheelchairAccessToilet	Toilet for wheelchair access.
shower	Shower.
wheelchairBabyChange	Baby change facility for wheelchair users.
babyChange	Baby change facility.
washingAndChangeFacilities	Wash and Change facilities.
other	Other sanitary facility.

7.7.14.8 Ticketing Facility Support Values

For most of the types of TICKETING FACILITies there is an enumeration that specifies the allowed values and a list type that allows the selection of multiple instances of the value.

Table 288 — *Ticketing Facilities – Value Lists*

List Of Values	TypeOfLocalService	Description
BookingProcessFacilityList	***BookingProcessFacility***	Type of Booking Process Facility.
n/a	***GroupBooking***	Type of Group Booking Facility.
ServiceReservationFacilityList	***ServiceReservationFacility***	Type of Special Reservation Facility.
TicketingFacilityList	***TicketingFacility***	Type of Ticketing Facility.
TicketingServiceList	***TicketingServiceFacility***	Type of Ticketing Service Facility.
UicProductCharacteristicList	***UicProductCharacteristic***	UIC Product categories.
UicTrainRateList	***UicTrainRate***	Type of Train Rate.
FareClassList	***FareClass***	Type Of Fare Class.

7.7.14.8.1 Ticketing Facilities – Physical Model

The following figure shows the allowed values for the various TICKETING FACILITY lists.

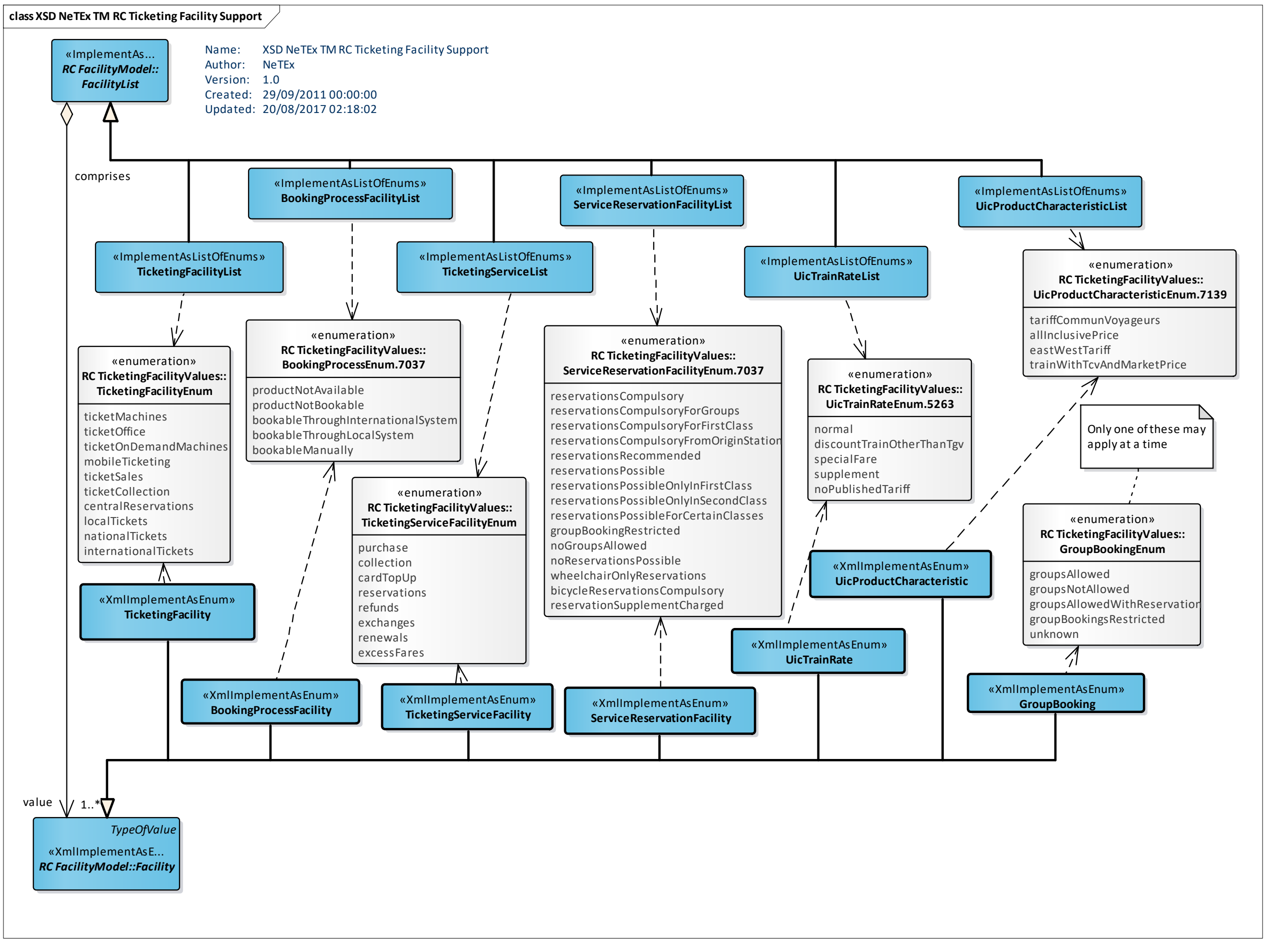


Figure 338 — *Ticketing Facility Values – Physical Model (UML)*

7.7.14.8.2 Ticketing Facilities – XSD Values

7.7.14.8.2.1 BookingProcessType – Allowed Values

Allowed values for type of TAP TSI service facility values - Based on TAP TSI 7037 (*BookingProcessEnumeration)*.

Table 289 — BookingProcessType – *AllowedValues*

Value	Description
productNotAvailable	Product is not available.
productNotBookable	Product is not bookable.
bookableThrough-InternationalSystem	Product is bookable through international systems.
bookableThrough-LocalSystem	Product is bookable through local systems.
bookableManually	Product is bookable manually.

7.7.14.8.2.2 GroupBookingType – Allowed Values

Allowed values for type of GROUP BOOKING *(GroupBookingEnumeration).*

Table 290 — GroupBookingType – *AllowedValues*

Value	Description
groupsAllowed	Groups are allowed.
groupsNotAllowed	Groups are not allowed.
groupsAllowedWithReservation	Groups are allowed with reservations.
groupsBookingsRestricted	There are restrictions on group bookings.
unknown	Possibility of Group booking not known.

7.7.14.8.2.3 ServiceReservationFacilityType – Allowed Values

Allowed values for type of TAP TSI service facility values. For comparison purposes - Based on TAP TSI 7037 (*ServiceReservationFacilityEnum*eration.7037*)*.

Table 291 — ServiceReservationFacilityType – *AllowedValues*

Value	Description
reservationsCompulsory	Reservations are compulsory.
reservationsCompulsoryForGroups	Reservations are compulsory for groups.
reservationsCompulsoryForFirstClass	Reservations are compulsory for First class.
reservationsCompulsoryFromOriginStation	Reservations are compulsory form origin station.
reservationsRecommended	Reservations are recommended.
reservationsPossible	Reservations can be made.

reservationsPossibleOnlyInFirstClass	Reservations can be made only in first class.
reservationsPossibleOnlyInSecondClass	Reservations can be made only in second class.
reservationsPossibleForCertainClasses	Reservations can be made only for certain classes.
groupBookingRestricted	Restrictions apply to group bookings.
noGroupsAllowed	Group tickets are not allowed.
noReservationsPossible	No reservations can be made.
wheelchairOnlyReservations	Only wheelchair reservations can be made.
bicycleReservationsCompulsory	Reservations for bicycles are compulsory.
reservationSupplementCharged	There is a supplementary charge for making a reservation.

7.7.14.8.2.4 **TicketingFacilityType – Allowed Values**

Allowed values for type of TICKETING FACILITY (T*icketingFacilityEnumeration)*.

Table 292 — TicketingFacilityType – *AllowedValues*

Value	Description
ticketMachines	Ticket machines.
ticketOffice	Ticket office.
ticketOnDemandMachines	Ticket on demand machines.
mobileTicketing	Mobile Ticketing.
ticketSales	Ticket sales .
ticketCollection	Ticket collection.
centralReservations	Central reservations can be made.
localTickets	Local Ticket sales.
nationalTickets	National Ticket sales.
internationalTickets	International Ticket sales.

7.7.14.8.2.5 **TicketingServiceFacilityType – Allowed Values**

Allowed values for type of TICKETING SERVICE (*TicketingServiceFacilityEnum*).

Table 293 — TicketingServiceFacilityType – *AllowedValues*

Value	Description
purchase	Purchase.
collection	Collection.
cardTopUp	Card Top up.
reservations	Reservations.

7.7.14.8.2.6 **UicProductCharacteristicType – Allowed Values**

Allowed values for type of TAP TSI service facility values - Based on TAP TSI 7037 (*UicProductCharacteristicEnum.7139)*.

Table 294 — UicProductCharacteristicType – *AllowedValues*

Value	Description
tariffCommunVoyageurs	Fare is for Tariff Communal Voyageurs.

allInclusivePrice	Price is all inclusive.
eastWestTariff	East west tariff.
trainWithTcvAndMarketPrice	Fare is for Tariff Communal Voyageurs and Market Price.

7.7.14.8.2.7 UicTrainRateEnum.5263 – Allowed Values

Allowed values for type of TAP TSI facilityTrain Rate values. Based on TAP TSI 5263 (*UicTrainRateEnum.5263)*.

Table 295 — UicTrainRateEnum.5263 – *AllowedValues*

Value	Description
normal	Normal fare.
discountTrainOtherThanTgv	Discount other than tgv.
specialFare	Special fare.
supplement	Supplementary fare.
noPublishedTariff	No Tariff is Published.

7.7.14.8.2.8 TapTsiTrainFacilityType – Allowed Values

Allowed values for type of TAP TSI Train facility values - Based on TAP TSI 5263 (*TapTsiTrainFacilityEnumeration)*.

Table 296 —TapTsiTrainFacilityType – *AllowedValues*

Value	Description
FirstClassSleeper	First Class Sleeper.
StandardClassSleeper	Standard Class Sleeper.
FirstClassSeats	First Class Seats.
SecondClassSeats	Second Class Seats.
FirstClassCouchettes	First Class Couchettes.
SecondClassCouchettes	Second Class Couchettes.
RecliningSeat	Reclining Seat.
Restaurant	Restaurant.
FirstClassSleeperSingle	First Class Sleeper Single.
FirstClassSleeperSpecial	First Class Sleeper Special.
FirstClassSleeperDouble	First Class Sleeper Double.
VehicleTransport	Vehicle Transport.
SecondClassSleeperT2	Second class sleeper T2.
SecondClassSleeperT3	Second class sleeper T3.
SecondClassSleeperT4	Second class sleeper T4.
FirstClassSleeperSingle-Shower	First Class Sleeper Single with shower.
FirstClassSleeperDoubleShower	First Class Sleeper Double with shower.
NonSmoker	Non - smoker accommodation.
HeavilyDisabled	Accommodation for the heavily disabled.
BabyCare	Baby care.
Bicycle	Bicycle.
WheelchairAccess	Wheelchair access.
Video	Video entertainment onboard.
Minibar	Minibar.
PanoramaCoach	Panorama coach.
TelephoneService	Telephone service.
PowerSupplySockets	Power supply sockets.

PullmanCoach	Pullman coach.
FamilyCompartment	Family compartment.
BuffetMachine	Buffet Machine.
PremiumClass	Premium Class.
Preferente	Preferente class.
Turista	Turista class.
FirstClassSleeperSingle-ShowerWC	First Class Sleeper Single with shower & WC.
FirstClassSleeperDoubleShowerWC	First Class Sleeper Double with shower & WC.

SecondClassSleeperT2-ShowerWC	Second Class Sleeper T2 with shower & WC.
SecondClassSleeperT3-ShowerWC	Second Class Sleeper T3 with shower & WC.
SecondClassCouchetteC2	Second Class Couchette C2.
SecondClassCouchetteC4	Second Class Couchette C4.
SecondClassCouchetteC5	Second Class Couchette C5.
SecondClassCouchetteC6	Second Class Couchette C6.
SecondClassCouchetteWheelchair	Second Class Couchette for Wheelchair passenger.

7.7.14.9 XML Examples of Facilities

7.7.14.9.1 Facillity – XML Fragment Example

The following example shows the use of a Service Facility Set for a Journey.

EXAMPLE

```
<ServiceJourney version="any" id="hde::sj_24o_02">
    <DepartureTime>15:20:00.0Z</DepartureTime>

    ::>
    <calls>
    ::>
    </calls>
    <facilities>
        <ServiceFacilitySetRef version="any" ref="hde::sfs_24o_01"/>
    </facilities>
</ServiceJourney>
::>
<ServiceFacilitySet version="any" id="hde::sfs_24o_01">
    <ProvidedByRef ref="xyz::4567">EXTERNAL</ProvidedByRef>
    <AssistanceFacilityList>boardingAssistance                                          conductor
wheechairAssistance</AssistanceFacilityList>
    <FareClasses>standardClass</FareClasses>
    <MobilityFacilityList> stepFreeAccess suitableForWheelchairs</MobilityFacilityList>
    <NuisanceFacilityList>noSmoking</NuisanceFacilityList>
    <TicketingServiceFacilityList> purchase </TicketingServiceFacilityList>
    <LuggageCarriageFacilityList>noBaggageStorage</LuggageCarriageFacilityList>
    <PassengerInformationFacilityList>nextStopIndicator                        realTimeConnections
stopAnnouncements</PassengerInformationFacilityList>
    <OtherFacility/>
</ServiceFacilitySet>
```

7.7.15 Service Restrictions

7.7.15.1 SERVICE RESTRICTIONs – Conceptual MODEL

The Service Restrictions model defines a small number of reusable elements for classifying ticketing, equipment and other elements.

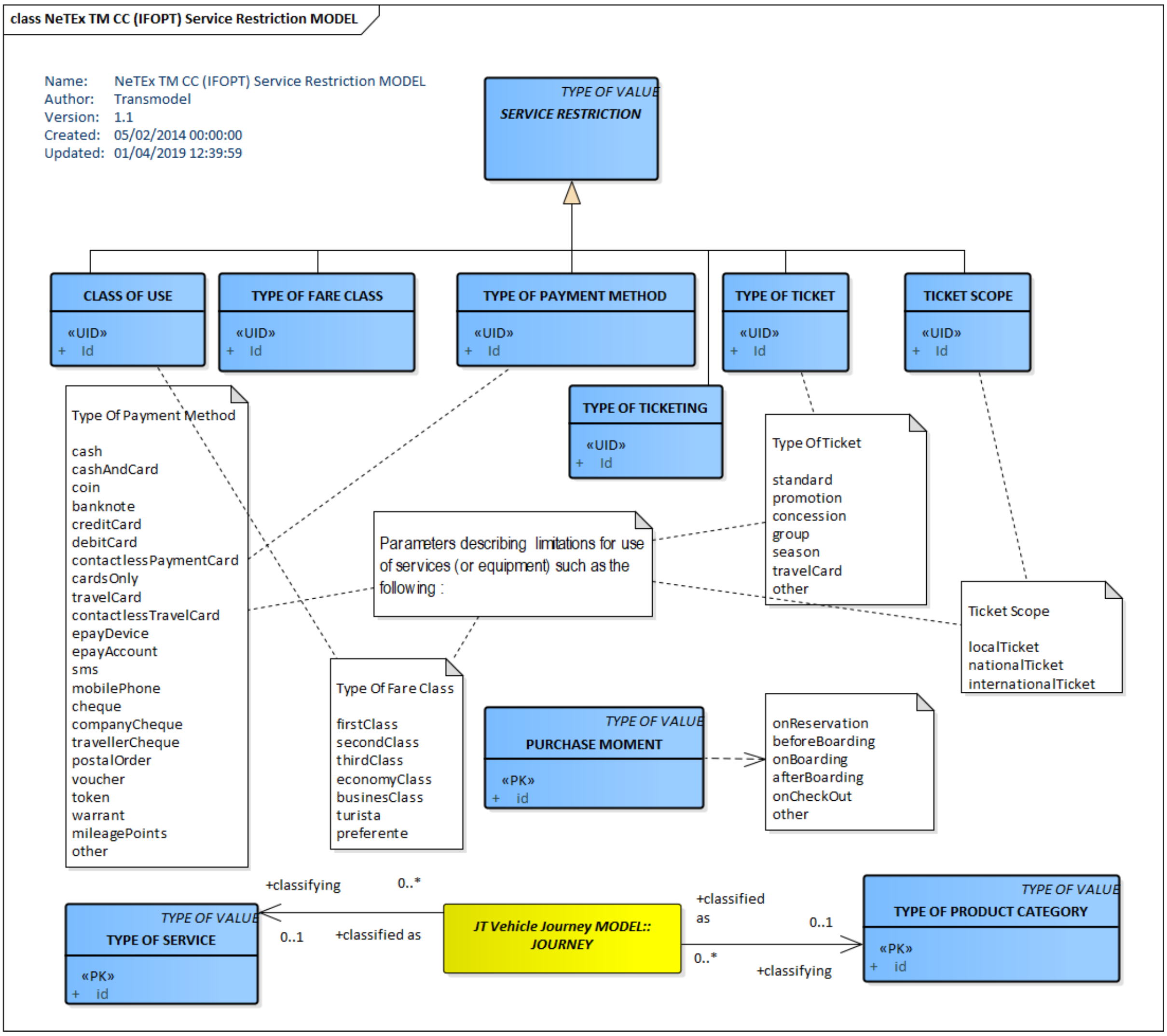


Figure 339 — ServiceRestrictions – Conceptual MODEL (UML)

7.7.15.2 Service Restriction – Physical Model

The Service Restrictions model defines a small number of reusable elements for classifying ticketing, equipment and other elements.

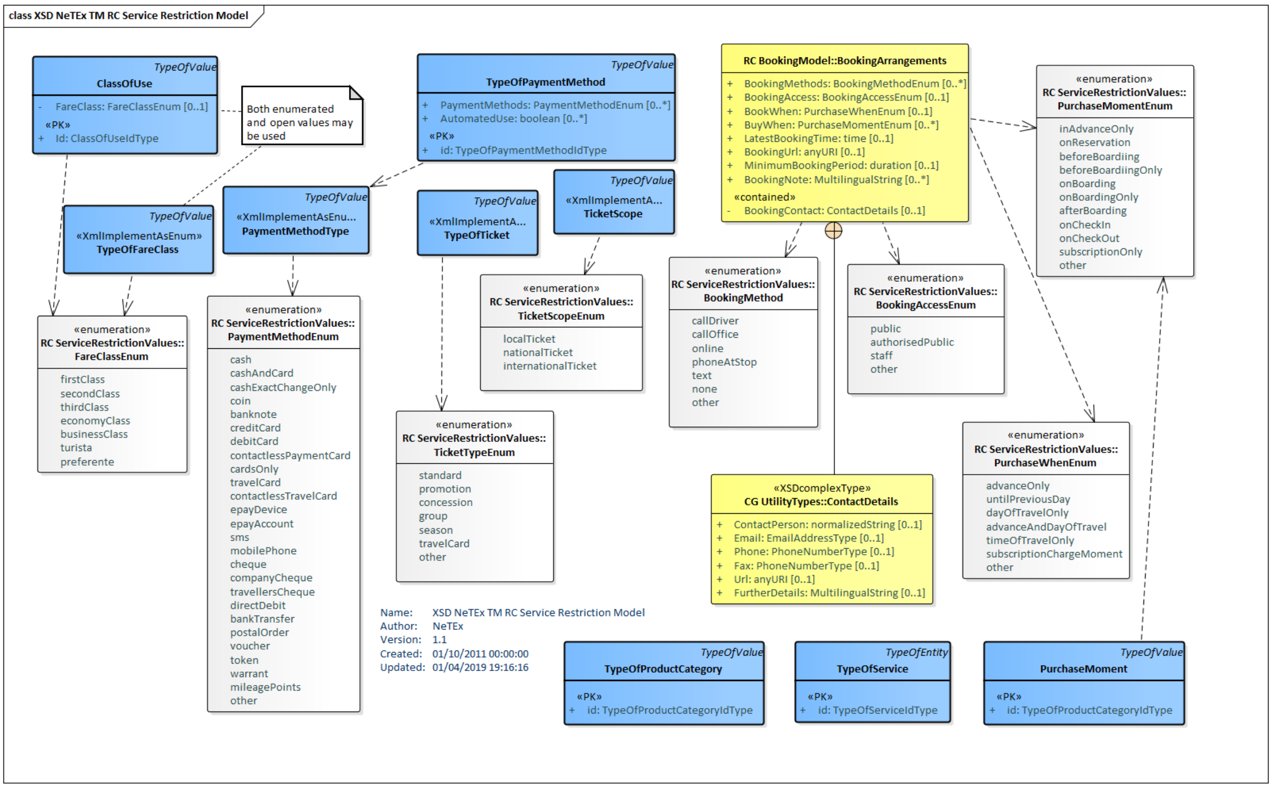


Figure 340 — ServiceRestrictions – Physical Model (UML)

7.7.15.3 Service Restrictions – Attributes and XSD

7.7.15.3.1 ClassOfUse – Model Element

An extendable classification of fare and other service classes by category of user entitled to use them.

Table 297 — ClassOfUse – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	CLASS OF USE inherits from TYPE OF VALUE.
«PK»	***id***	*ClassOfUseIdType*	1:1	Identifier of CLASS Of USE.
«enum»	***FareClass***	*FareClassEnum*	1:1	Existing FareClass associated with f CLASS Of USE. See allowed values below.

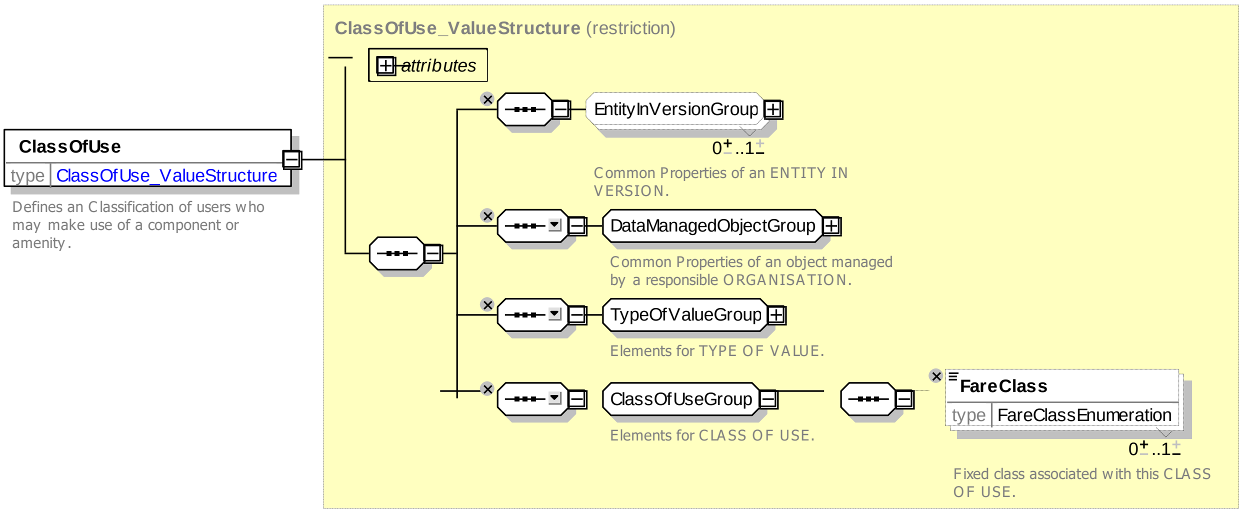


Figure 341 — *ClassOfUse – XSD*

7.7.15.3.2 TypeOfFareClass – Model Element.

A fixed classification for FARE CLASSes, implemented as an enumeration.

Allowed values for FARE CLASS (*FareClassEnumeration).*

Table 298 — *FareClass – AllowedValues*

Value	Description
firstClass	First class.
secondClass	Second class.
thirdClass	Third class.
economyClass	Economy class.
businessClass	Business class.
premiumClass	Premium class
turista	Spanish Turista class .
preferente	Spanish Preferente class .
unknown	Unnown class of use.

7.7.15.3.3 TypeOfPaymentMethod – Model Element

An extensible classification for PAYMENT METHOD (e.g. cash, credit card, debit card, travel card, contactless travel card, mobile phone, token, etc.).

Table 299 — TypeOfPaymentMethod – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE OF PAYMENT METHOD inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfPayment-MethodIdType*	1:1	Identifier of TYPE OF PAYMENT METHOD.
«enum»	***PaymentMethod***	*PaymentMethodEnum*	0:1	Existing fixed value payment method associated with TYPE OF PAYMENT METHOD . See allowed values below.

	AutomatedUse	xsd:boolean	0:1	Whether payment method can be used or automated payments, e.g. for direct debit.

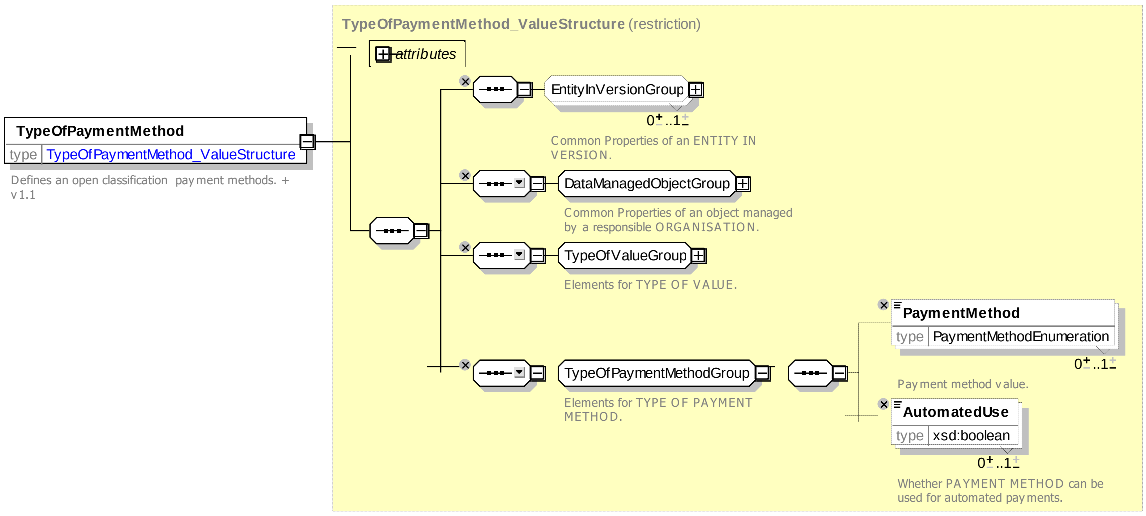


Figure 342 — *TypeOfPaymentMethod – XSD*

7.7.15.3.3.1 PaymentMethod – Allowed Values

Allowed values for PAYMENT METHOD (*PaymentMethodEnumeration*).

Table 300 — *PaymentMethod – AllowedValues*

Value	Description
cash	Cash
cashExactChangeOnly	Cash in exact coins and notes. +v1.1
cashAndCard	Cash and Card
coin	Coin only (no notes)
banknote	Banknotes only +v1.1
cheque	Cheque
travellersCheque	Traveller's cheque
companyCheque	Company cheque +v1.1
*postal**Xxxxx***	Postal order
creditCard	Credit card
debitCard	Debit card
cardsOnly	Only credit and debit cards accepted
travelCard	Travel card
contactlessPaymentCard	Contactless credit or debit card +v1.1
contactlessTravelCard	Contactless travel card (Smartcard)
directDebit	Direct debit +v1.1
bankTransfer	Bank transfer +v1.1
epayDevice	Payment with device e.g. ApplePay, etc. +v1.1
epayAccount	Payment with online account. +v1.1
sms	SMS
mobilePhone	Charge to mobile account
voucher	Voucher

token	Token
warrant	Warrant
mileagePoints	Mileage points +v1.1
other	Other payment method

7.7.15.3.4 TicketScope – Model Element

*S*cope of ticket that may be purchased or use.

7.7.15.3.4.1 TicketScope – Allowed Values

Allowed values for TICKET SCOPE (*ScopeOfTicketEnumeration).*

Table 301 — *TicketScope – AllowedValues*

Value	Description
localTicket	Local ticket
nationalTicket	National ticket
internationalTicket	International ticket
other	other

7.7.15.3.5 TypeOfTicket – Model Element

A classification for TICKETs available at a TICKETING EQUIPMENT (e.g. standard, concession, promotion, group, season, travel card, etc.).

7.7.15.3.5.1 TicketType – Allowed Values

Allowed values for TICKET TYPE (*TicketTypeEnumeration*).

Table 302 — *TicketType – AllowedValues*

Value	Description
standard	Standard ticket
promotion	Promotional ticket
concession	Concessionary ticket
group	Group ticket
season	Season ticket
carnet	Carnet of tickets
travelCard	Travel card ticket
other	Other ticket
all	All tickets

7.7.15.3.6 BookingArrangements – Model Element

Details of the booking arrangements for a LINE or SERVICE JOURNEY, e.g. if flexible.

Table 303 – *BookingArrangements* – Element

Classification	Name	Type	Cardinality	Description
««cntd»	***BookingContact***	*ContactDetails*	0:1	Booking contact details. +v1.1
«enum»	***BookingMethods***	*BookingMethodEnum*	0:*	Booking method(s) to use. See allowed values below.
«enum»	***BookingAccess***	*BookingAccessEnum*	0:1	Who can make a booking. See allowed values below.
«enum»	***BookWhen***	*PurchaseWhenEnum*	0:1	When Purchase may be made. See allowed values below. +v1.1

«enum»	***BuyWhen***	*PurchaseMomentEnum*	0:*	When purchase may be made. See allowed values below. +v1.1
	LatestBookingTime	*MultilingualString*	0:1	Latest time in day that booking can be made.
	MinimumBooking-Period	*xsd:duration*	0:1	Minimum interval in advance of departure day or time that service may be ordered.
	BookingUrl	*anyURI*	0:1	Booking url if different from info url (ie Contact URL). +v1.1
«cntd»	***BookingNote***	*MultilingualString*	0:1	Additional notes about Booking.

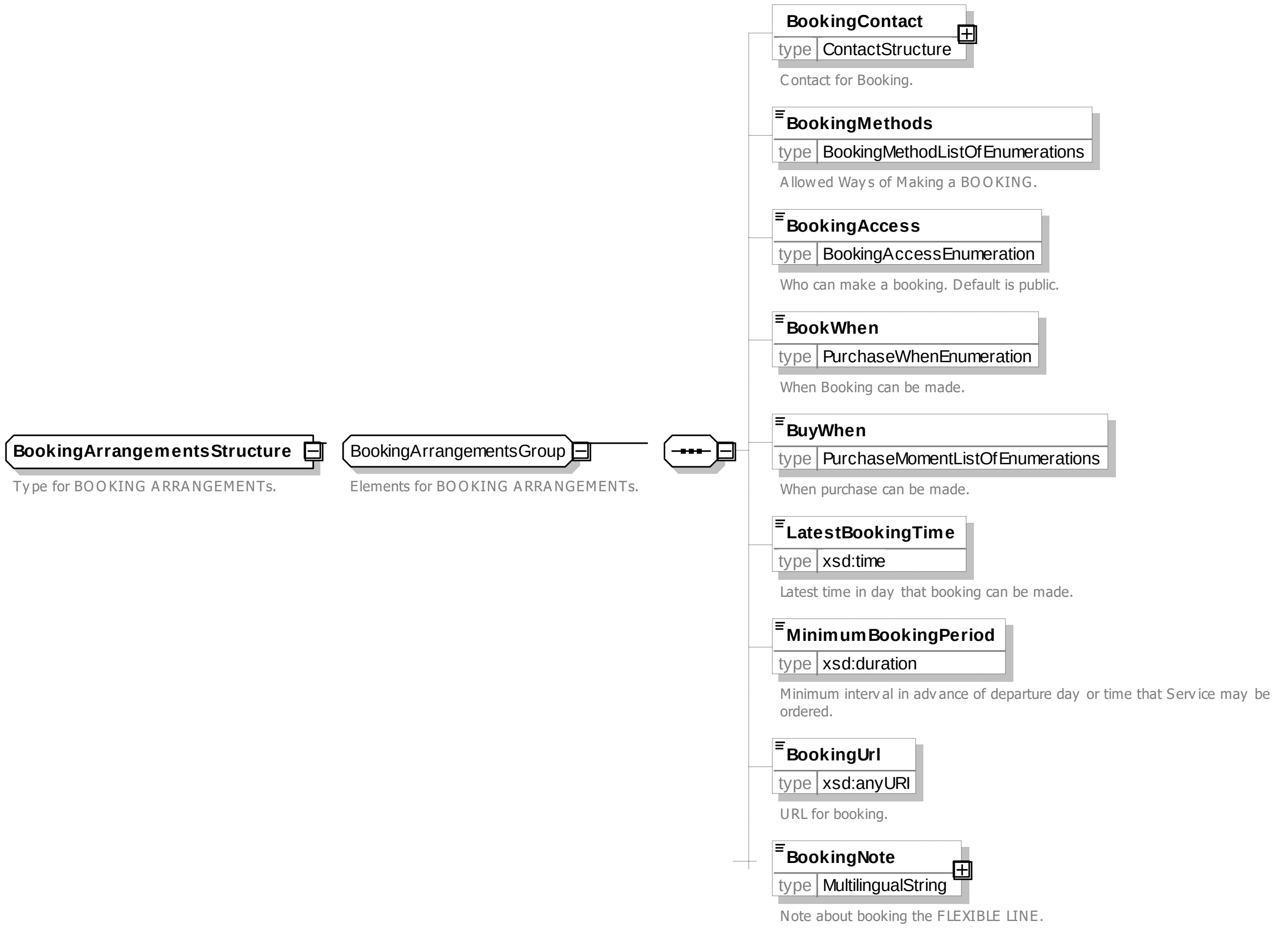


Figure 343 – *BookingArrangements – XSD*

7.7.15.3.6.1 BookingMethod – Allowed values

Allowed values for ***BookingMethod*** (*BookingMethodEnumeration).*

Table 304 –*BookingMethod* – Allowed Values

Value	Description
callDriver	Call the driver to book a journey.
callOffice	Call an office to book a journey.
online	Book a journey online.

other	Book a journey by other means.
phoneAtStop	Book by using phone at stop.
text	Send a text message to book a journey.
none	No booking method.

7.7.15.3.6.2 BookingAccessType – Allowed values

Allowed values for Booking Access Type *(BookingAccessEnumeration).*

Table 305 – *BookingAccessType* – Allowed Values

Value	Description
public	Public may make a booking.
authorisedPublic	Certain members of public may make a booking.
staff	Staff may make a booking.
other	Other limitation on who may make a booking.

7.7.15.3.6.3 PurchaseWhen – Allowed values

Allowed values for ***BookWhen*** *(PurchaseWhenEnumeration).*

Table 306 – *PurchaseWhenType* – Allowed Values

Value	Description
advanceOnly	Purchase may only be made in advance.
untilPreviousDay	Purchase may only be made in advance up until the day previous to travel.
dayOfTravelOnly	Purchase may only be made on day of travel.
advanceAndDayOfTravel	Purchase may be made in advance or on day of travel.
timeOfTravelOnly	Purchase only at time of travel.
subscriptionChargeMoment	Purchase only at time of subscription installment.
other	Other limitation on who may make a booking.
advanceOnly	Purchase may only be made in advance.

7.7.15.3.6.4 *PurchaseMoment* – Allowed values

Allowed values for ***BuyWhen*** *(PurchaseMomentEnumeration).*

Table 307 – *PurchaseMoment* – Allowed Values

Value	Description
inAdvanceOnlyy	Ticket must be purchased in advance of travel.
onReservation	Ticket must be purchased on reservation.
beforeBoarding	Ticket can be purchased before boarding.

beforeBoardiingOnly	Ticket must be purchased before boarding.
onBoarding	Ticket can be purchased on boarding.
onBoardingOnly	Ticket must be purchased on boarding.
afterBoarding	Ticket may be purchased after boarding.
onCheckIn	Ticket may be purchased on entering the system
onCheckOut	Ticket may be purchased on leaving the system.
subscriptionOnly	Ticket may only be purchased on subscription
other	Other point of purchase.

7.7.15.3.7 TypeOfProductCategory– Model Element

An extensible classification for PRODUCT CATEGORY used to categories both journeys and fares.

Table 308 — TypeOfProductCategory – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE PRODUCT CATEGORY inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfProductCategory-IdType*	1:1	Identifier of TYPE OF PRODUCT CATEGORY .
«FK»	***ExternalProduct-CategoryRef***	*ExternalObjectRef*	0:1	dentifies the PRODUCT CATEGORY. Specifically for use in AVMS systems that require an alias, if code is different from main identifier. For VDV compatibility.

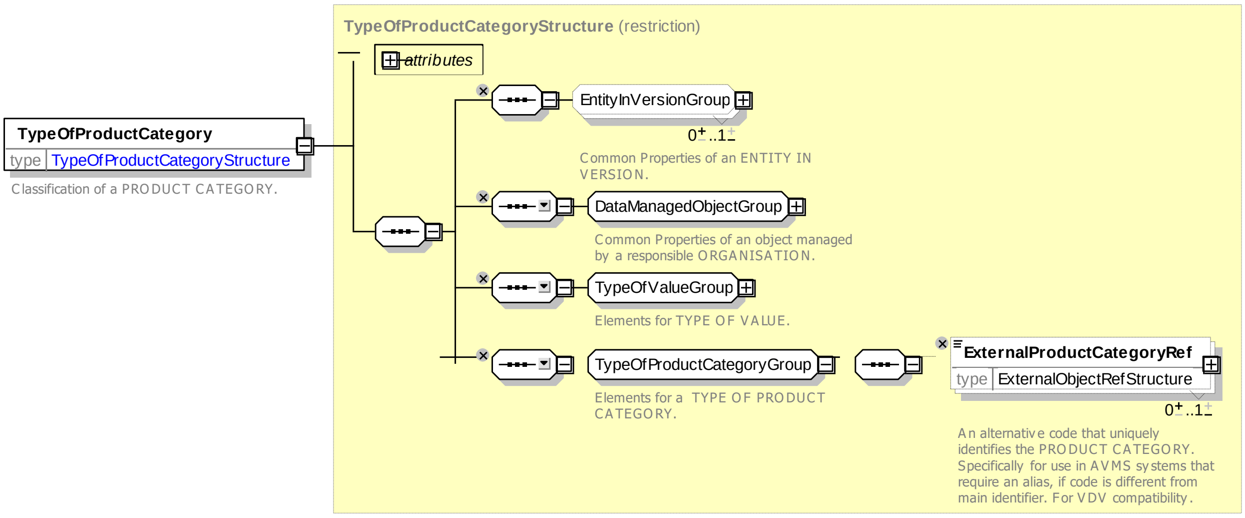


Figure 344 —*TypeOfProductCategory– XSD*

7.7.15.3.8 TypeOfService – Model Element

An extensible classification for SERVICE

Table 309 — TypeOfService – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfValue*	::>	TYPE OF SERVICE inherits from TYPE OF VALUE.
«PK»	***id***	*TypeOfServiceIdType*	1:1	Identifier of TYPE OF SERVICE .

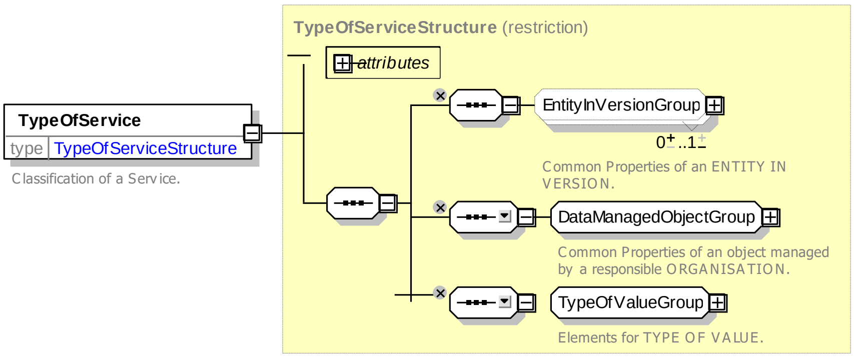


Figure 345 — *TypeOfService –XSD*

7.7.16 Train

7.7.16.1 TRAIN – Conceptual MODEL

The TRAIN Conceptual MODEL represents VEHICLE TYPE properties that are peculiar to TRAINs. A TRAIN may comprise not just a single VEHICLE but a chain of carriages, TRAIN ELEMENTS, assembled as TRAIN COMPONENTs. Groups of carriages may be managed as sections by composing TRAINs into a COMPOUND TRAIN made up of TRAINs IN COMPOUND TRAIN, for example in a train that joins or splits.

TRAIN ELEMENTS can be classified with a TYPE OF TRAIN.

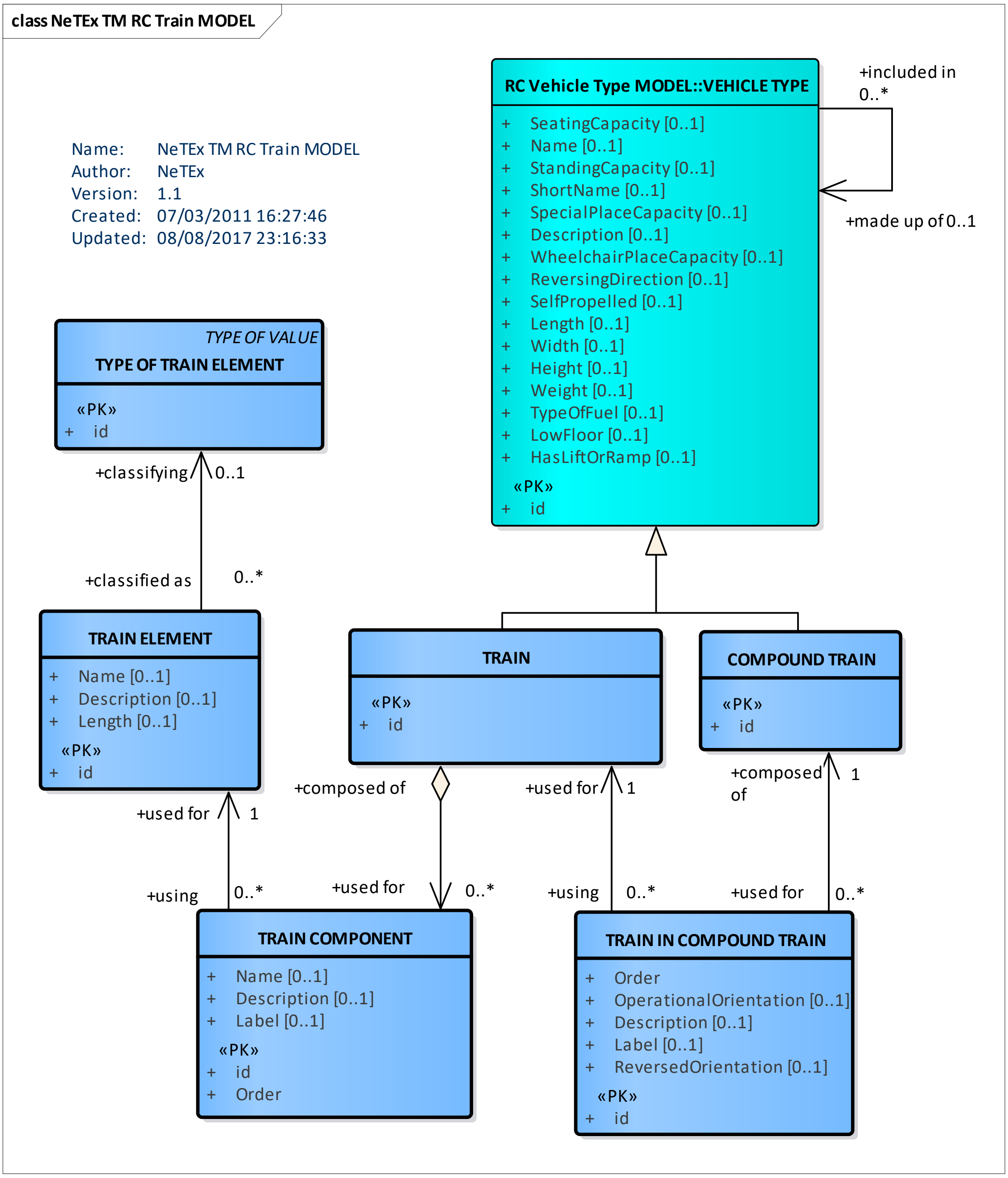


Figure 346 — Train – Conceptual MODEL (UML)

7.7.16.1.1 Example of a Train

The following figure shows how a train can be represented as an ordered collection of TRAIN COMPONENTs.

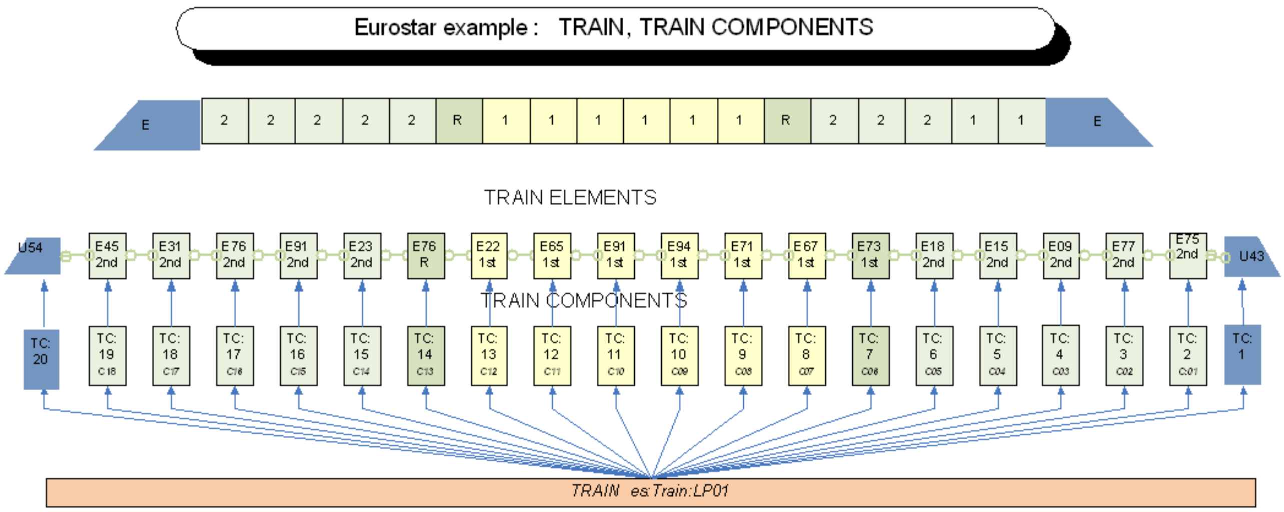


Figure 347 — Train Elements Example

The following figure shows a real life example of a Train Makeup.

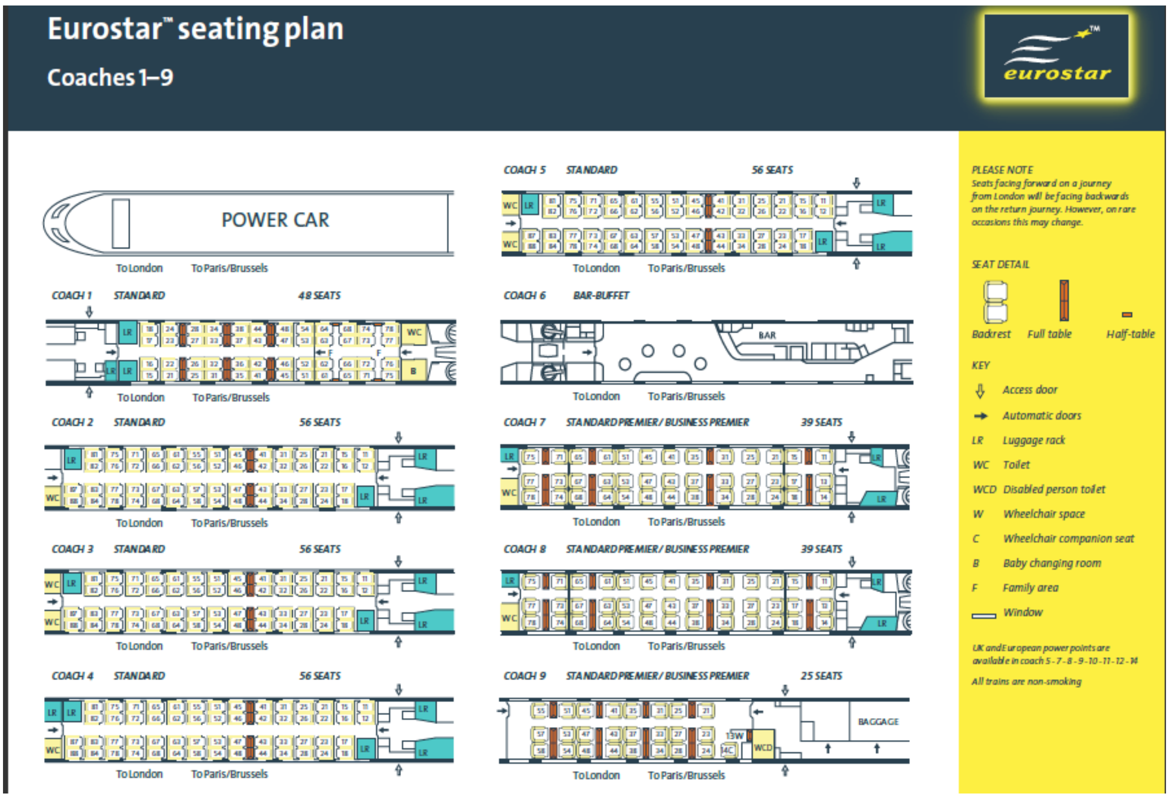


Figure 348 — Eurostar Train Makeup

7.7.16.2 Train – Physical Model

The TRAIN Physical model specialised the VEHICLE TYPE element.

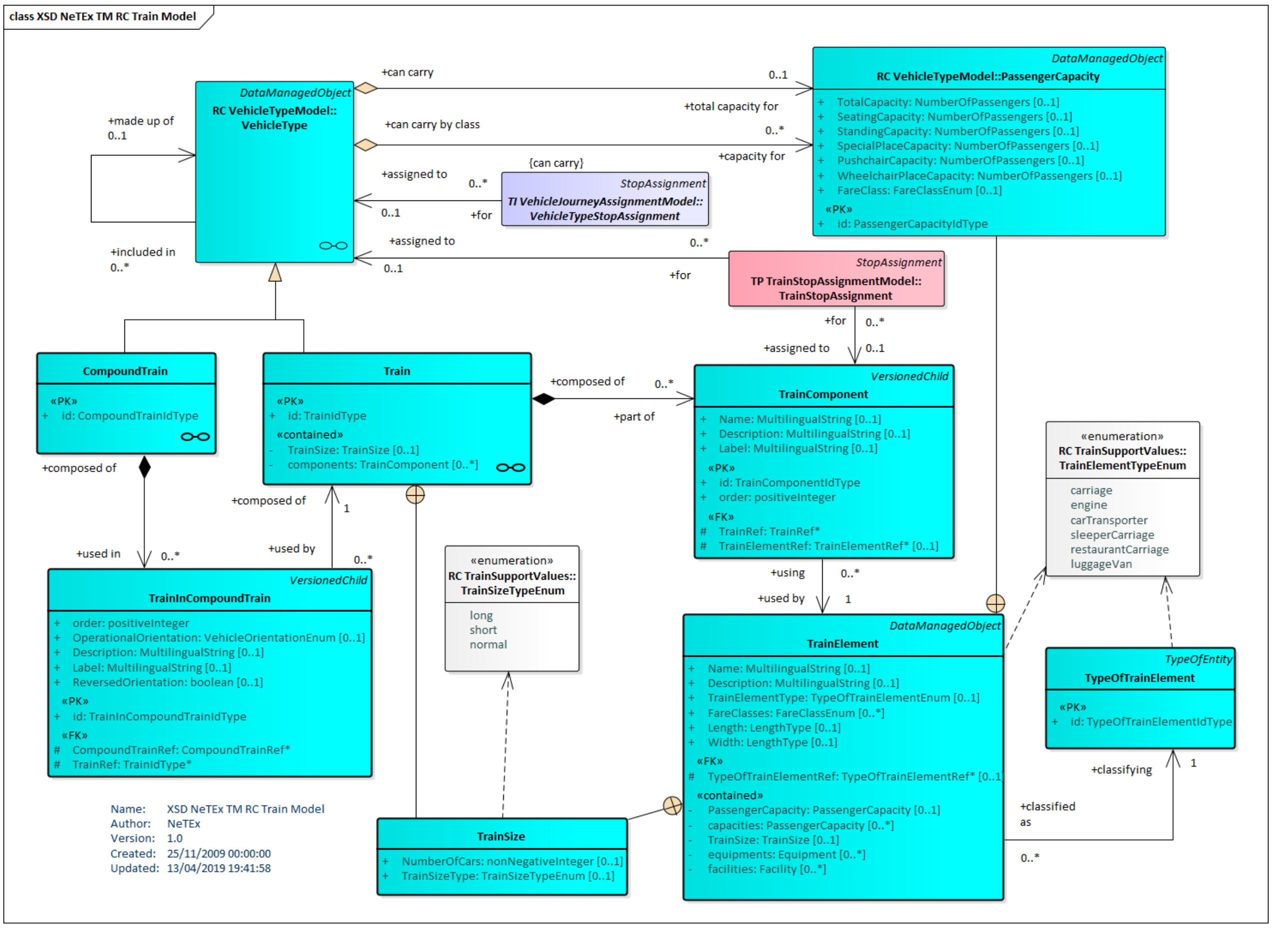


Figure 349 — Train – Physical Model (UML)

7.7.16.3 Train – Attributes and XSD

7.7.16.3.1 Train – Model Element

A vehicle composed of TRAIN ELEMENTs in a certain order, i.e. of wagons assembled together and propelled by a locomotive or one of the wagons.

Table 310 — *Train – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VehicleType*	::>	TRAIN inherits from VEHICLE TYPE.
«PK»	***id***	*TrainIdType*	1:1	Identifier of TRAIN.
	TrainSize	*xsd:nonNegativeIntege*	0:1	Train size Properties of TRAIN.
«cntd»	***components***	*TrainComponent*	0:*	Components of TRAIN.

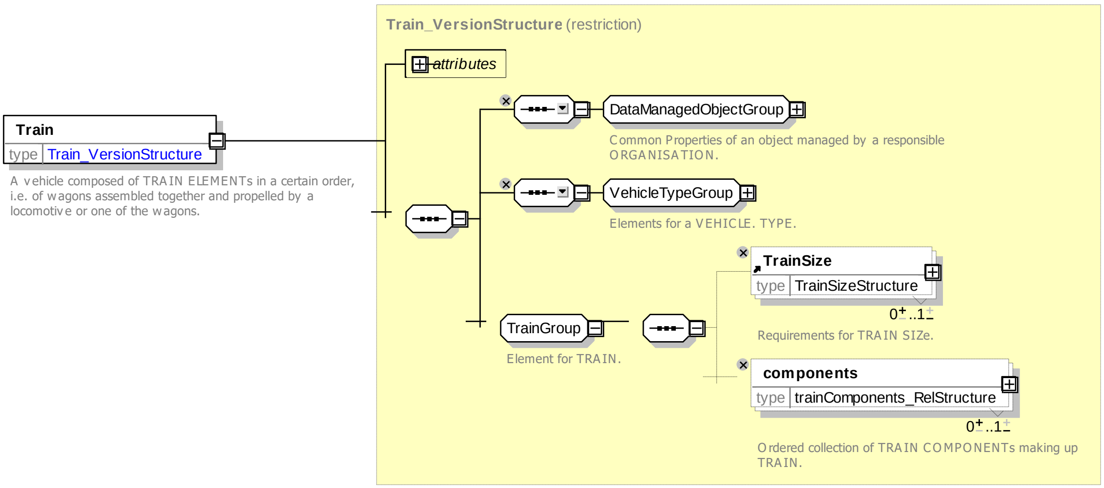


Figure 350 — *Train – XSD*

7.7.16.3.2 TrainSize – Model Element

Properties categorising the size of a train.

Table 311 — *TrainSize – Element*

Classification	Name	Type	Cardinality		Description
	NumberOfCars	*xsd:nonNegativeInteger*	0:1		Number of cars
«enum»	***TrainSizeType***	*TrainSizeEnum*	0:1		Categorisation values below.

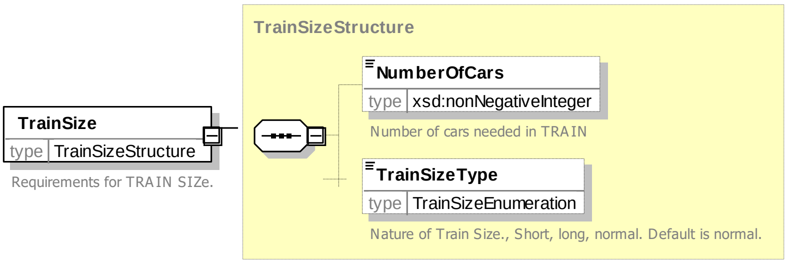


Figure 351 — TrainSize – XSD

7.7.16.3.2.1 TrainSizeTypeEnum – Allowed Values

Allowed values for ***TrainSize***. (*TrainSizeEnumeration*)

Table 312 — *TrainSize – AllowedValues*

Value	Description
long	Long train
short	Short train
normal	Normal Train

7.7.16.3.3 TrainComponent – Model Element

A specification of the order of TRAIN ELEMENTs in a TRAIN.

Table 313 — *TrainComponent – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	TRAIN COMPONENT inherits from VERSIONED CHILD.
«PK»	***id***	*TrainComponentIdType*	1:1	Identifier of TRAIN COMPONENT.
«atr»	***order***	*xsd:positiveInteger*	0:1	Order of COMPONENT within train
	Label	*MultilingualString*	0:1	Label of TRAIN COMPONENT.
	Description	*MultilingualString*	0:1	Description of TRAIN COMPONENT.
«FK»	***TrainRef***	*TrainRef*	0:1	Reference to TRAIN of which this is a part.
		CHOICE		Reference or inline embedd.
«FK»	a ***Train-ElementRef***	*TrainElementRef*	1:1	Reference to TRAIN ELEMENT associated with TRAIN COMPONENT.
«cntd»	b ***TrainElement***	*TrainElement*	1:1	TRAIN ELEMENT associated with TRAIN COMPONENT.
«enum»	***Operational-Orientation***	*VehicleOrientationEnum*	0:1	Orientation of the TRAIN. See VEHICLE TYPE model for allowed values.

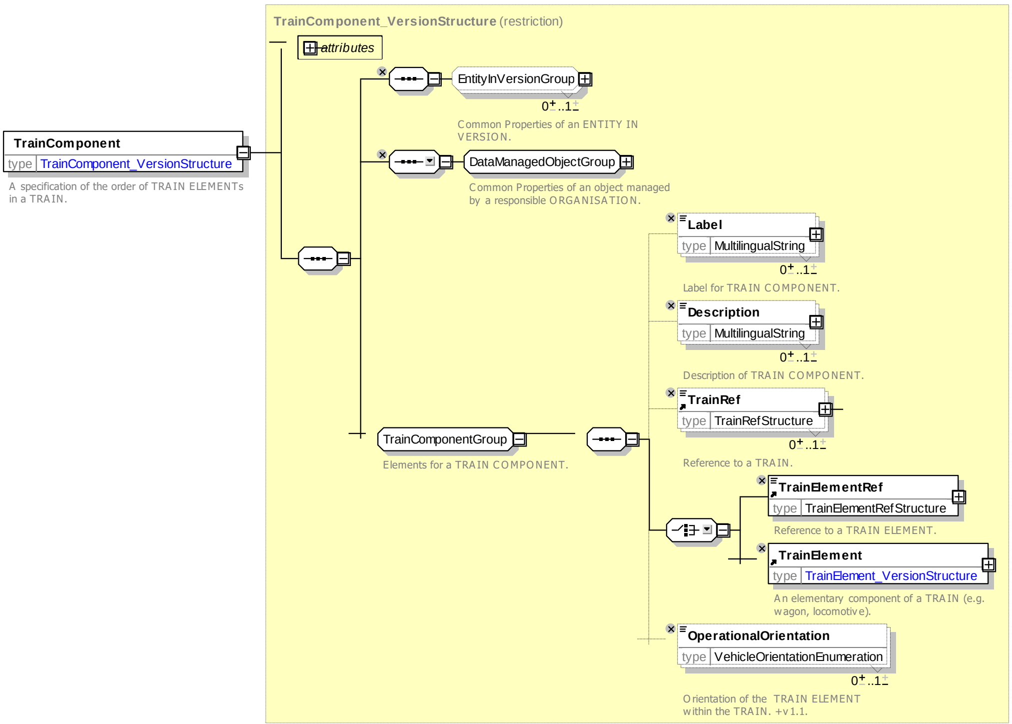


Figure 352 — *TrainComponent – XSD*

7.7.16.3.4 TrainElement – Model Element

An elementary component of a TRAIN (e.g. wagon, locomotive).

Table 314 — *TrainElement – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	TRAIN ELEMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*TrainElementIdType*	1:1	Identifier of TRAIN ELEMENT.
	Name	*MultilingualString*	0:1	Name of TRAIN ELEMENT.
	Description	*MultilingualString*	0:1	Description of TRAIN ELEMENT.
«enum»	***TrainElement-Type***	*TypeOfTrainElementEnum*	1:1	Classification of TRAIN ELEMENT.
«enum»	***FareClasses***	*FareClassEnum*	0:*	FARE CLASSes for TRAIN ELEMENT.
«cntd»	***Passenger-Capacity***	*PassengerCapacity*	0:1	Capacity of TRAIN ELEMENT.
Cntd»	***TrainSize***	*TrainSize*	0:1	Size of TRAIN.
	Length	*LengthType*	0:1	Length of TRAIN ELEMENT.
	Width	*LengthType*	0:1	The width of a TRAIN ELEMENT.
«cntd»	***facilities***	*ServiceFacilitySet*	0:*	SERVICE FACILITY SETs of TRAIN ELEMENT.
«cntd»	***equipments***	*(Equipment)*	0:*	Equipment of TRAIN ELEMENT.

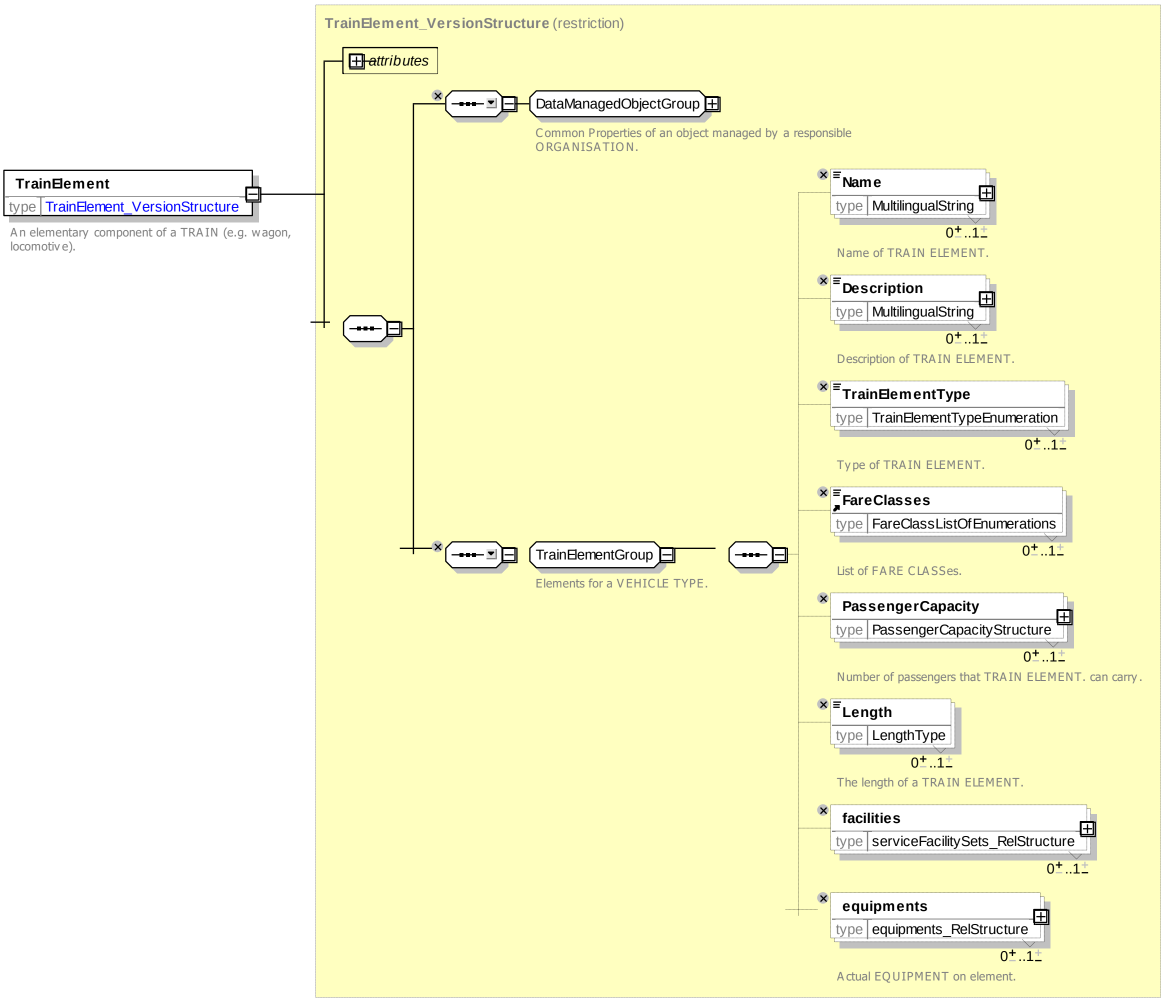


Figure 353 — *TrainElement – XSD*

7.7.16.3.4.1 TrainElementType – Allowed Values

Allowed values for type of Train Element (*TrainElementTypeEnumeration*).

Table 315 — TrainElementType – *AllowedValues*

Value	Description
carriage	Carriage
engine	Engine
carTransporter	Car Transporter carriage
sleeperCarriage	Sleeper carriage
restaurantCarriage	Restaurant carriage
luggageVan	Luggage van

7.7.16.3.5 TypeOfTrainElement – Model Element

A classification of TRAIN ELEMENTs.

Table 316 — *TypeOfTrain – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF TRAIN inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfTrainElementIdType*	1:1	Identifier of TYPE OF TRAIN.

7.7.16.3.6 CompoundTrain – Model Element

The specification of a COMPOUND TRAIN.

Table 317 — CompoundTrain – *Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VehicleType*	::>	COMPOUND TRAIN inherits from VEHICLE TYPE.
«PK»	***id***	*CompoundTrainIdType*	1:1	Identifier of COMPOUND TRAIN.

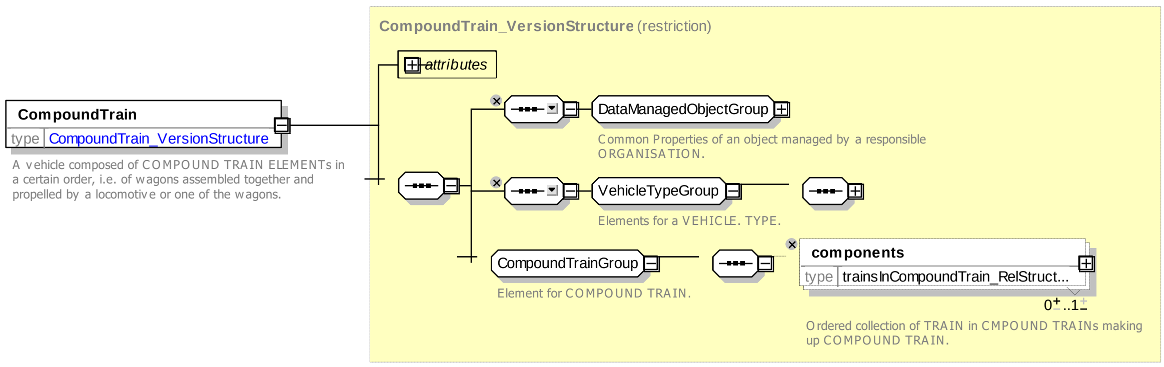


Figure 354 — *CompoundTrain – XSD*

7.7.16.3.7 TrainInCompoundTrain – Model Element

The specification of the order of TRAINs in a COMPOUND TRAIN.

Table 318 — *TrainInCompoundTrain – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	TRAIN IN COMPOUND TRAIN inherits from VERSIONED CHILD.
«atr»	***order***	*xsd:positiveInteger*	1:1	Order of TRAIN in COMPOUND TRAIN.

«PK»	***id***	*TrainInCompoundTrainIdType*	1:1	Identifier of TRAIN in COMPOUND TRAIN.
	Description	*MultilingualString*	0:1	Description of TRAIN in COMPOUND TRAIN.
«FK»	***CompoundTrain-Ref***	*CompoundTrainRef*	1:1	Reference to COMPOUND TRAIN.
		CHOICE		Can reference or embed Train
«FK»	a ***TrainRef***	*TrainRef*	1:1	Reference to TRAIN that is part of a COMPOUND TRAIN.
«cntd»	b ***Train***	*Train*	1:1	TRAIN that is part of a COMPOUND TRAIN.
«enum»	***Operational-Orientation***	*VehicleOrientationEnum*	0:1	Orientation of the TRAIN. See VEHICLE TYPE model for allowed values.
	Reversed-Orientation	*xsd:boolean*	0:1	Whether the component order of the TRAIN IN COMPOUND TRAIN is reversed compared to the order in the TRAIN.
	Label	*MultilingualString*	0:1	Label of TRAIN in COMPOUND TRAIN.

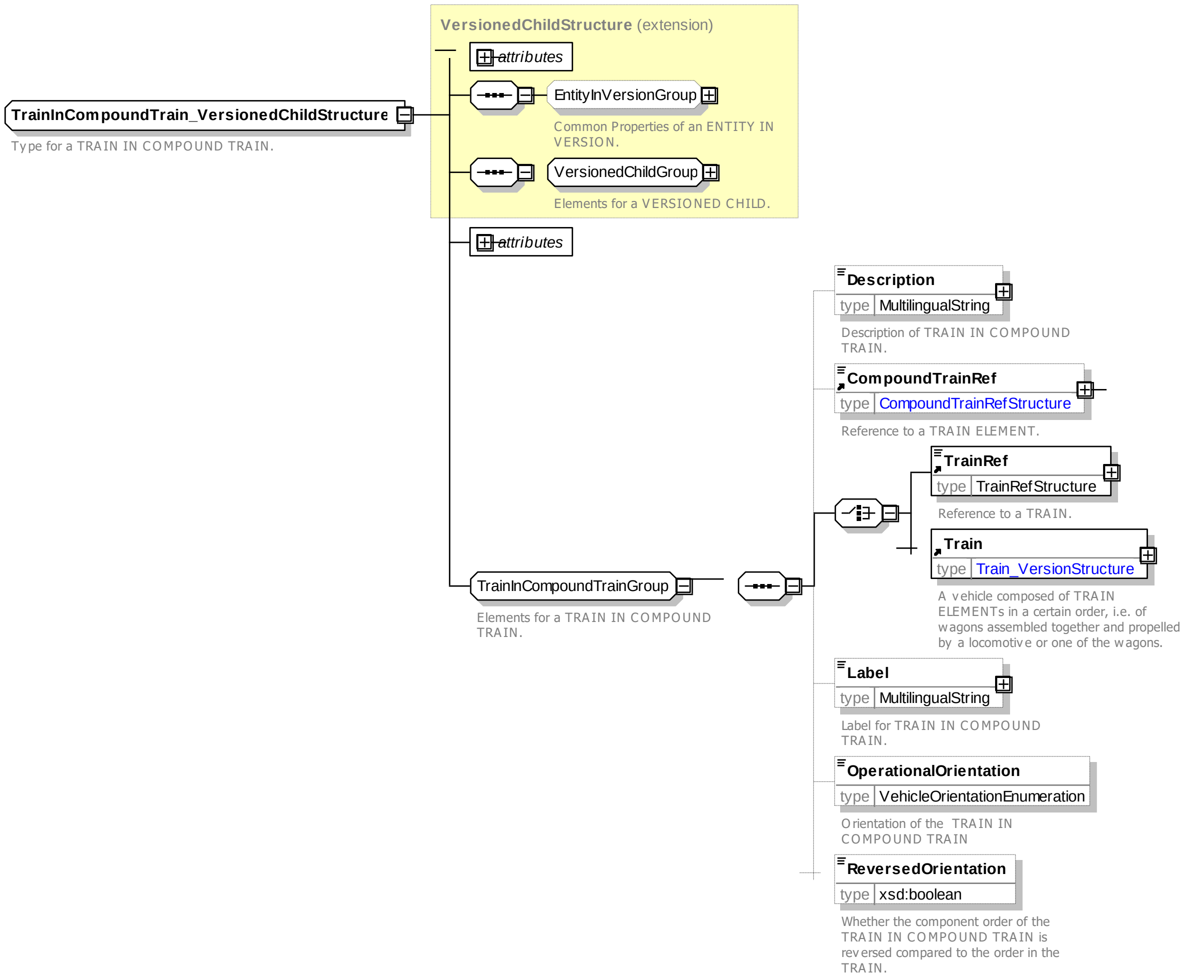


Figure 355 — *TrainInCompoundTrain – XSD*

7.7.16.3.8 TrainComponentView – Model Element

A VIEW of a TRAIN COMPONENT .

Table 319 — *TrainInComponentView– Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DerivedView*	::>	TRAIN COMPONENT VIEW inherits from DerivedView
«PK»	***TrainComponentRef***	*TrainComponentRef*	1:1	Reference to a TRAIN COMPONENT.
	Description	*MultilingualString*	0:1	Description of TRAIN COMPONENT.
	Label	*MultilingualString*	0:1	Label of TRAIN COMPONENT.
	ShortName	*MultilingualString*	0:1	Short Name of TRAIN ELEMENT.
«FK»	***TrainElement-Type***	*TypeOfTrainElementEnum*	1:1	Classification of TRAIN ELEMENT. See allowed values above.

«enum»	***FareClasses***	*FareClassEnum*	0:*	FARE CLASSes for TRAIN ELEMENT. See Service Restrictions model.

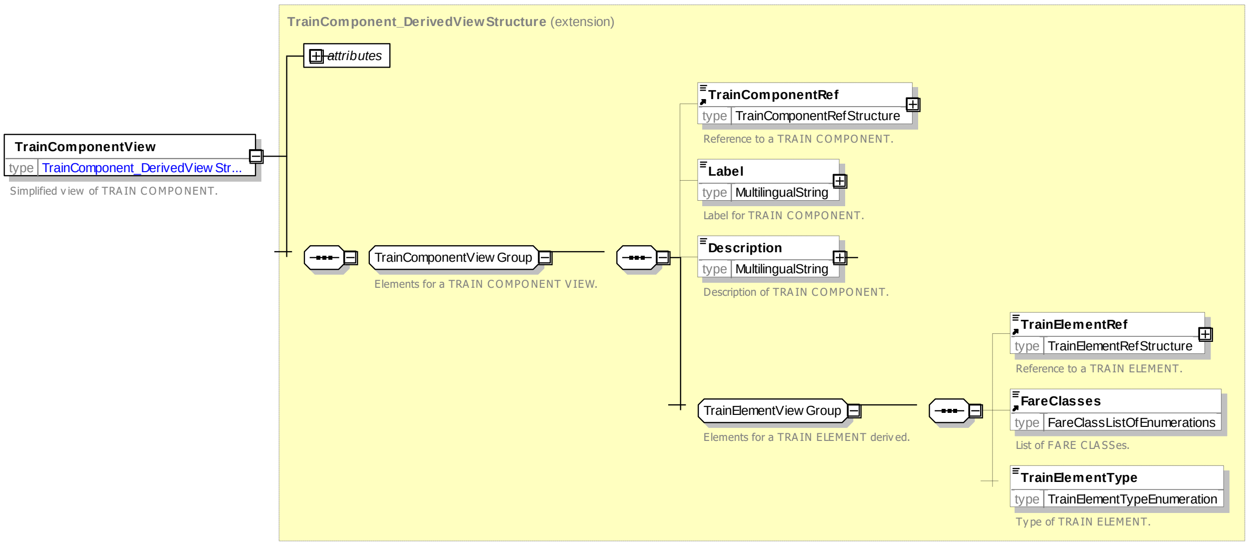


Figure 356 — *TrainInComponentView– XSD*

7.7.16.4 XML Examples of Trains

7.7.16.4.1 Train – XML Fragment Example

The following XML fragment shows a TRAIN.

EXAMPLE

```
<Train version="any" id="hde:trn_40447">
    <Name>40447 Hanover - Copenhagen</Name>
    <Description>2 2 2 1 1 E</Description>
    <SelfPropelled> true</SelfPropelled>
    <facilities>
        <ServiceFacilitySetRef version="any" ref="hde:svcfc_general"/>
        <ServiceFacilitySetRef version="any" ref="hde:svcfc_first"/>
    </facilities>
    <components>
        <TrainComponent version="any" id="hde:trncmp_40447_01" order="1">
            <Label>Engine </Label>
            <Description>Engine</Description>
            <TrainElement version="any" id="hde:trne_40447_01">
                <Name/>
                <TrainElementType>engine</TrainElementType>
            </TrainElement>
        </TrainComponent>
        <TrainComponent version="any" id="hde:trncmp_40447_02" order="2">
            <Label>Carriage A</Label>
            <Description>Front Carriage 1st Class</Description>
            <TrainElement version="any" id="hde:trne_40447_02">
                <Name/>
                <TrainElementType>carriage</TrainElementType>
                <FareClasses> firstClass</FareClasses>
            </TrainElement>
        </TrainComponent>
        <TrainComponent version="any" id="hde:trncmp_40447_03" order="3">
            <Label>Carriage B</Label>
            <Description>2nd Carriage 1st CLass</Description>
            <TrainElement version="any" id="hde:trne_40447_03">
                <Name/>
                <TrainElementType>carriage</TrainElementType>
                <FareClasses> firstClass</FareClasses>
            </TrainElement>
```

```
                </TrainComponent>
::>….
::::
            <Description>5th Carriage</Description>
            <TrainElement version="any" id="hde:trne_40447_06">
                <Name/>
                <TrainElementType>carriage</TrainElementType>
                <FareClasses>standardClass</FareClasses>
            </TrainElement>
                </TrainComponent>
            </components>
        </Train>
```

7.7.16.4.2 Compound Train – XML Fragment Example

The following XML fragment shows a Compound TRAIN.

EXAMPLE

```
        <CompoundTrain version="any" id="hde:ctrn_XX-447">
            <Name>447 + 457 + 40447 Amsterdam - Hannover</Name>
            <Description>E - 2 2 2 2 2 2 R 1 1 - 2 2 2 2 2 - 2 2 2 1 1 - E</Description>
            <SelfPropelled> true</SelfPropelled>
            <components>
                <TrainInCompoundTrain version="any" id="hde:trninctrn_XX-447_40447" order="1">
                    <TrainRef version="any" ref="hde:trn_40447"/>
                    <Label>40447</Label>
                </TrainInCompoundTrain>
                <TrainInCompoundTrain version="any" id="hde:trninctrn_XX-447_457" order="2">
                    <TrainRef version="any" ref="hde:trn_457"/>
                    <Label>457</Label>
                </TrainInCompoundTrain>
                <TrainInCompoundTrain version="any" id="hde:trninctrn_XX-447_447" order="3">
                    <TrainRef version="any" ref="hde:trn_447"/>
                    <Label>447</Label>
                </TrainInCompoundTrain>
            </components>
        </CompoundTrain>
```

7.7.17 Schematic Map

7.7.17.1 SCHEMATIC MAP – Conceptual MODEL

The published passenger Information for a complex transport interchange often includes schematic maps to show the relative parts and facilities located within the interchange. In an interactive presentation to passengers using an electronic device, these maps may be linked to other elements, for example, to see the properties of a piece of equipment.

NeTEx includes a generic representation of such a map that may be linked to NeTEx elements independently of any rendering in a specific mark-up such as HTML or JavaScript

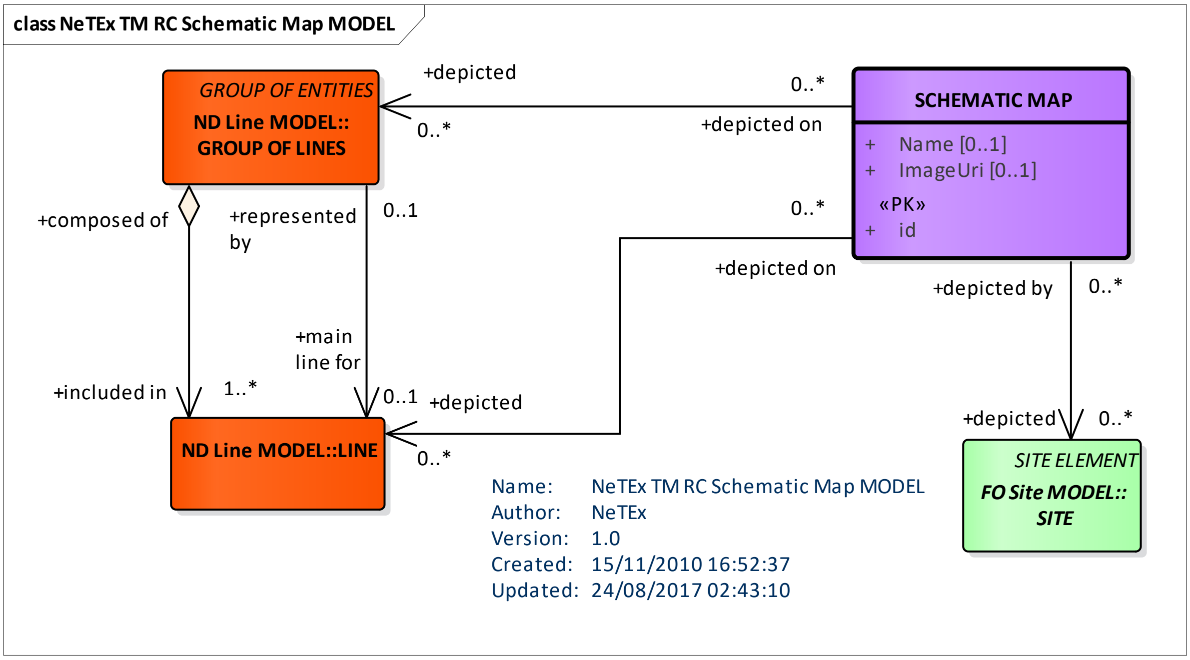


Figure 357 — Schematic Map – Conceptual MODEL (UML)

7.7.17.1.1 Schematic Map Example – Network Diagram

The following figures show examples of a SCHEMATIC MAP for a line.

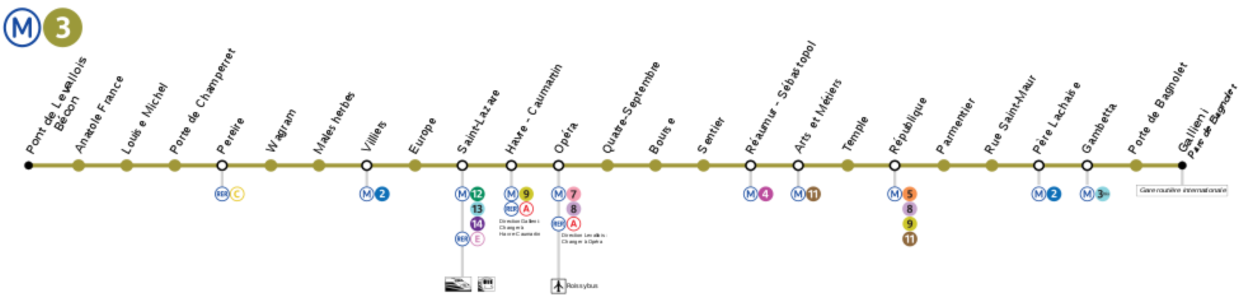


Figure 358 — Schematic Map – Line example *(image from http://commons.wikimedia.org/wiki/File:Metro_Paris_M3-plan.svg)*

The following figures show examples of a SCHEMATIC MAP for a Network (Paris Metro).

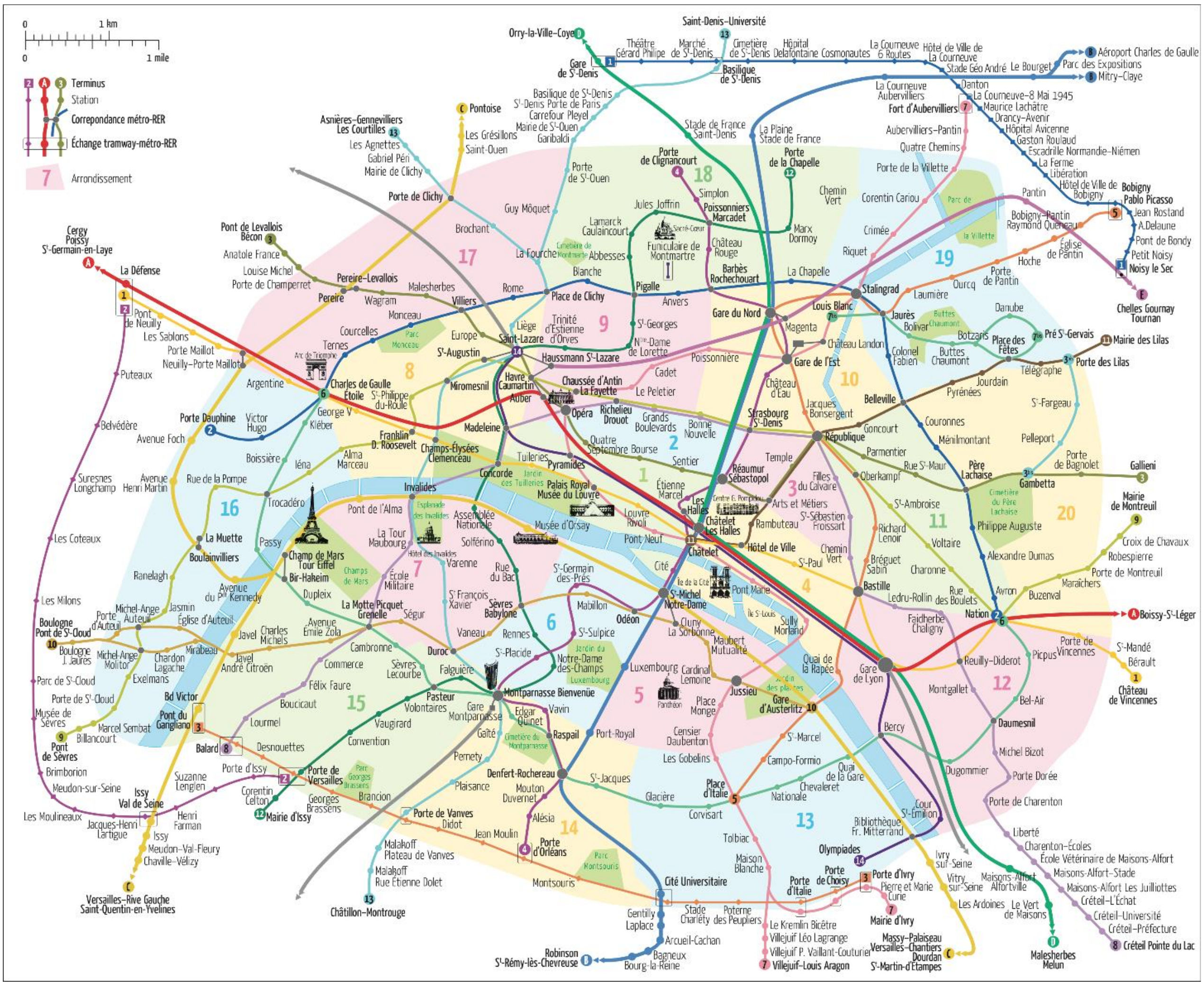


Figure 359 — Schematic Map – Network example *(Paris metro map from Antoine Raby http://checkmylabs.tumblr.com)*

7.7.17.1.2 Schematic Map Example – Station Layout

The following figures show examples of a SCHEMATIC MAP for Wimbledon station. Each of the elements on the map, such as QUAYs, ENTRANCEs, EQUIPMENT such as Lifts, Stairs, Barriers, etc. can be linked to a model element using a SCHEMATIC MAP.

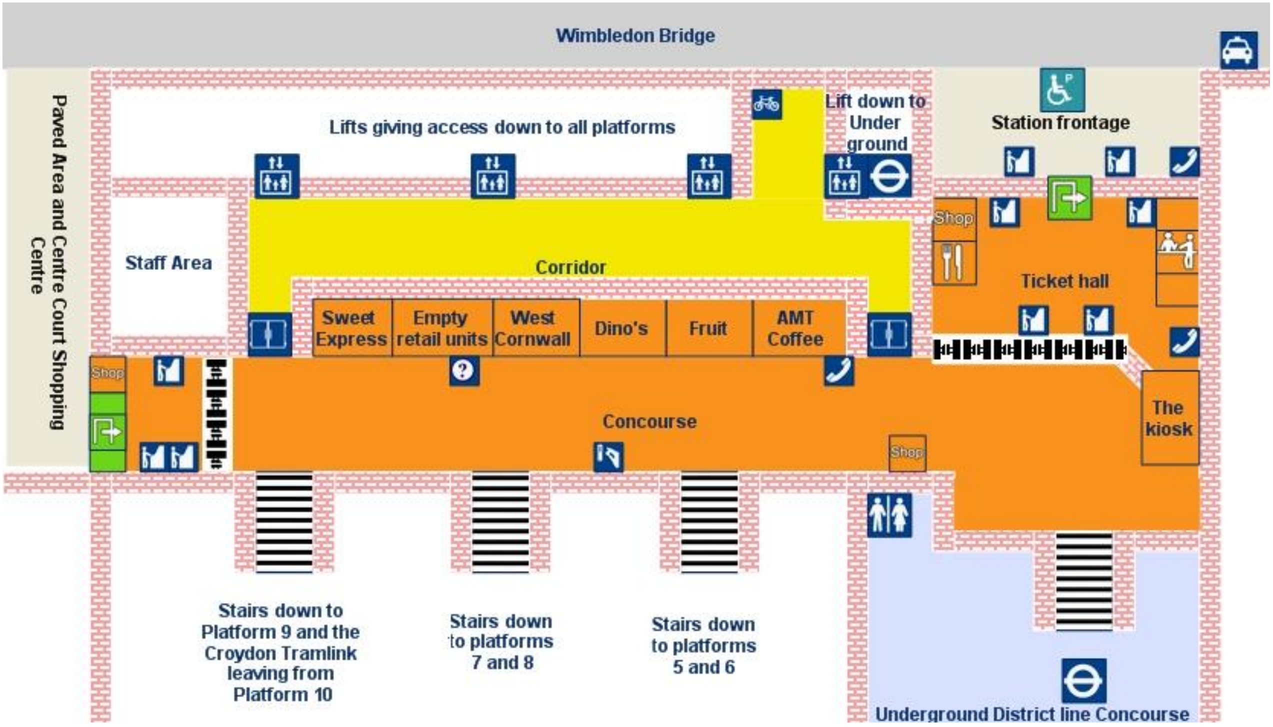


Figure 360 — Wimbledon Station plan: Ground floor (NRE Stations Made Easy)

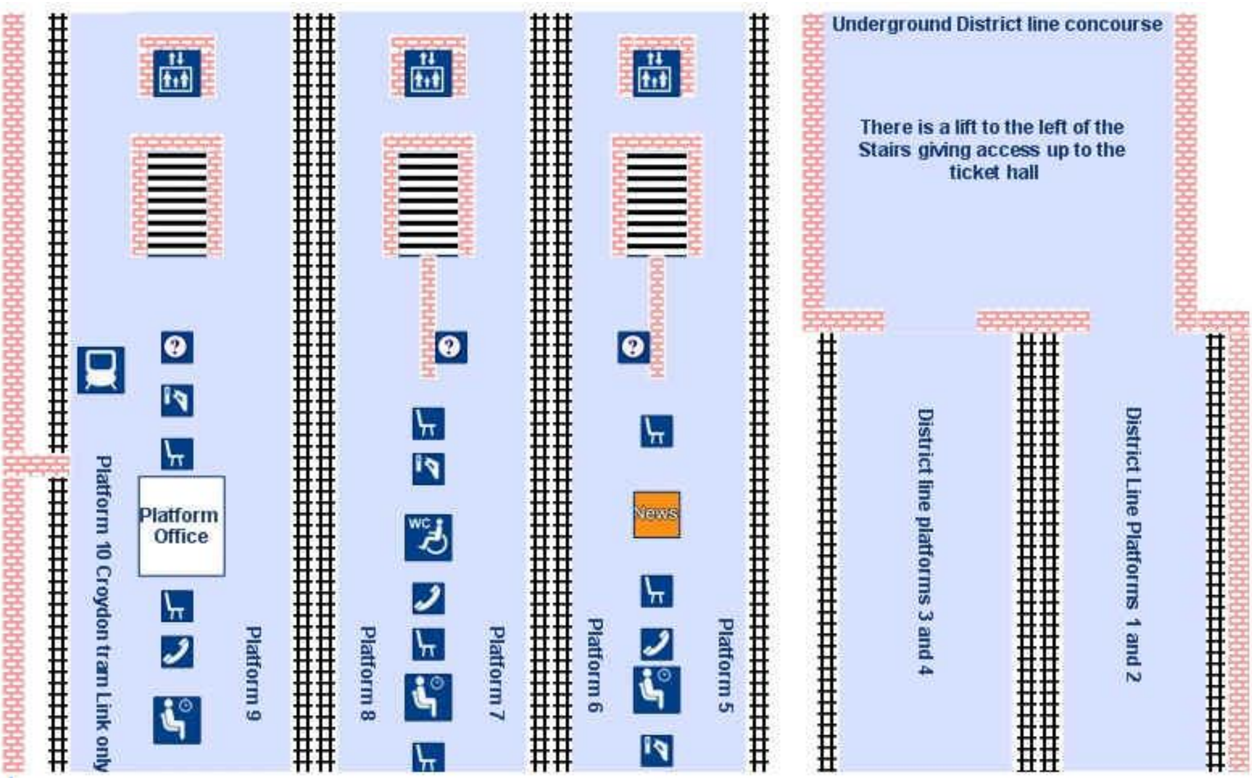


Figure 361 — Wimbledon Station plan: Lower floor (NRE Stations Made Easy)

7.7.17.2 Schematic Map – Physical Model

A SCHEMATIC MAP provides a set of references to model elements which can be organised into a spatial layout, linkable back to model elements.

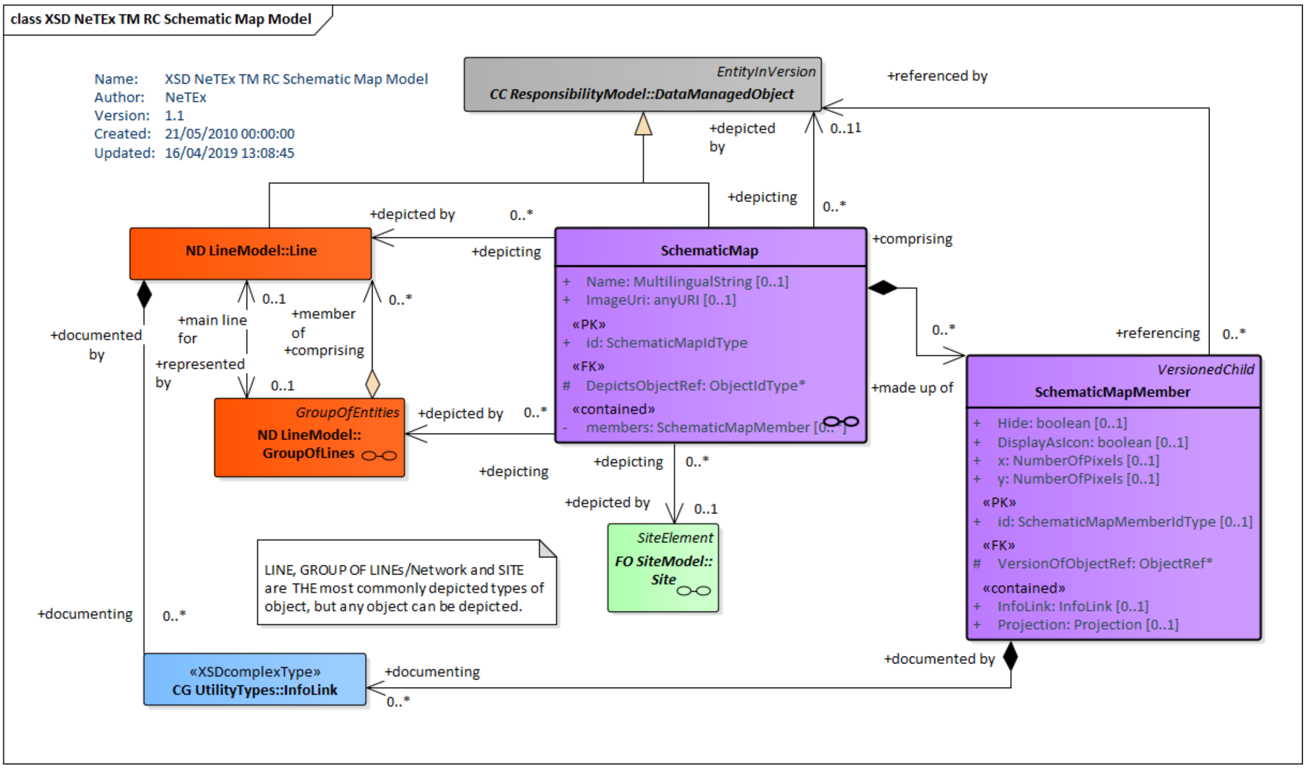


Figure 362 — Schematic Map – Physical Model (UML)

7.7.17.3 Schematic Map – Attributes and XSD

7.7.17.3.1 SchematicMap – Model Element

A map representing schematically the layout of the topographic structure of the public transport network (a set of LINEs) and a set of SITEs.

Can include a pixel projection of a set of ENTITies onto a bitmap image so as to support hyperlinked interactions.

Table 320 — *SchematicMap – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	SCHEMATIC MAP inherits from DATA MANAGED OBJECT.
«PK»	***id***	*SchematicMapIdType*	1:1	Identifier of SCHEMATIC MAP.
	Name	*MultilingualString*	0:1	Name of SCHEMATIC MAP.
	ShortName	*MultilingualString*	0:1	Short Name of SCHEMATIC MAP.
	ImageUri	*xsd:anyURI*	0:1	URL associated with SCHEMATIC MAP.
«FK»	***DepictsObjectRef***	*ObjectRef*	1:1	Main ENTITY depicted by SCHEMATIC MAP.
«cntd»	***members***	<u>*SchematicMapMember*</u>	0:*	ENTITies referenced by points on SCHEMATIC MAP.

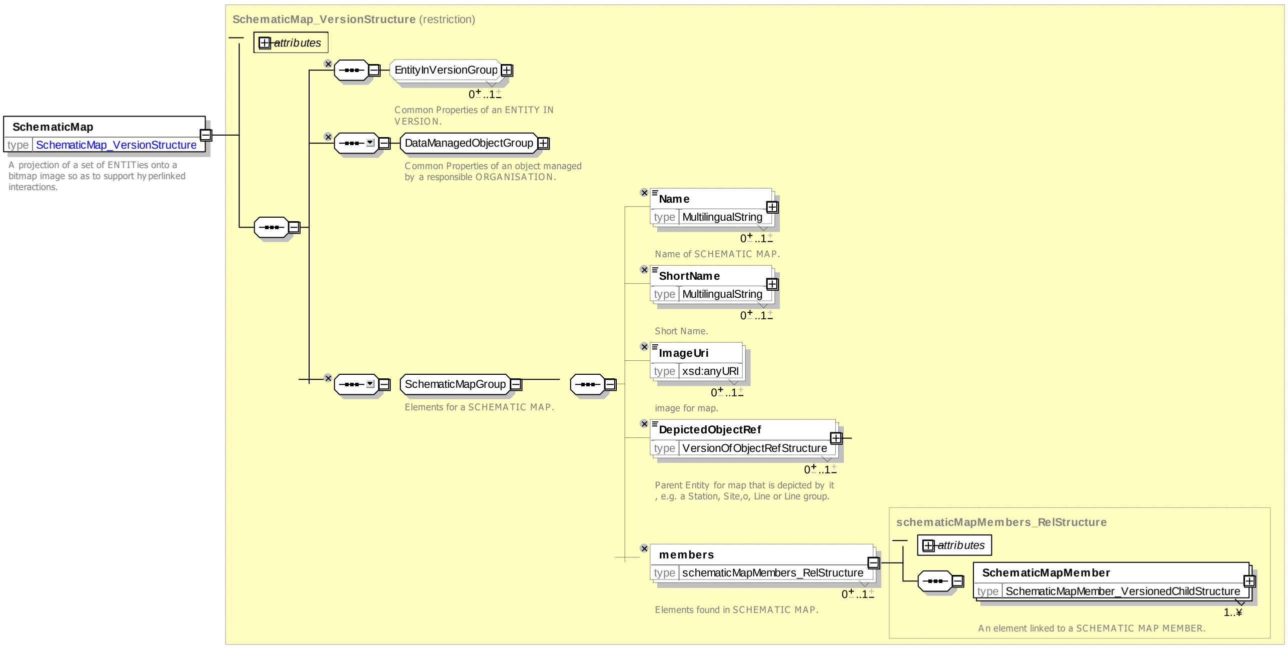


Figure 363 — *SchematicMap – XSD*

7.7.17.3.2 SchematicMapMember – Model Element

A projection of an individual ENTITY onto a SCHEMATIC MAP.

Table 321 — *SchematicMapMember – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	SCHEMATIC MAP MEMBER inherits from VERSIONED CHILD.
«PK»	***id***	*SchematicMapMember-IdType*	0:1	Identifier of a member of a SCHEMATIC MAP.
	Name	*MultilingualString*	0:1	Name of a member of a SCHEMATIC MAP.
«FK»	***VersionOfObjectRef***	*ObjectRef*	1:1	Reference to element on map.
	Hide	*xsd:boolean*	0:1	Whether element is visible on map.
«FK»	***DisplayAsIcon***	*xsd:boolean*	1:1	Whether the element is to be as an icon.
«cntd»	***InfoLink***	<u>*InfoLink*</u>	0:1	Info link associated with element.
	x	*NumberOfPixels*	0:1	X coordinate of element on SCHEMATIC MAP.
	y	*NumberOfPixels*	0:1	y coordinate of element on SCHEMATIC MAP.
«cntd»	***Projection***	<u>*Projection*</u>	0:1	PROJECTION of element onto a spatial element.

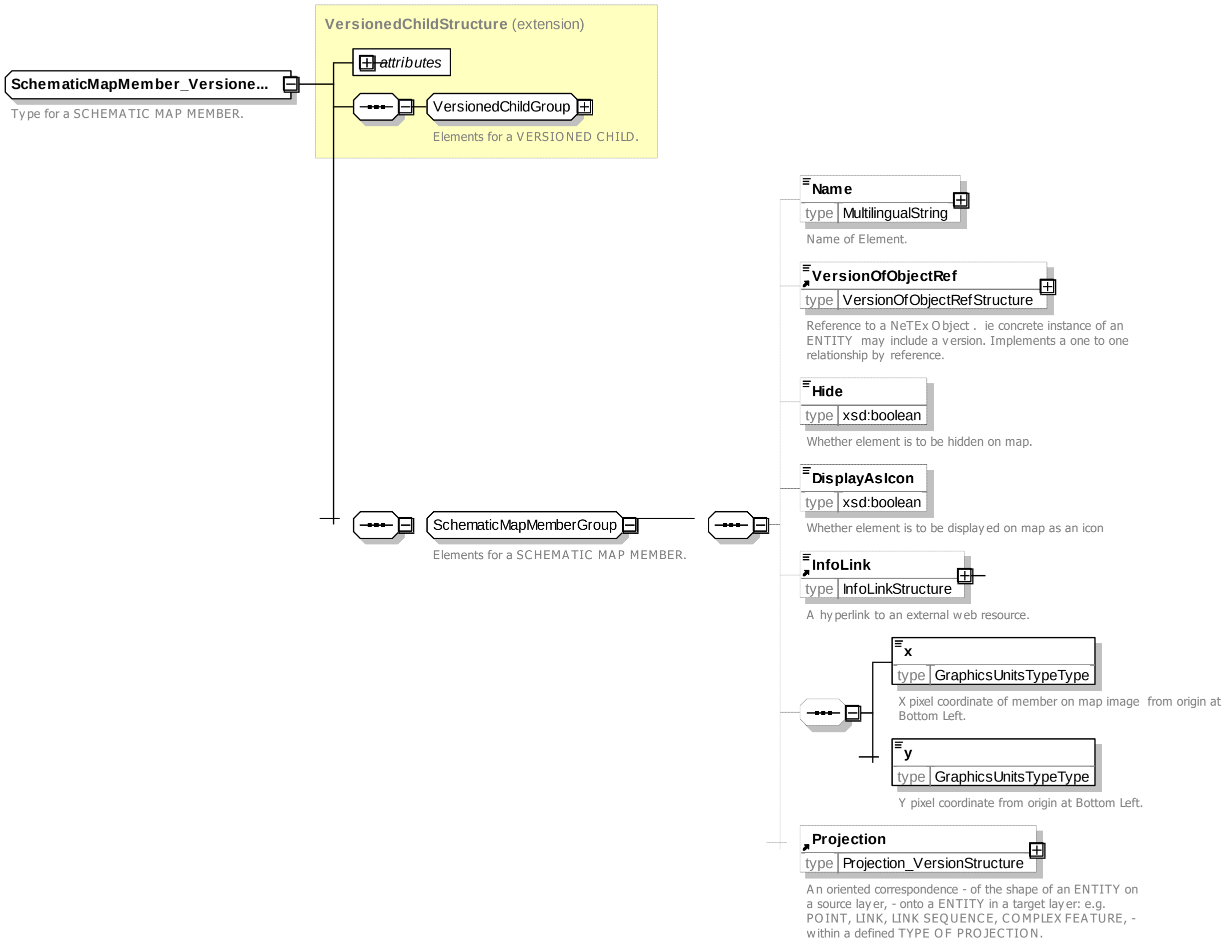


Figure 364 — *SchematicMapMember – XSD*

7.7.17.4 XML Examples of Schematic Maps

7.7.17.4.1 Station Map – XML Fragment Example

The following XML code fragment shows part of a ***SchematicMap*** for a rail station that identifies the pixel coordinates for two entrances (ENTRANCEs) and a toilet (i.e. SANITARY EQUIPMENT)**.**

EXAMPLE

```
    <SchematicMap version="001" created="2001-12-17T09:30:47Z" id="tbd::WimMap_001">
        <Name>Map of Wimbledon Station - Upper Level</Name>
        <ImageUri>http://www.tbde.com/Wimbledonplan1.jpg</ImageUri>
        <DepictedObjectRef nameOfRefClass="StopPlace" ref="napt::910GWIMBLDN"/>
        <members>
            <SchematicMapMember id="tbd::WimMap_001_01">
                <Name>Entrance</Name>
                <SchematicMapMemberRef  nameOfRefClass="StopPlaceEntrance"  ref="tbd::9100WIMBLDN_A1-
EE1"/>
                <x>0123</x>
                <y>346</y>
            </SchematicMapMember>
            <SchematicMapMember id="tbd::WimMap_001_02">
                <Name>Side Entrance</Name>
                <SchematicMapMemberRef  nameOfRefClass="StopPlaceEntrance"  ref="tbd::9100WIMBLDN_A4-
EE1"/>
                <x>014</x>
                <y>344</y>
            </SchematicMapMember>
            <SchematicMapMember id="tbd::WimMap_001_03">
                <Name>Men and Women's Toilets</Name>
```

```
                <SchematicMapMemberRef  nameOfRefClass="SanitaryEquipment"  ref="tbd::9100WIMBLDN_A2-
Eq-WC1"/>
                <x>0145</x>
                <y>365</y>
            </SchematicMapMember>
        </members>
    </SchematicMap>
```

7.7.18 Notice

7.7.18.1 NOTICE – Conceptual MODEL

The NOTICE Model defines reusable text note elements that may be attached to timetables as footnotes, used as announcements, etc. NOTICEs are associated with other entities using a NOTICE ASSIGNMENT. NOTICES may be classified with a TYPE OF NOTICE.

Each NOTICE may have several alternative formats as specified by a DELIVERY VARIANT. The variant may be for use on a specific media (Printed, displayed, sms, spoken, etc).

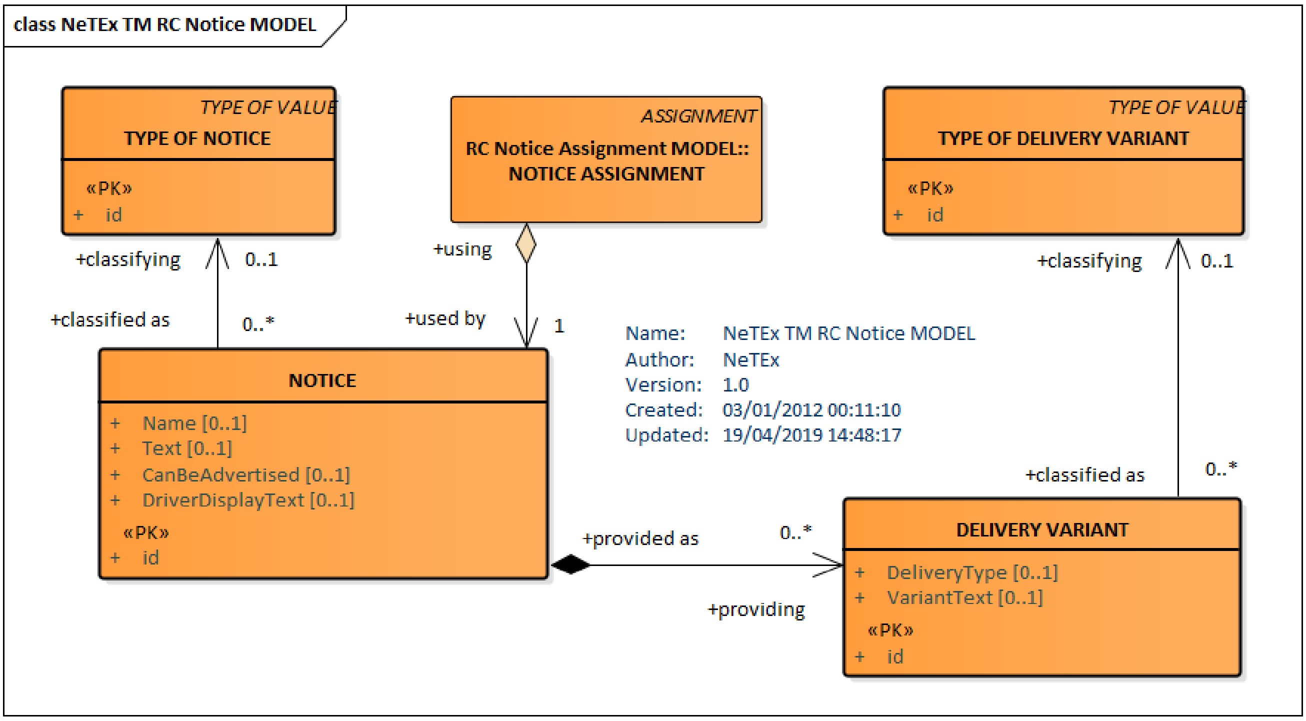


Figure 365 — Notice – Conceptual MODEL (UML)

7.7.18.2 Notice – Physical Model

The following figure shows the Physical model for NOTICEs. The content of a NOTICE is of type a *MultilingualString*, thus enabling the text of the NOTICE to be made available in several languages using the ALTERNATIVE TEXTs mechanism..

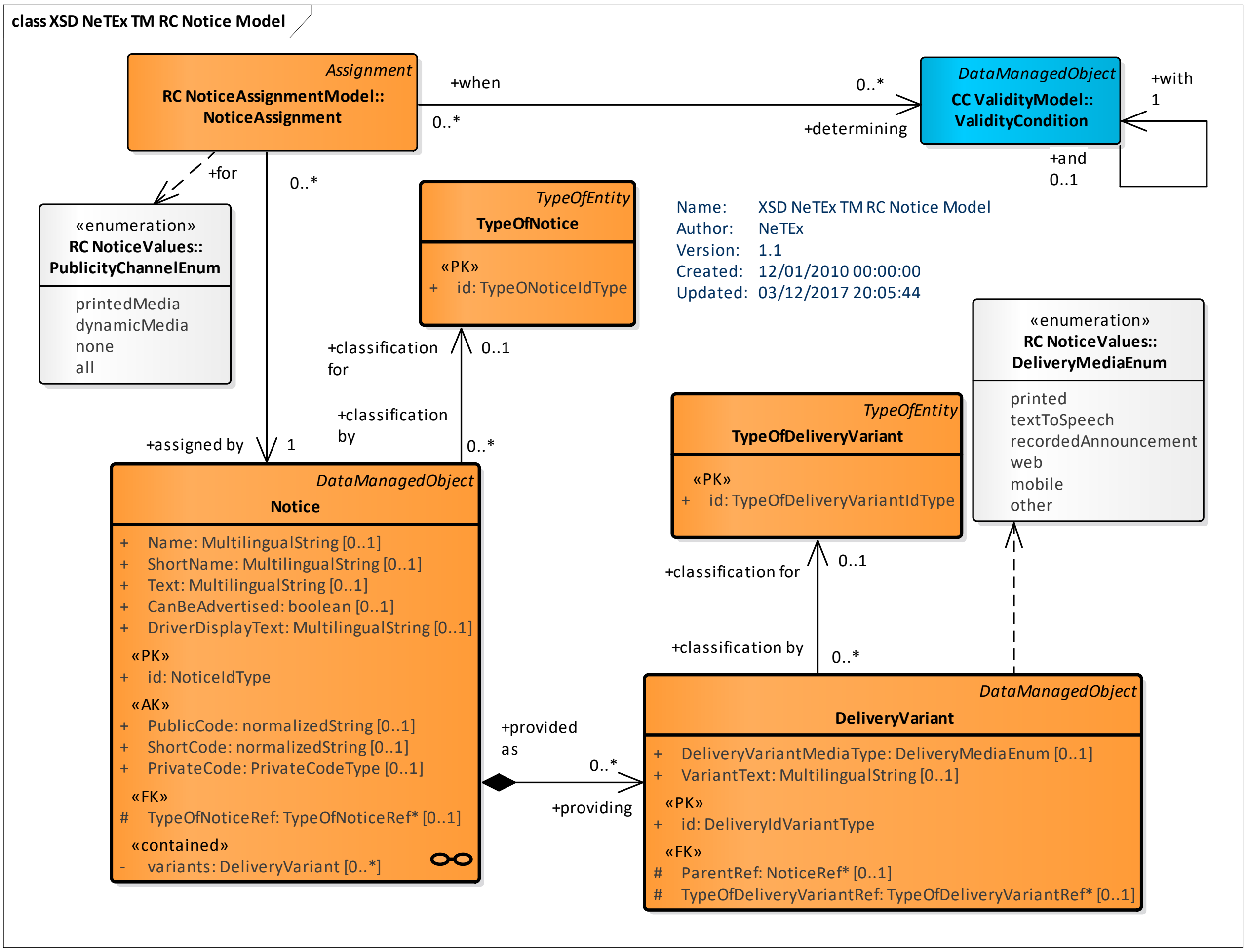


Figure 366 — Notice – Physical Model (UML)

7.7.18.2.1 Use of Notices

A NOTICE carries additional information for passengers that may help them planning their travel or during their travel. Footnotes are one type of NOTICE, a human-understandable text that may be made available in various delivery formats (print, internet, mobile media, etc). Using a NOTICE ASSIGNMENT, NOTICEs can be assigned to stops of one service (POINT IN JOURNEY PATTERN), to stops in more services (COMMON SECTION), to JOURNEY PATTERNs, to one VEHICLE JOURNEY or more of them (GROUP OF SERVICEs) and to INTERCHANGEs. Each NOTICE ASSIGNMENT can be restricted in its validity by specifying a VALIDITY CONDITION.

The following figure shows the use of NOTICEs with some other element types.

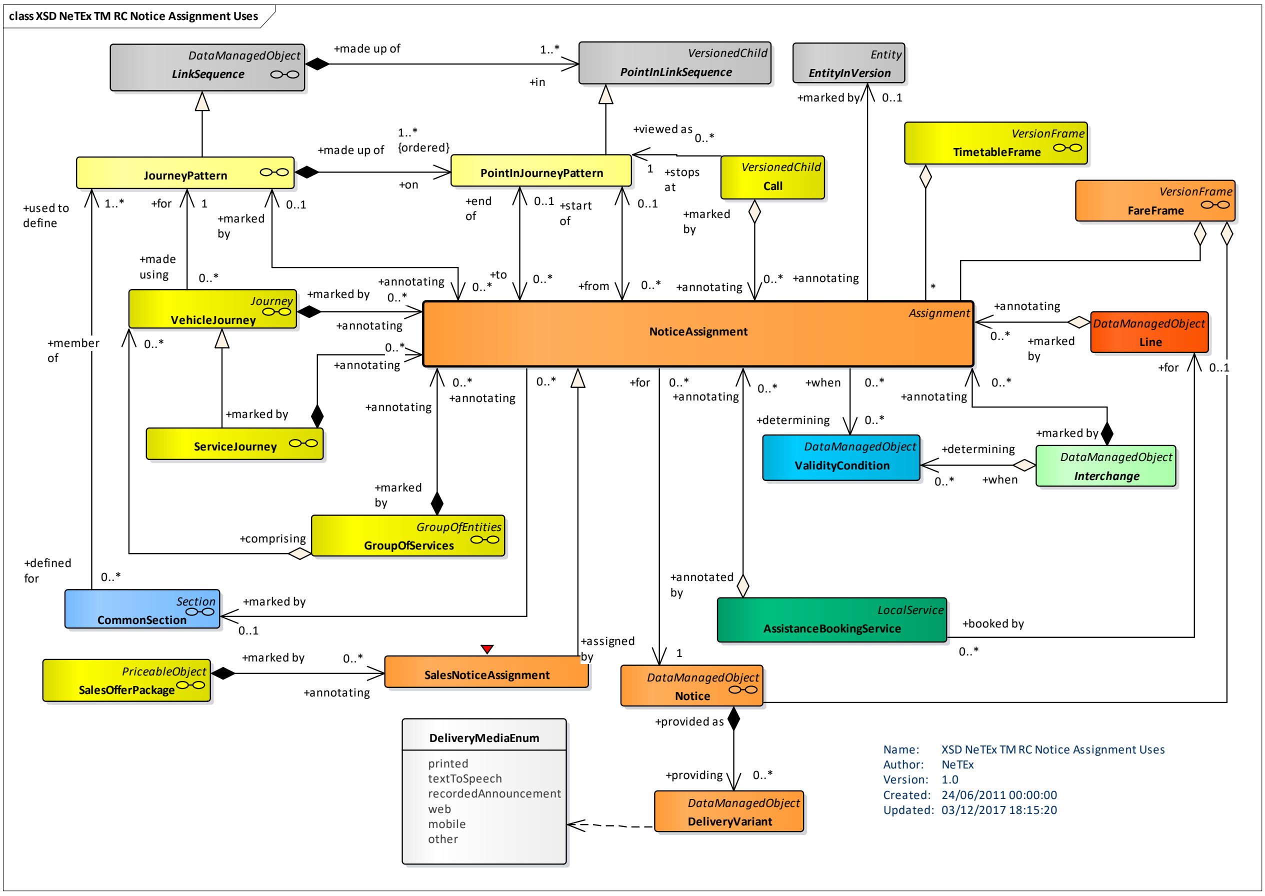


Figure 367 — Notice Usage – Physical Model (UML)

7.7.18.3 Notice – Attributes and XSD

7.7.18.3.1 Notice – Model Element

A text for informational purposes on exceptions in a LINE, a JOURNEY PATTERN, etc. The information may be useful for passenger or driver information.

Table 322 —*Notice – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	NOTICE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*NoticeIdType*	1:1	Identifier of NOTICE.
	Name	*MultilingualString*	0:1	Name of NOTICE.
	ShortName	*MultilingualString*	0:1	Short Name of NOTICE. Use Text for content. +V1.1.
	Text	*MultilingualString*	0:1	Content text for NOTICE.
«AK»	***PublicCode***	*xsd:normalizedString*	1:1	Private code for NOTICE.
«AK»	***ShortCode***	*xsd:normalizedString*	1:1	Short code for NOTICE.

«AK»	***PrivateCode***	*PrivateCode*	1:1	Public Code for NOTICE.
«FK»	***TypeOfNoticeRef***	*TypeOfNoticeRef*	1:1	Reference to TYPE OF NOTICE.
	CanBeAdvertised	*xsd:boolean*	0:1	Whether NOTICE is advertised.
	DriverDisplayText	*MultilingualString*	0:1	Driver Display text associated with NOTICE.
«cntd»	***variants***	*DeliveryVariant*	0:*	DELIVERY VARIANTs for NOTICE.

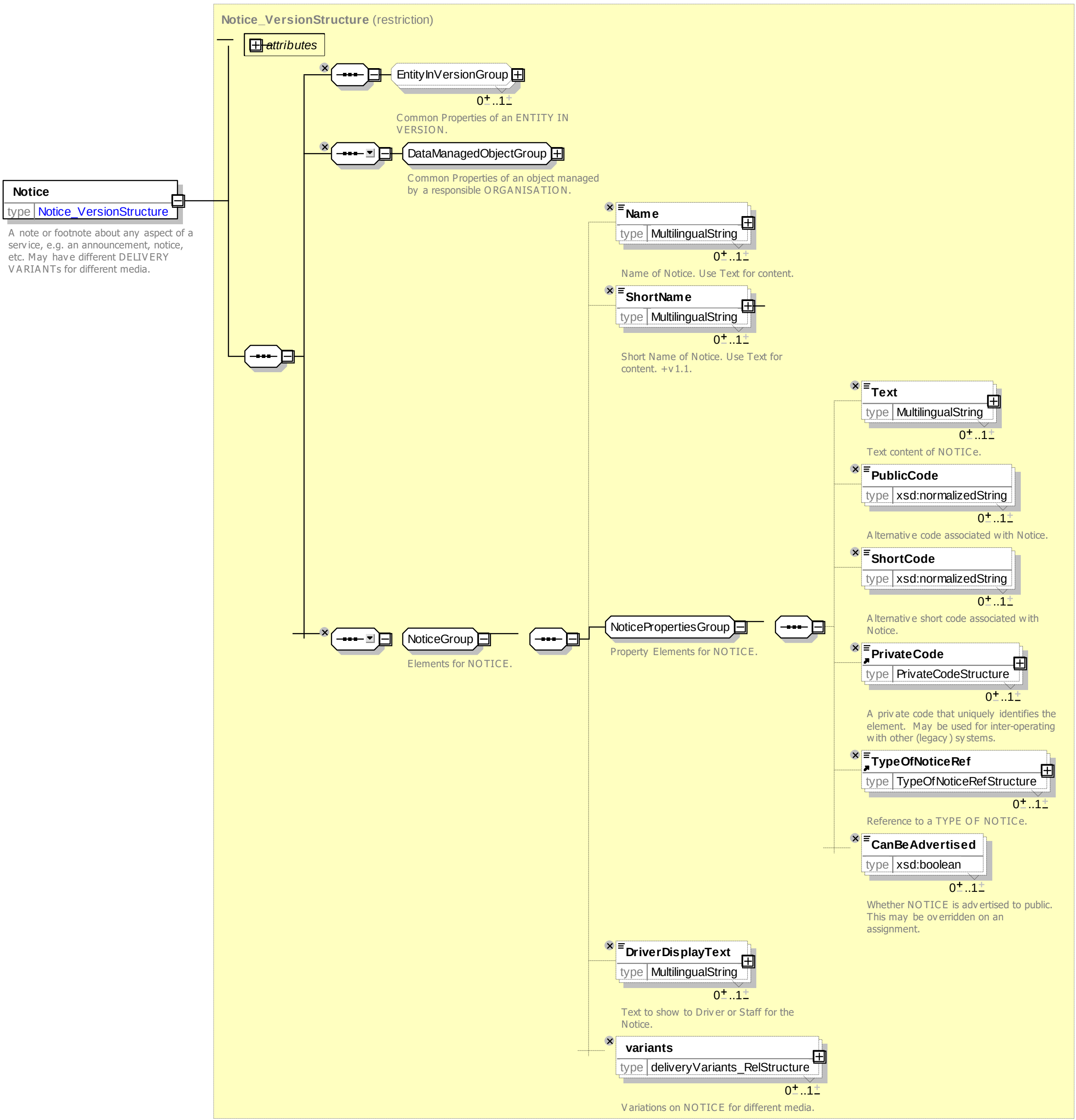


Figure 368 — *Notice – XSD*

7.7.18.3.2 DeliveryVariant – Model Element

A variant text of a NOTICE for use in a specific media or delivery channel (voice, printed material, etc.).

Table 323 —*DeliveryVariant – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	DELIVERY VARIANT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*DeliveryVariantIdType*	1:1	Identifier for DELIVERY VARIANT.
«FK»	***ParentRef***	*NoticeRef*	0:1	Reference to a NOTICE.
«enum»	***DeliveryVariant-MediaType***	*DeliveryMediaEnum*	1:1	Way a NOTICE is delivered, characterized by the delivery medium (voice, printed material, etc.). See allowed values below.
«FK»	***TypeOfDelivery-VariantRef***	*TypeOfDeliveryVariantRef*	0:1	Reference to a classification of DELIVERY VARIANT.
	VariantText	*MultilingualString*	0:1	Text for DELIVERY VARIANT.

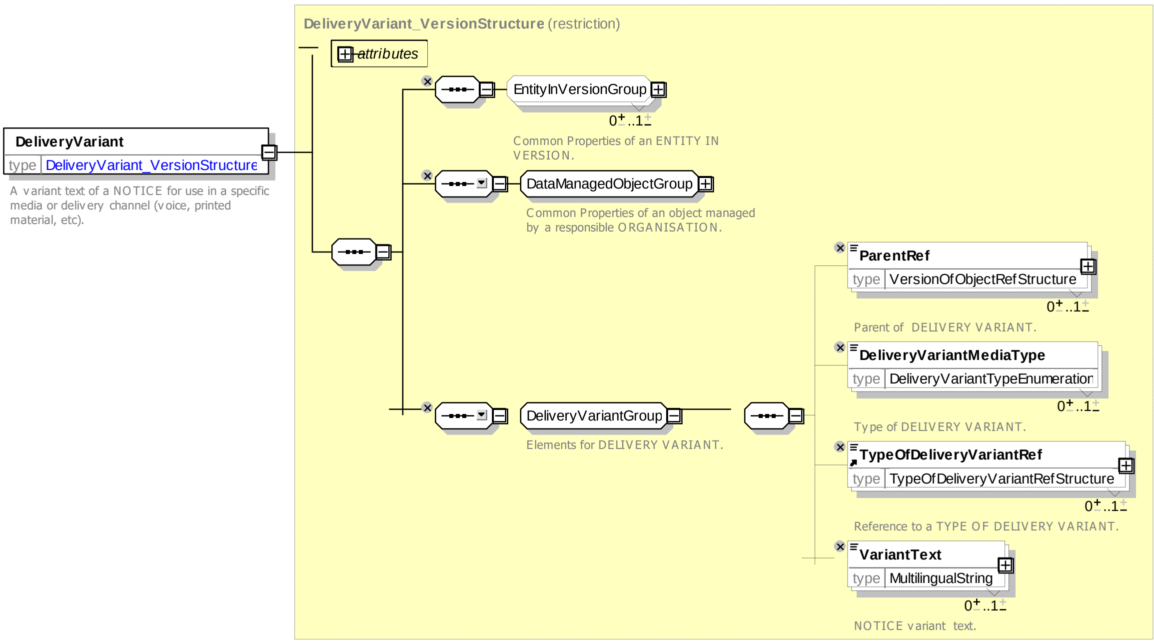


Figure 369 —*DeliveryVariant – XSD*

7.7.18.3.2.1 PublicityChannel – Allowed Values

Allowed values for ***PublicityChannel*** *(PublicityChannelEnumeration).*

Table 324 – *PublicityChannel* – Allowed Values

Value	Description
printedMedia	*Printed media channels*
dynamicMedia	*Dynamic media channels*
none	*No channels*
all	*All channels*

7.7.18.3.3 TypeOfNotice – Model Element

A classification for a NOTICE.

Table 325 — *TypeOfNotice – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF NOTICE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeONoticeIdType*	1:1	Identifier of TYPE OF NOTICE.

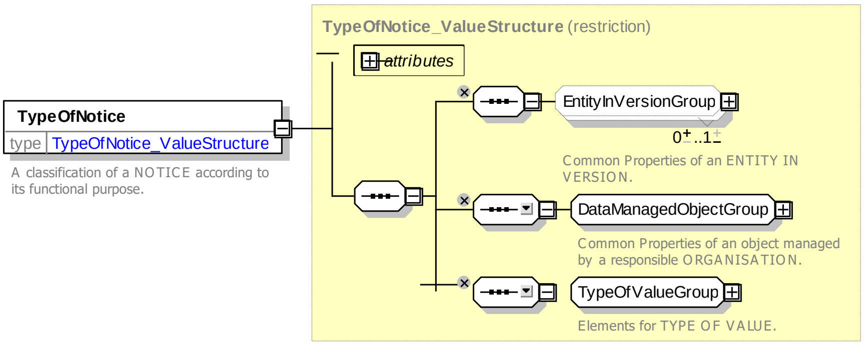


Figure 370 — *TypeOfNotice – XSD*

7.7.18.3.4 TypeOfDeliveryVariant – Model Element

A classification of a DELIVERY VARIANT corresponding to the way of delivering a NOTICE: by vocal announcement, by visual display, issuing printed material.

Table 326 — *TypeOfDeliveryVariant – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF DELIVERY VARIANT inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfDeliveryVariantIdType*	1:1	Identifier of TYPE OF DELIVERY VARIANT.

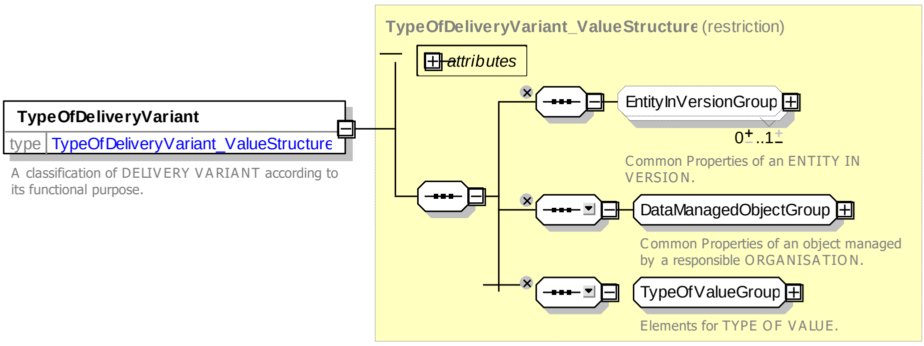


Figure 371 — *TypeOfDeliveryVariant – XSD*

7.7.18.4 Notice Assignment – Physical Model

The following figure shows the Physical model for NOTICE.

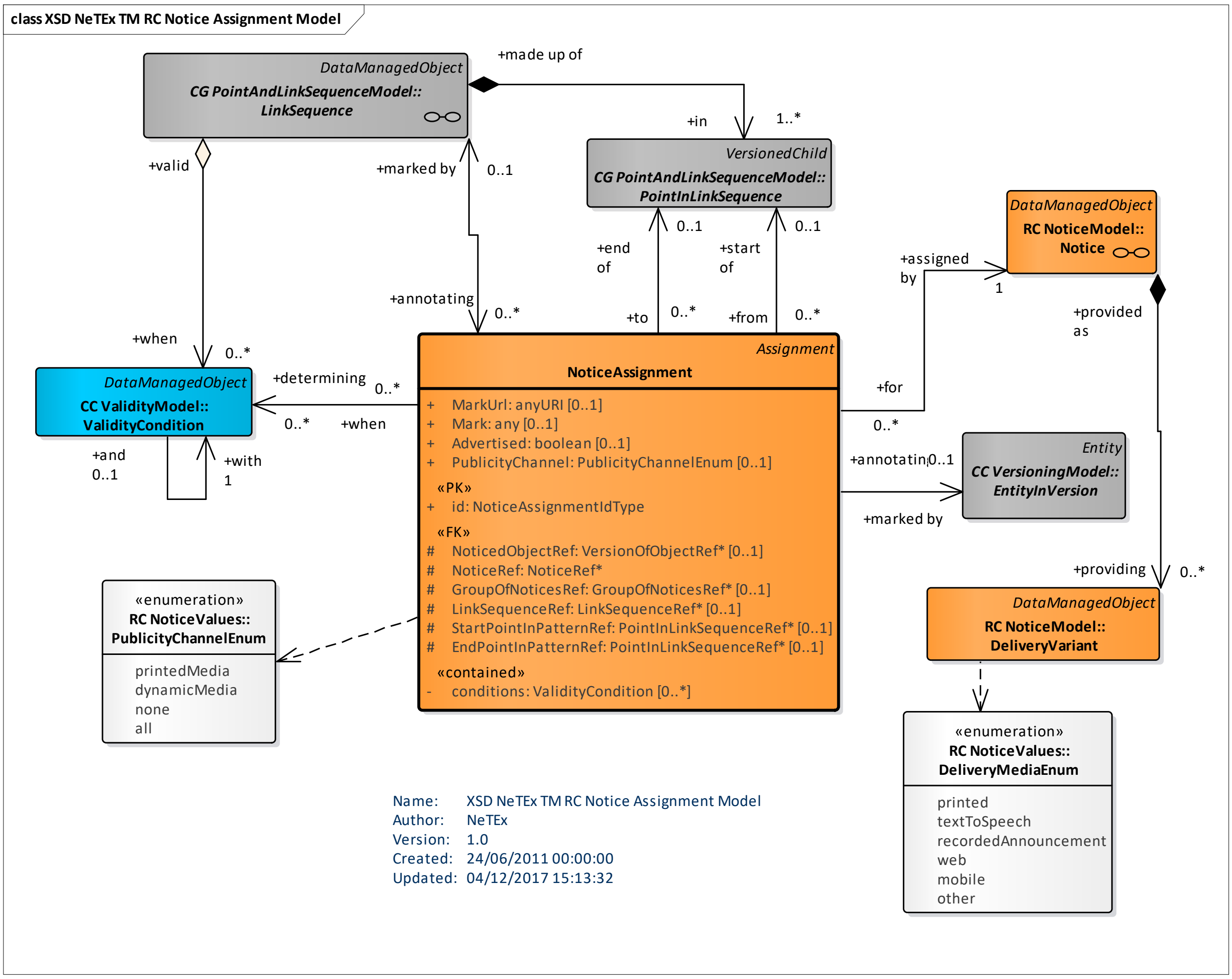


Figure 372 — Notice Assignment – Physical Model (UML)

7.7.18.5 NoticeAssignment – Attributes and XSD

7.7.18.5.1.1 NoticeAssignment – Model Element

The assignment of a NOTICE to any model element. Can be used in particular to show an exception in a JOURNEY PATTERN, a COMMON SECTION, or a VEHICLE JOURNEY, possibly specifying at which POINT IN JOURNEY PATTERN the validity of the NOTICE starts and ends respectively.

Table 327 — NoticeAssignment – *Element*

Classification	Name		Type	Cardinality	Description
::>	::>		Assignment	::>	NOTICE ASSIGNMENT inherits from ASSIGNMENT.
«PK»	***id***		*TypeOfNoticeAssignment IdType*	1:1	Identifier of NOTICE ASSIGNMENT.
«FK»	***a***	***NoticeRef***	*NoticeRef*	0:1	Reference to a NOTICE i.e. footnote, note, announcement or other informational text element.
«FK»	***b***	***GroupOfNotices-Ref***	*GeneralGroupOf-EntitiesRef*	0:1	A group of Notices.
«cntd»	***c***	***Notice***	*Notice*	0:1	A note or footnote about any aspect of a service, e.g. an announcement, notice, etc. May have different DELIVERY VARIANTs for different media.
«FK»	***NoticedObjectRef***		*(VersionOfObjectRef)*	0:1	Object with which NOTICE is associated. If given by context can be omitted.
«FK»	***LinkSequenceRef***		*(LinkSequenceRef)*	0:1	Reference to a LINK SEQUENCE.
«FK»	***SectionRef***		*(SectionRef)*	0:1	Reference to a SECTION.
«FK»	***StartPointInPattern-Ref***		*PointInSequenceRef*	0:1	POINT IN SEQUENCE at which applicability of NOTICE starts.
«FK»	***EndPointInPatternRef***		*PointInSequenceRef*	0:1	POINT IN SEQUENCE at which applicabiity of NOTICE ends. If absent same as Start Point.
	Mark		*xsd:string*	0:1	Mark associated with NOTICE.
	MarkUrl		*xsd:anyURI*	0:1	URL for image associated with NOTICE.
«enum»	***PublicityChannel***		*Publicity ChannelEnum*	0:1	How NOTICE is to be publicised. The default is '*all*'. See allowed values below.
	Advertised		*xsd:boolean*	0:1	Whether NOTICE is advertised to public.

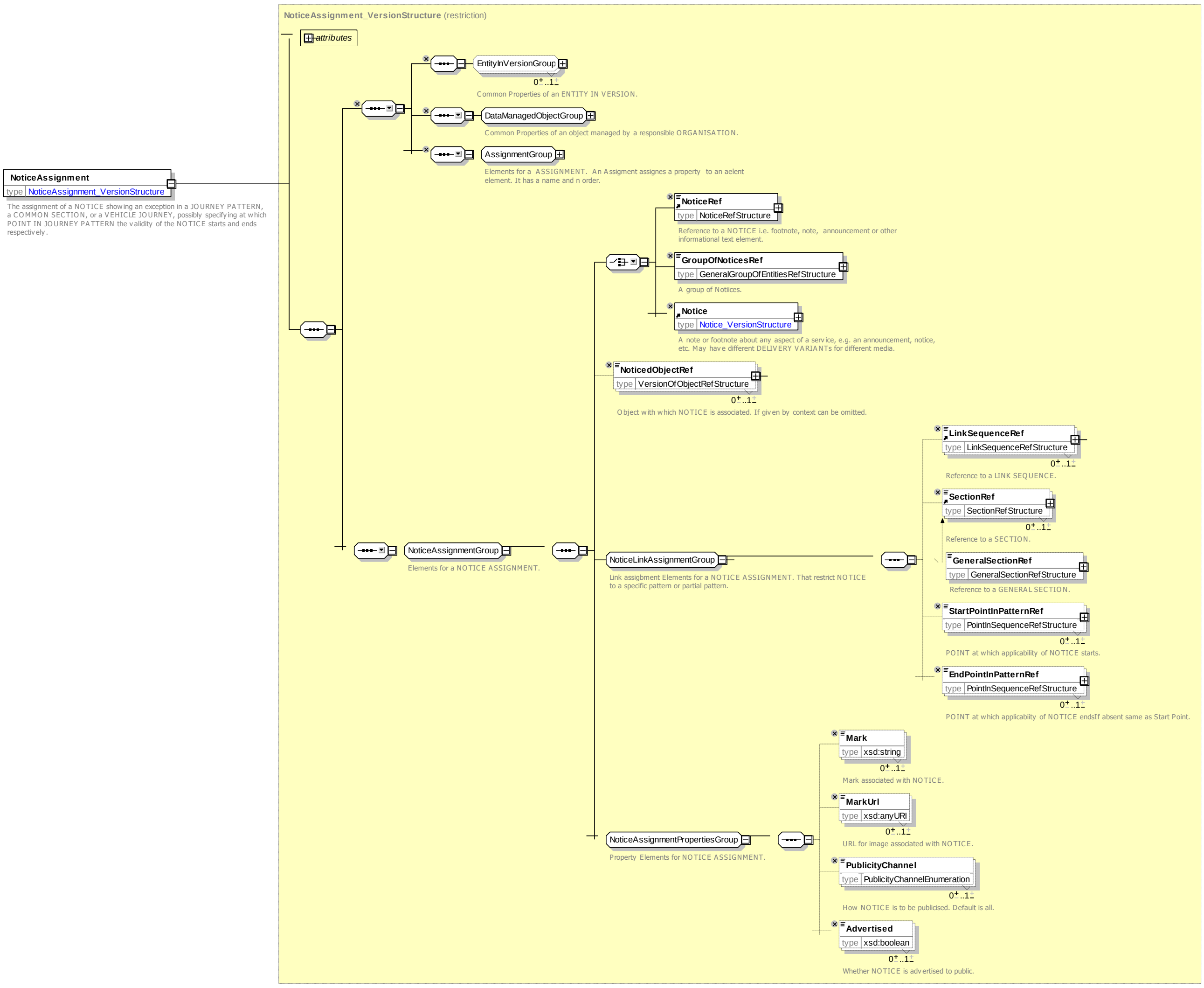


Figure 373 — NoticeAssignment – *XSD*

DeliveryVariantMediaType – Allowed Values

Allowed values for DELIVERY MEDIA (*DeliveryMediaEnumeration)*

Table 328 — DeliveryVariantMediaType – *AllowedValues*

Value	Description
printed	Media is printed
textToSpeech	Media is text to speech
recordedAnnouncement	Media is recorded announcement
web	Media is web page
mobile	Media is mobile device
other	Other media

7.7.18.6 XML Examples of Notices

7.7.18.6.1 Notices & Variants – XML Fragment Example

The following XML fragment shows uses of NOTICEs to specify three footnotes for a timetable. There are shorter DELIVERY VARIANTs to use in a mobile context.

EXAMPLE

```
    <notices>
        <Notice version="any" id="hde:sj_46o_01">
            <Text>Only runs during termtimes of Dothebys Hall</Text>
            <variants>
                <DeliveryVariant version="any" id="hde:sj_46_01">
                    <DeliveryVariantMediaType>mobile</DeliveryVariantMediaType>
                    <VariantText>Termtimes Only</VariantText>
                </DeliveryVariant>
            </variants>
        </Notice>
        <Notice version="any" id="hde:sj_46_021">
            <Text>Stops at Q only during term times of Dothebys Hall</Text>
            <variants>
                <DeliveryVariant version="any" id="hde:sj_46_01">
                    <DeliveryVariantMediaType>mobile</DeliveryVariantMediaType>
                    <VariantText>Termtimes Only</VariantText>
                </DeliveryVariant>
            </variants>
        </Notice>
        <Notice version="any" id="hde:Notice:sj_24o_02">
            <Text>Can connect to LINE 4</Text>
            <variants>
                <DeliveryVariant version="any" id="hde::sj_24o_02_01">
                    <DeliveryVariantMediaType>textToSpeech</DeliveryVariantMediaType>
                    <VariantText>At this stop it is possible to change to line 4 for Dotheboys academy
and Stop R</VariantText>
                </DeliveryVariant>
                <DeliveryVariant version="any" id="hde::sj_24o_02_02">
                    <DeliveryVariantMediaType>mobile</DeliveryVariantMediaType>
                    <VariantText>X Line 4</VariantText>
                </DeliveryVariant>
            </variants>
        </Notice>
    </notices>
```

7.7.18.6.2 Stop Announcements & Variants – XML Fragment Example

The following XML fragment shows uses of NOTICE & DELIVERY VARIANT to specify the contents of stop announcements in English & German.

EXAMPLE

```
        <Notice version="any" id="mybus:Nxa_SSP_001">
            <Name>Next Stop Announcement for SSP_001</Name>
            <Text lang="en">Welcome This service goes to Charley Crescent. Stopping everywhere</Text>
            <variants>
                <DeliveryVariant version="any" id="mybus:Nxa_SSP_001_en">
                    <DeliveryVariantMediaType>textToSpeech</DeliveryVariantMediaType>
                    <VariantText     lang="de">Welcome     This     service     goes     to     Charley
Crescent.</VariantText>
                </DeliveryVariant>
                <DeliveryVariant version="any" id="mybus:Nxa_SSP_001_de">
                    <DeliveryVariantMediaType>textToSpeech</DeliveryVariantMediaType>
                    <VariantText    lang="de">Willkommen    Dieser    Dienst    fahrt    nach    Charley
Crescent.</VariantText>
                </DeliveryVariant>
            </variants>
        </Notice>
```

7.7.19 Security List

7.7.19.1 SECURITY LIST – Conceptual MODEL

The Security List Model defines lists of ENTITies marked as explicitly valid (WHITE LIST) or as under suspicion or blocked (BLACK LIST). The presence of an ENTITY on a SECURITY LIST is represented by a SECURITY LISTABLE that holds a reference to the ENTITY.

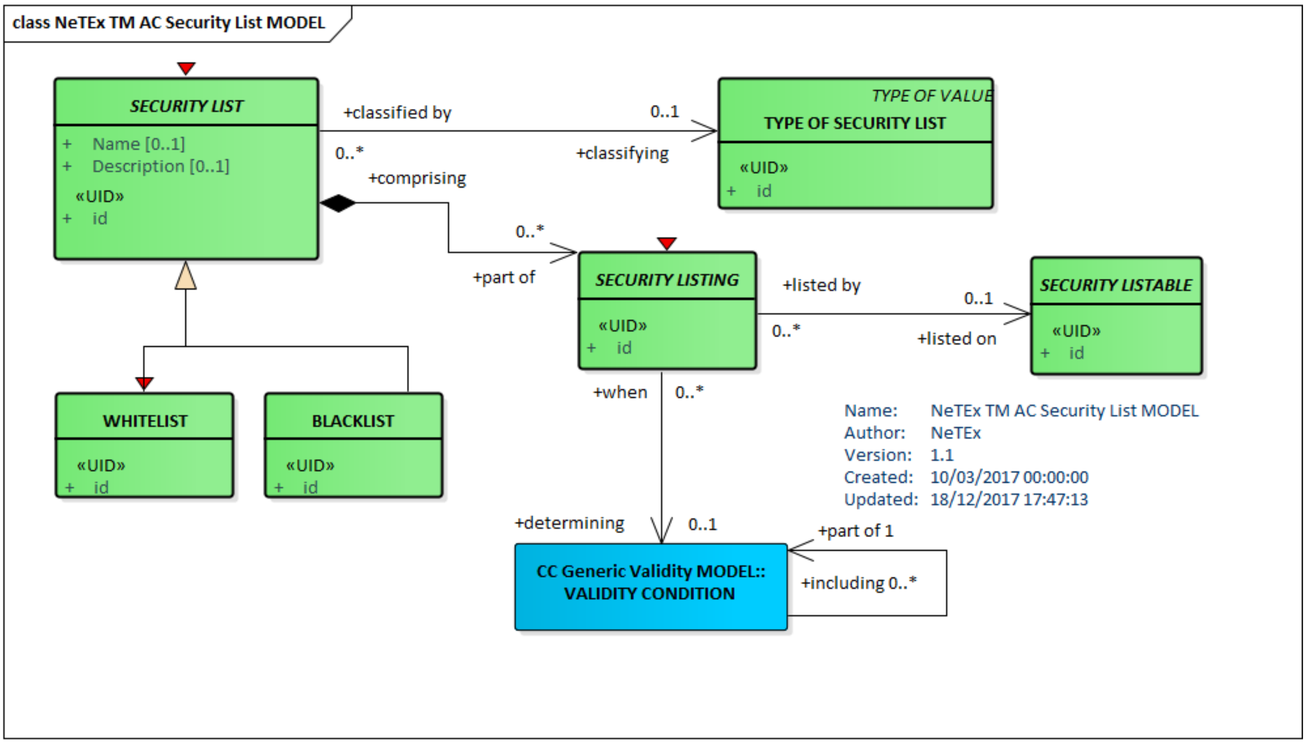


Figure 374 — Security list – Conceptual MODEL (UML)

7.7.19.2 Security List – Physical Model

The following figure shows the Physical model for SECURITY LISTs..

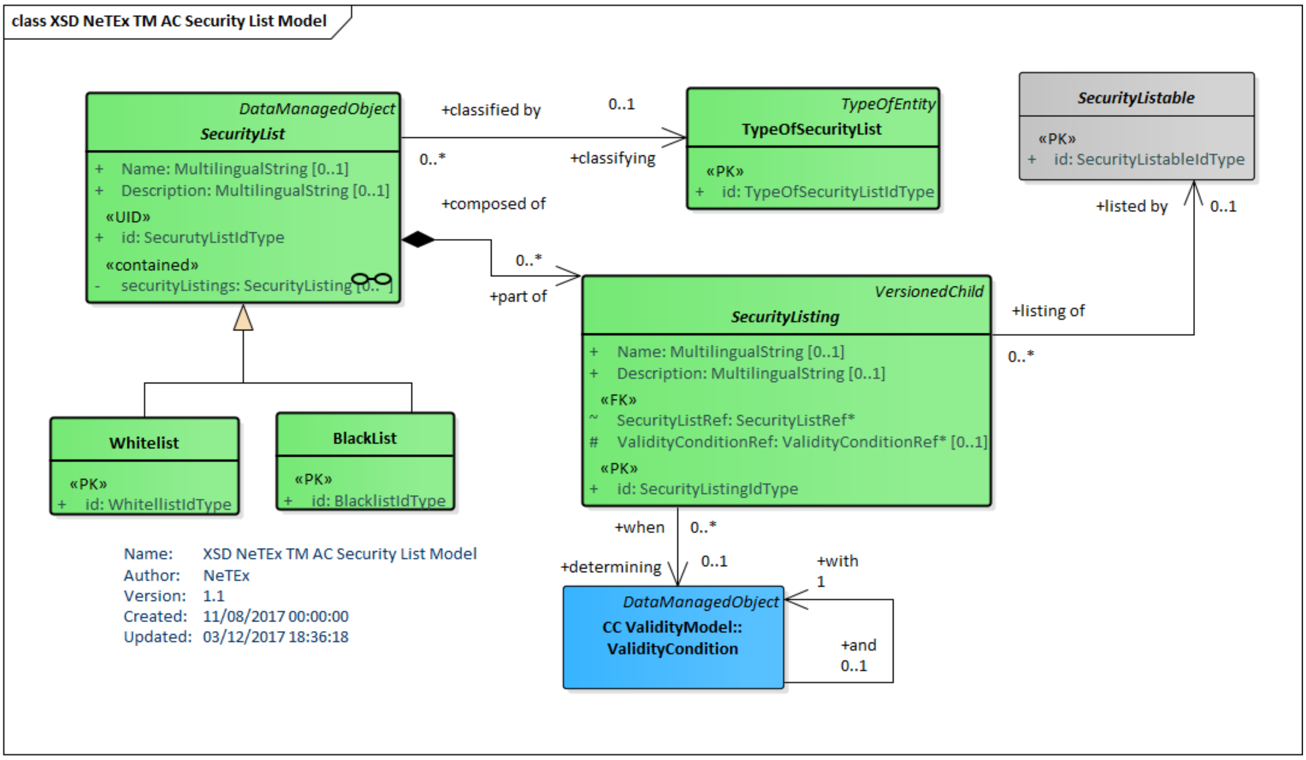


Figure 375 — Security list – Physical Model (UML)

7.7.19.2.1 Use of Security lists

The main use of SECURITY LISTs is for revenue protection in fare collection and control. They can also be used for example for access security.

There are specialisations of SECURITY LISTING for specific entities, for example CUSTOMER SECURITY LISTING, FARE CONTRACT SECURITY LISTING, RETAIL DEVICE SECURITY LISTING, etc etc.

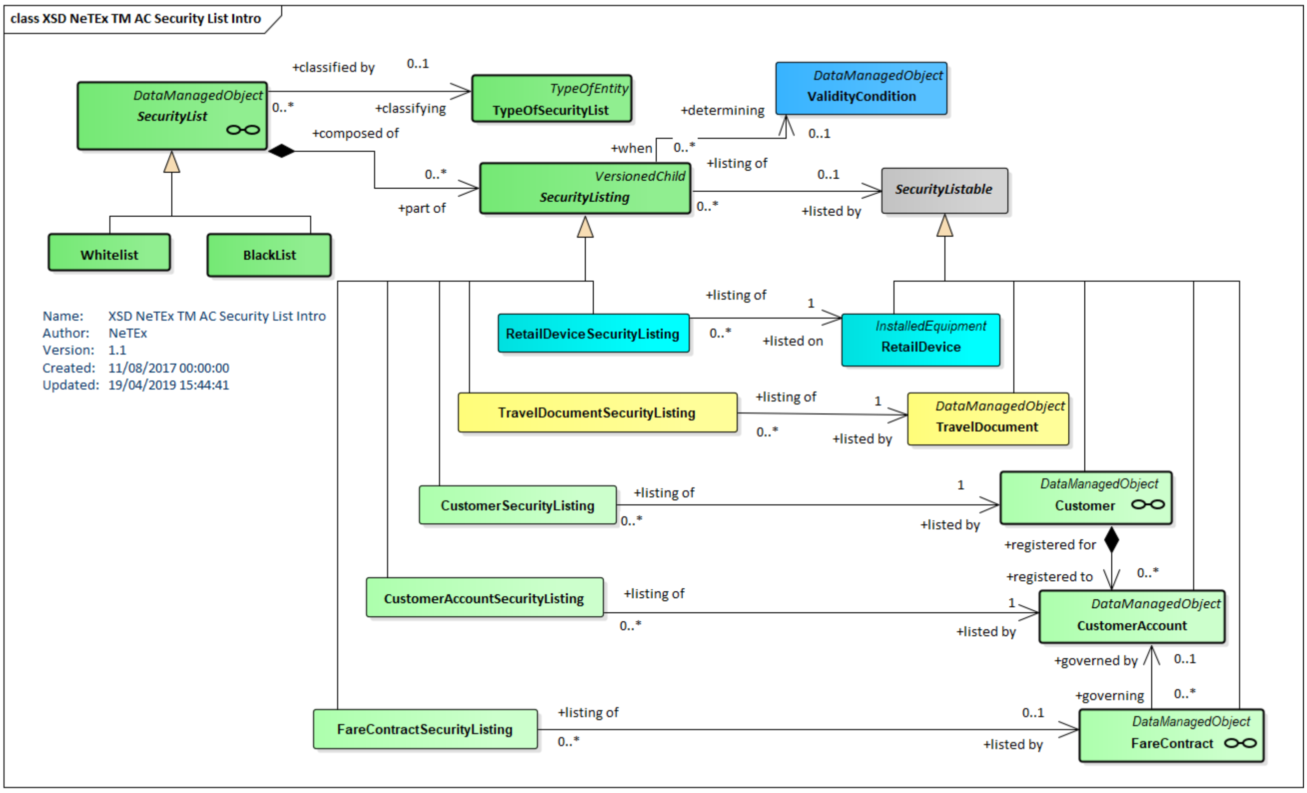


Figure 376 — Security list Usage – Physical Model (UML)

7.7.19.3 Security List – Attributes and XSD

7.7.19.3.1 SecurityList – Model Element

A list of items whose status is to be accepted or denied for a process such as purchase or validation.

Table 329 —*Security list – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	SECURITY LIST inherits from DATA MANAGED OBJECT.
«PK»	***id***	*SecurityListIdType*	1:1	Identifier of SECURITY LIST.
	Name	*MultilingualString*	0:1	Name of SECURITY LIST.
	Description	*MultilingualString*	0:1	Description of SECURITY LIST.
«FK»	***TypeOfSecurity-ListRef***	*TypeOfSecurityListRef*	1:1	Reference to TYPE OF SECURITY LIST.
«FK»	***OrganisationRef***	*OrganisationRef*	1:1	ORGANISATION managing SECURITY LIST.
«cntd»	***securityListings***	*(SecurityListing)*	0:*	Members of the SECURITY LIST..

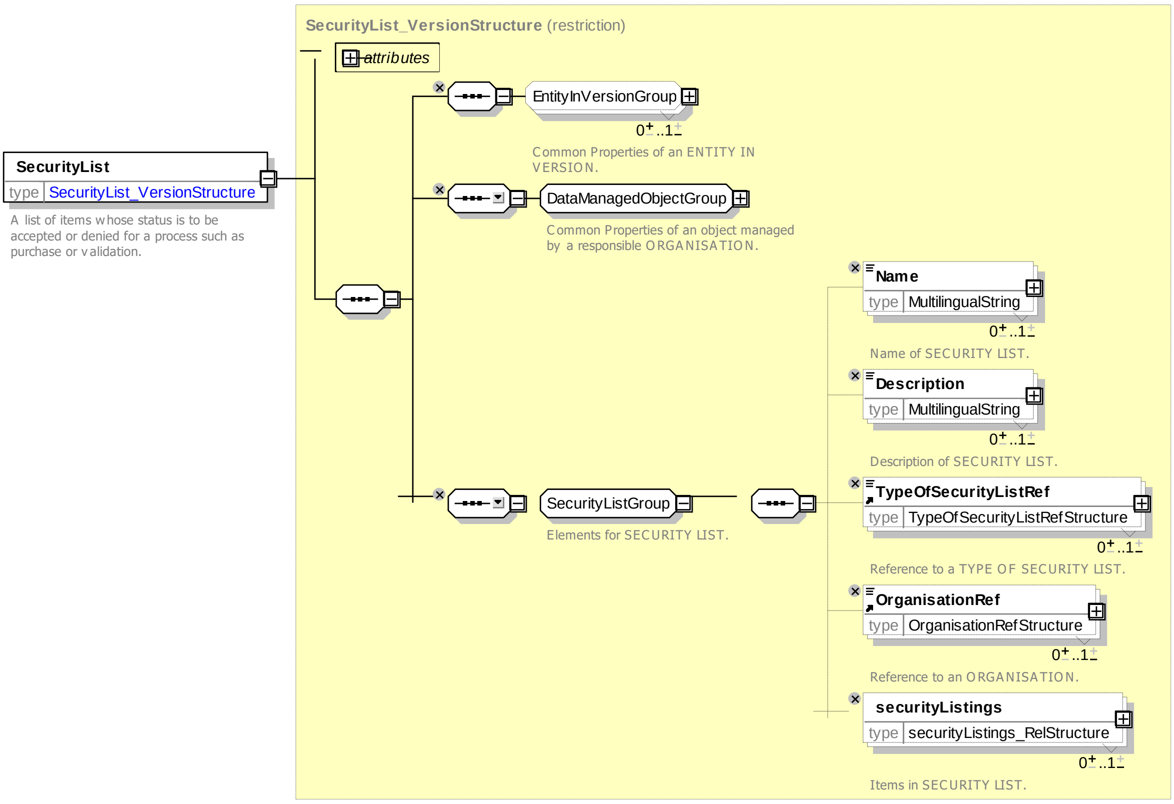


Figure 377 — *SecurityList – XSD*

7.7.19.3.2 Blacklist – Model Element

A list of items (TRAVEL DOCUMENTs, CONTRACTs etc) the validity of which has been cancelled temporarily or permanently, for a specific reason like loss of the document, technical malfunction, no credit on bank account, offences committed by the customer, etc.

Table 330 —*Blacklist – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*SecurityList*	::>	BLACKLIST inherits from SECURITY LIST.
«PK»	***id***	*BlacklistIdType*	1:1	Identifier of BLACKLIST.

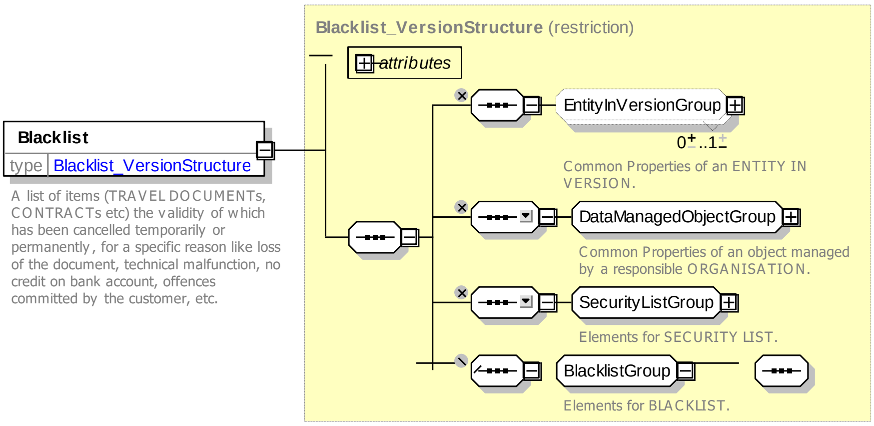


Figure 378 — *Blacklist – XSD*

7.7.19.3.3 Whitelist – Model Element

A list of items (TRAVEL DOCUMENTs, CONTRACTs etc) the validity of which has been cancelled temporarily or permanently, for a specific reason like loss of the document, technical malfunction, no credit on bank account, offences committed by the customer, etc.

Table 331 —*Whitelist – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*SecurityList*	::>	WHITELIST inherits from SECURITY LIST.
«PK»	***id***	*WhitelistIdType*	1:1	Identifier of WHITELIST.

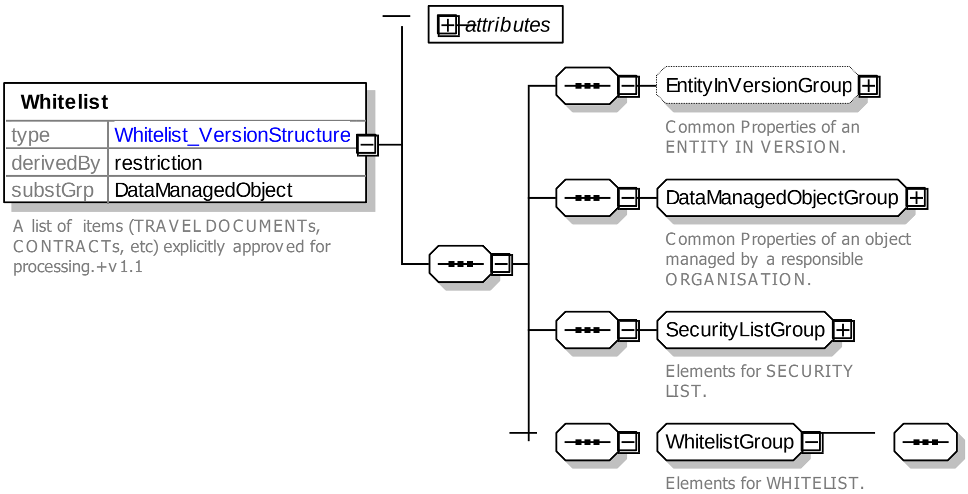


Figure 379 — *Whitelist – XSD*

7.7.19.3.4 SecurityListing – Model Element

The presence of an identified Entity on a SECURITY LIST.

Table 332 —*Security list – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	SECURITY LISTING inherits from VERSIONED CHILD
«PK»	***id***	*SecurityListingIdType*	1:1	Identifier of SECURITY LISTING.
	Name	*MultilingualString*	0:1	Name of SECURITY LISTING.
«FK»	***Security-ListRef***	(*SecurityListRef*)	0:1	Reference to a SECURITY LIST to which this listing belongs (Abstract).

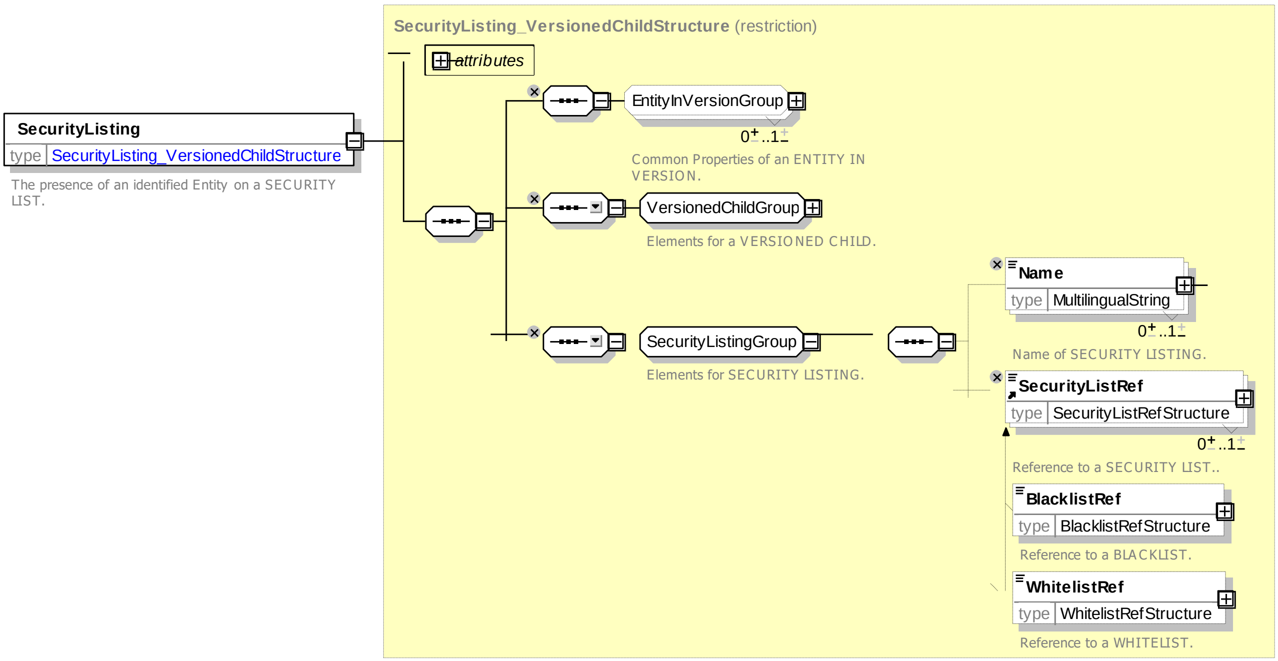


Figure 380 — *SecurityListing – XSD*

7.7.19.3.5 TypeOfSecuritList – Model Element

A classification for a SECURITY LIST.

Table 333 — *TypeOfSecurityList – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF SECURITY LIST inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOSecurityListIdType*	1:1	Identifier of TYPE OF SECURITY LIST.

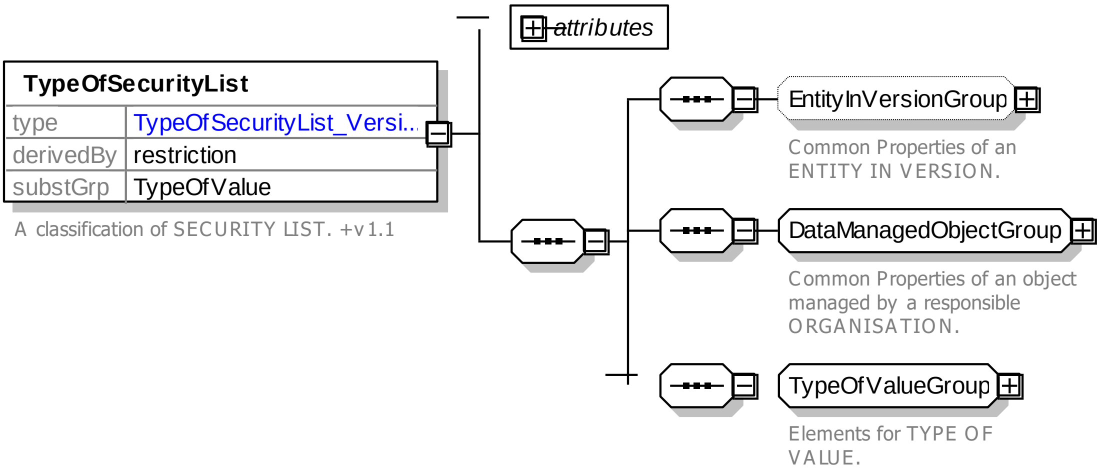


Figure 381 — *TypeOfSecurityList – XSD*

8 Part 1 – The Network Topology

Part1 of NeTEx provides a Network Topology models split into three main submodels:

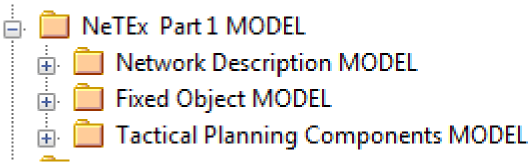


Figure 382 – NeTEx Part 1 MODEL packages

— Network Description model: describes infrastructure elements and physical paths (routes and lines) dedicated to public transport operation;

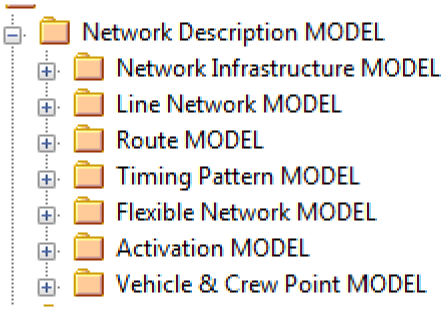


Figure 383 – Network Description MODEL packages

— Fixed Object model: describes geographical aspects of fixed elements such as stopping locations, points of interest and associated equipment ;

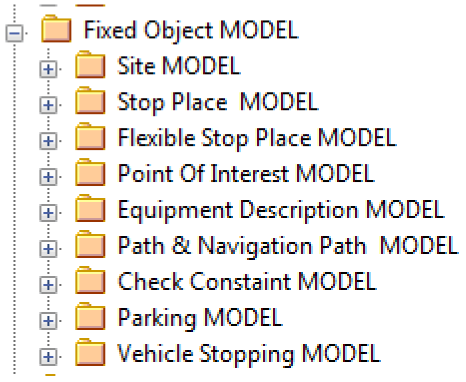


Figure 384 – Fixed Object MODEL packages

— Tactical Planning Components model: describes basic concepts related to the work patterns of public transport vehicles, such as journey patterns and service patterns, useful for planning transport and some related aspect

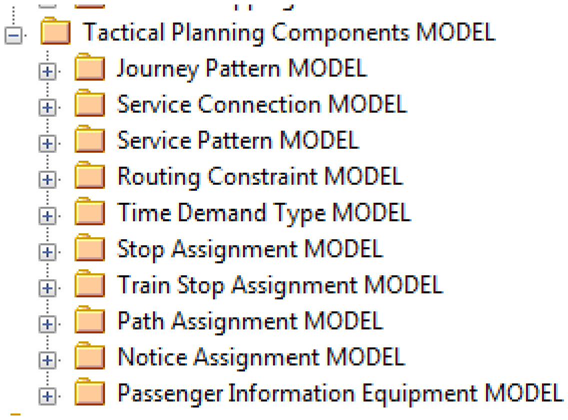


Figure 385 – Tactical Planning Components MODEL packages

8.1 Network Description – Model dependencies

The Network Description model describes the basic physical network for transport and is itself divided into a number of separate submodels covering different aspects of the Network. For ease of understanding, the submodels are presented one at a time, each describing only a small set of related concepts.

The submodels depend on a number of general NeTEx framework models (for example, Generic Point and Link model, Notice model, etc.) described in the Framework model.

NeTEx Framework - Conceptual and Physical data model).

Figure 386 shows the dependencies between the Network Description physical submodels. The terminal packages contain the SERVICE FRAME and the INFRASTRUCTURE FRAME. These two VERSION FRAMEs organise the other elements into a coherent set of elements suitable for exchange as a serialised file. The payload elements are contained in the following packages:

Infrastructure Frame model, composed of the following parts:

— **Infrastructure Network** sub-model: basic physical links of the transport network; incorporates Infrastructure Restrictions.

— **Activation** sub-model: ACTIVATION POINTs and LINKs where monitoring equipment is located in the network.

— **Vehicle and Crew Points** sub-model: different points dedicated to the vehicles and crew changeover that are referenced by vehicle and duty schedules.

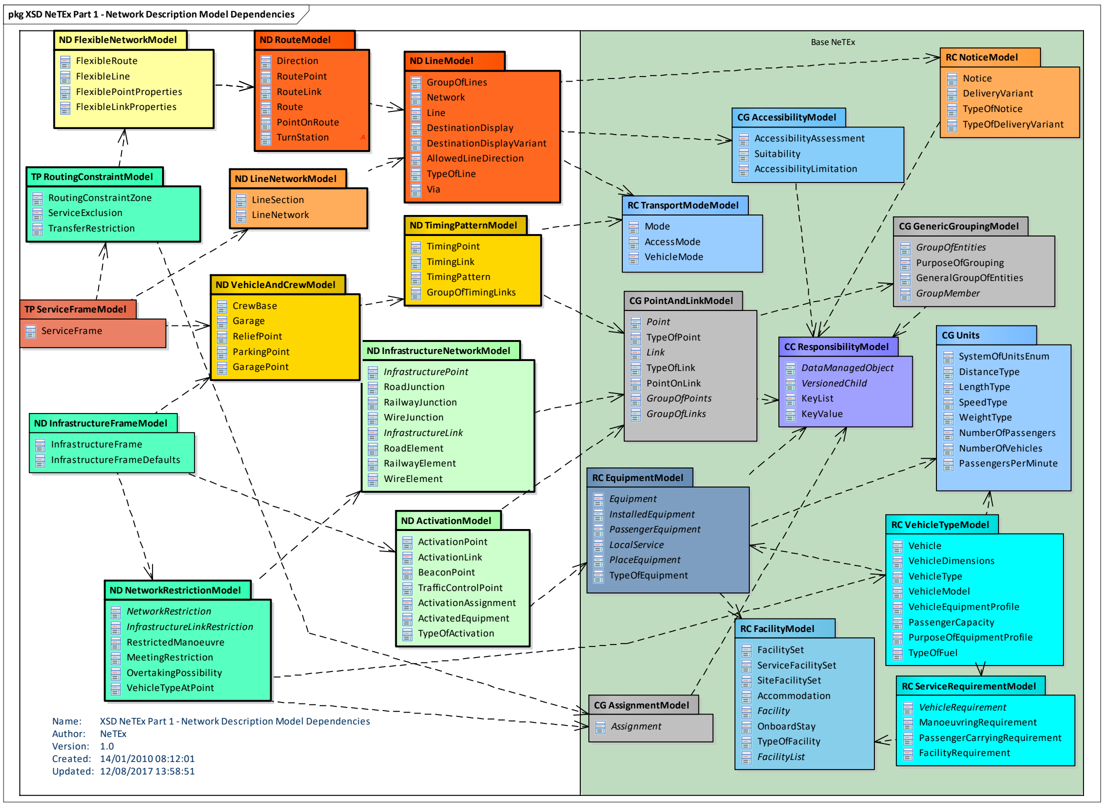
Service Frame model, comprising:

— **Route** model: fixed LINEs and ROUTEs of a transport network.

— **Flexible Network** model: flexible LINEs and ROUTEs of a demand responsive transport network.

— **Line Network** model: overall topology of the LINEs and LINE SECTIONs that make up a transport network.

— **Timing Pattern** model: TIMING POINTs and TIMING LINKs i.e. points and links referenced by schedules.



—

Figure 386 – Network Description – Model dependencies (UML)

8.2 Network Description – Version Frarmes

8.2.1 Infrastructure Frame

8.2.1.1 INFRASTRUCTURE FRAME – Conceptual MODEL

The Network Infrastructure elements of the Network Description MODEL can be grouped with an INFRASTRUCTURE FRAME (Figure 387) which holds a coherent set of infrastructure elements for data exchange. These elements are explained in the subsequent sections. See VERSION FRAME in the NeTEx Framework section for general concepts relating to version frames (7.3.6).

NOTE Explicit frames, such as the INFRASTRUCTURE FRAME are described at the beginning of each major section, and serve as an overview to the overall scope of the chapter and also a form of index to the constituent elements. The actual objects and concepts indicated as being in the frame are explained later in the respective chapter.

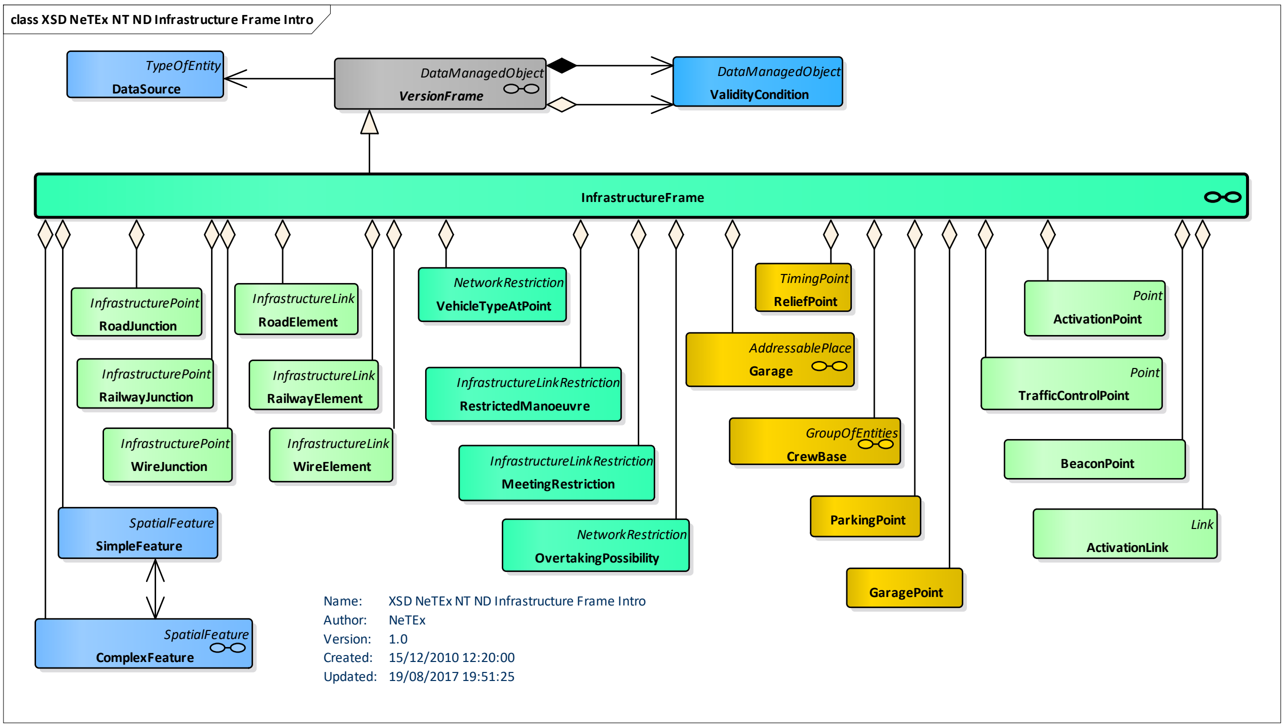


Figure 387 – Infrastructure Frame – Physical Model Overview (UML)

8.2.1.2 Infrastructure Frame – Physical Model

The ***InfrastructureFrame*** element (Figure 388) is a type of ***VersionFrame*** that holds infrastructure elements describing the physical infrastructure.

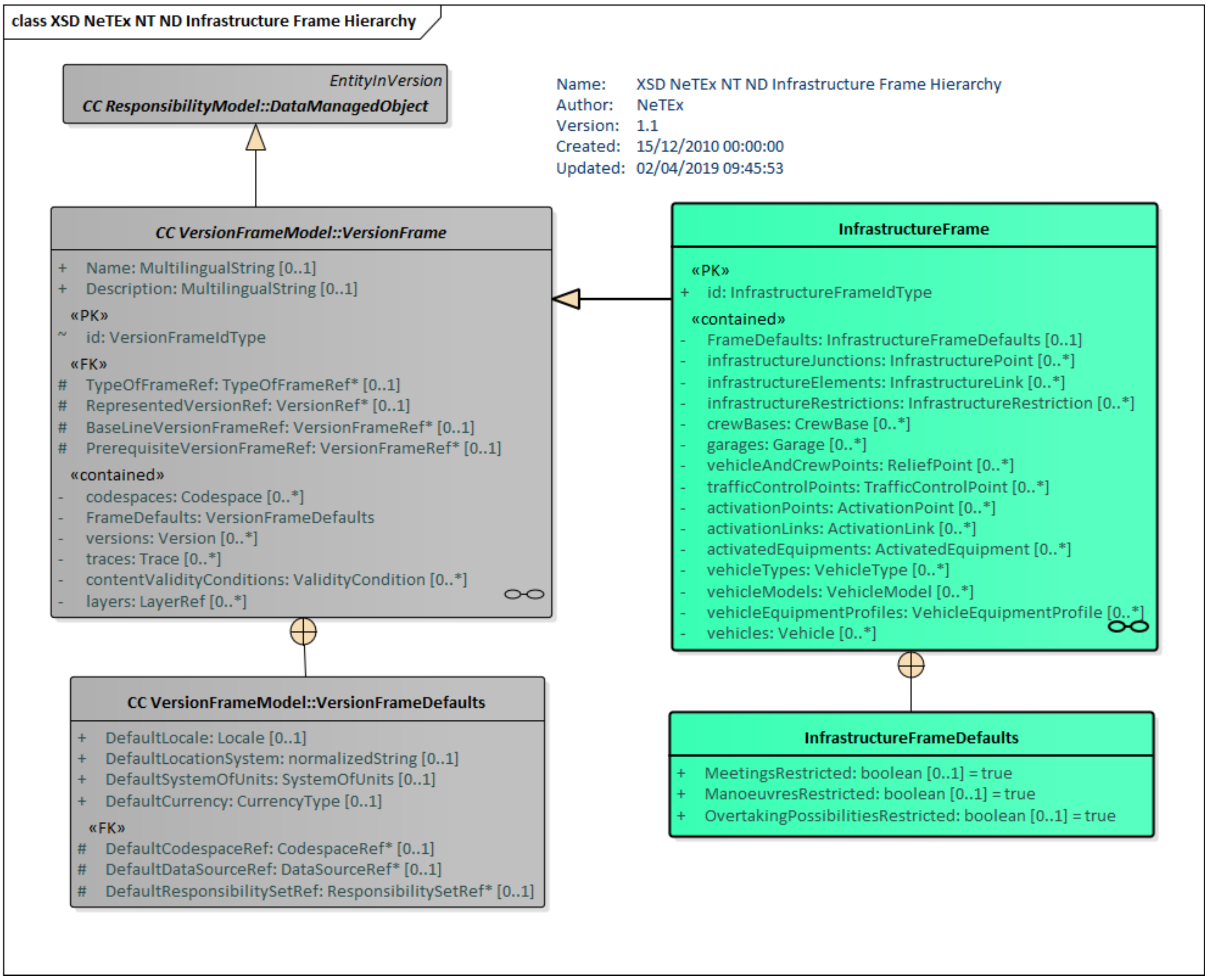


Figure 388 – Infrastructure Frame – Hierarchy (UML)

8.2.1.3 Infrastructure Frame – Attributes and XSD

8.2.1.3.1 Infrastructure Frame – Model Element

An Infrastructure Frame is a set of infrastructure network data (and other data logically related to these) to which the same VALIDITY CONDITIONs have been assigned.

Table 334 – *InfrastructureFrame* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionFrame*	::>	INFRASTRUCTURE FRAME inherits from VERSION FRAME.
«PK»	***id***	*InfrastructureFrameIdType*	1:1	Identifier of INFRASTRUCTURE FRAME.
	MeetingsRestricted	*xsd:boolean*	0:1	Default sense for MEETING RESTRICTIONs in frame. If 'true', meetings at all points are restricted unless explicitly permitted. If 'false', meetings at all

				points are allowed, unless explicitly forbidden by a MEETING RESTRICTION. The default is *'false'*.
	ManoeuvresRestricted	*xsd:boolean*	0:1	Default sense for RESTRICTED MANOEUVRE in frame. If *'true'*, manoeuvres at all points are restricted unless explicitly permitted. If *'false'*, meeting at all points is allowed, unless explicitly forbidden by a RESTRICTED MANOEUVRE. The default is *'false'*.
	OvertakingPossibilities Restricted	*xsd:boolean*	0:1	Default sense for OVERTAKING POSSIBILITY restrictions in frame. If '*true'*, overtaking at all points is restricted unless explicitly permitted. If *'false'*, meeting at all points is allowed, unless explicitly forbidden by an OVERTAKING POSSIBILITY. The default is *'false'*.
«cntd»	***infrastructure-Junctions***	*InfrastructurePoint*	0:*	Infrastructure junctions in the frame.
«cntd»	***infrastructureElements***	*InfrastructureLink*	0:*	Infrastructure elements in the frame.
«cntd»	***infrastructure-Restrictions***	*InfrastructureRestriction*	0:*	Infrastructure restrictions in the frame.
«cntd»	***crewBases***	*CrewBase*	0:*	CREW BASEs in the frame.
«cntd»	***garages***	*Garage*	0:*	GARAGEs in the frame.
«cntd»	***vehicleAndCrewPoints***	*ReliefPoint*	0:*	VEHICLE & CREW POINTs in the frame.
«cntd»	***trafficControlPoints***	*TrafficControlPoint*	0:*	TRAFFIC CONTROL POINTs in the frame.
«cntd»	***activationPoints***	*ActivationPoint+*	0:*	ACTIVATION POINTs in the frame.
«cntd»	***activationLinks***	*ActivationLink*	0:*	ACTIVATION LINKs in the frame.
«cntd»	***activatedEquipments***	*ActivatedEquipment*	0:*	ACTIVATED EQUIPMENTs in the frame.
«cntd»	***vehicleTypes***	*VehicleType*	0:*	VEHICLE TYPEs in the frame.
«cntd»	***vehicleModels***	*VehicleModel*	0:*	VEHICLE MODELs in the frame.
«cntd»	***vehicleEquipment-Profiles***	*VehicleEquipmentProfile*	0:*	VEHICLE EQUIPMENT PROFILE in the frame.
«cntd»	***vehicles***	*Vehicle*	0:*	VEHICLEs in the frame.

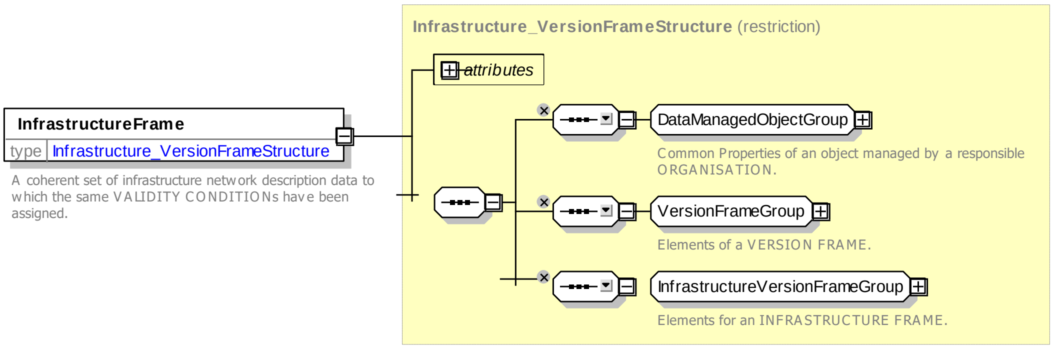


Figure 389 – *InfrastructureFrame* – XSD

8.2.1.3.1.1 InfrastructureFrameGroup – XML Group

Elements for the Infrastructure Frame.

Figure 390 – *InfrastructureFrameGroup* – XSD

8.2.2 Service Frame

8.2.2.1 SERVICE FRAME – Conceptual MODEL

The service related elements of the Network Description model can be grouped into a SERVICE FRAME which holds a coherent set of elements for data exchange. These elements are explained in the subsequent sections. See VERSION FRAME in the NeTEx Framework section for general concepts relating to version frames.

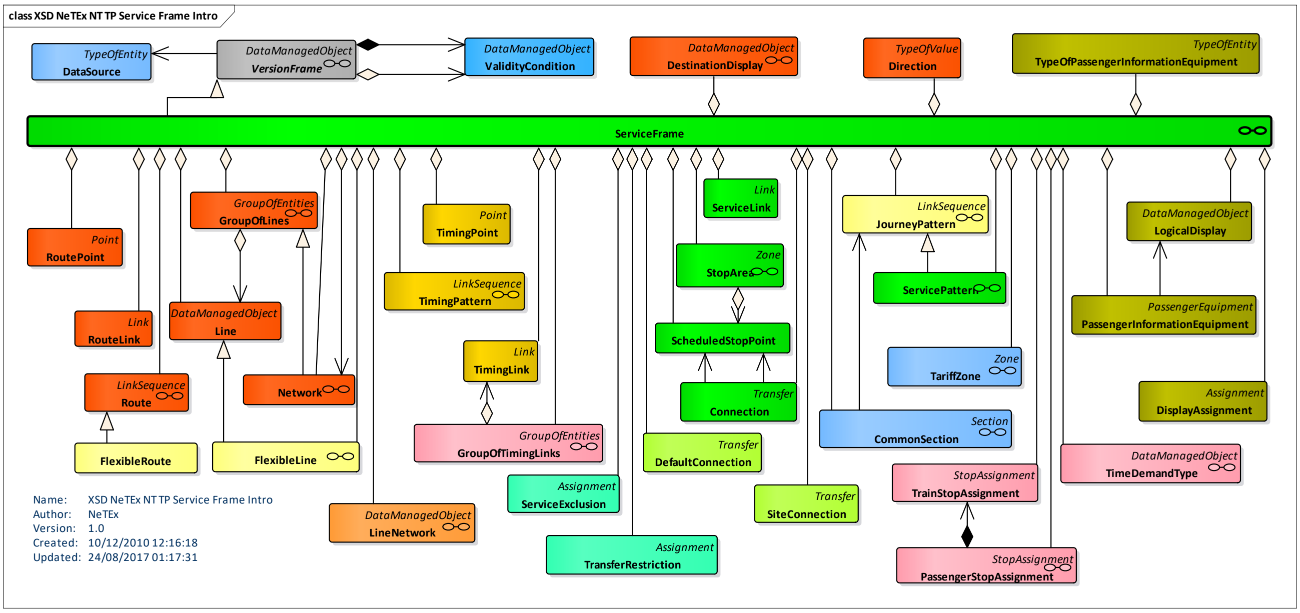


Figure 391 – Service Frame – Physical Model Overview (UML)

8.2.2.2 Service Frame – Physical Model

The ***Service Frame*** element (Figure 392) is a type of ***VersionFrame*** that holds elements describing various underlying aspects of the transport Service.

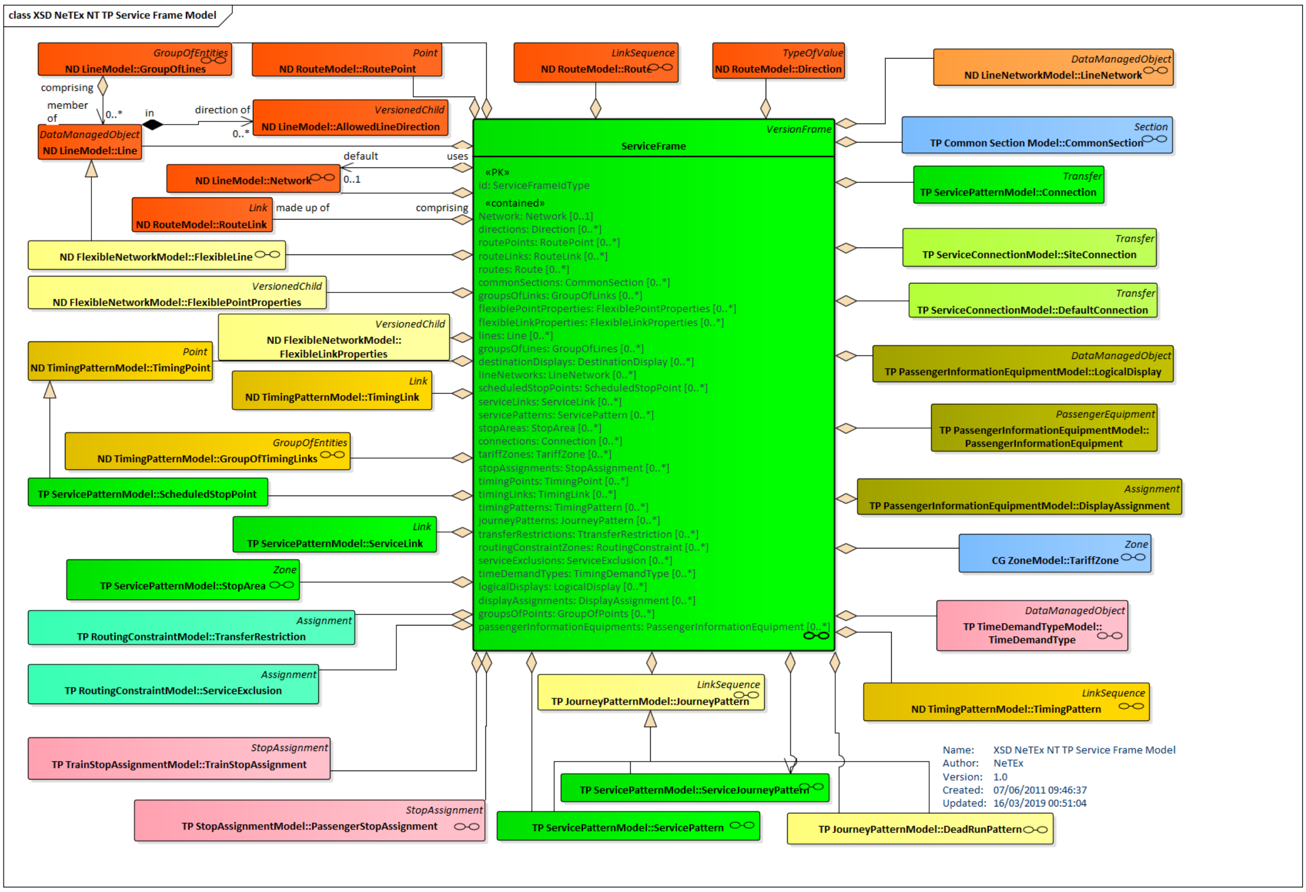


Figure 392 – Service Frame – Physical Model (UML)

8.2.2.3 Service Frame – Attributes and XSD

8.2.2.3.1 Service Frame – Model Element

The SERVICE FRAME holds a coherent set of Service elements for data exchange.

These elements are explained in subsequent sections.

It is a set of Service network data (and other data logically related to these) to which the same VALIDITY CONDITIONs have been assigned.

Table 335 – *ServiceFrame* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionFrame*		SERVICE FRAME inherits from VERSION FRAME.
«PK»	***id***	*ServiceFrameIdType*	1:1	Identifier of SERVICE FRAME.
«cntd»	***Network***	*Network*	0:1	Reference to a NETWORK of which this is a frame.
«cntd»	***directions***	*Direction*	0:*	DIRECTIONs in SERVICE FRAME.
«cntd»	***routePoints***	*RoutePoint*	0:*	ROUTE POINTs in SERVICE FRAME.

«cntd»	***routeLinks***	*RouteLink*	0:*	ROUTE LINKs in SERVICE FRAME.
«cntd»	***routes***	*Route*	0:*	ROUTEs in SERVICE FRAME.
«cntd»	***commonSections***	*CommonSection*	0:*	COMMON SECTIONs in SERVICE FRAME.
«cntd»	***groupsOfLinks***	*GroupOfLinks*	0:*	GROUP OF LINKs in SERVICE FRAME.
«cntd»	***lines***	*Line*	0:*	LINEs in SERVICE FRAME.
«cntd»	***groupsOfLines***	*GroupOfLines*	0:*	GROUP OF LINEs in SERVICE FRAME.
«cntd»	***destinationDisplays***	*DestinationDisplay*	0:*	DESTINATION DISPLAYs in SERVICE FRAME.
«cntd»	***lineNetworks***	*LineNetwork*	0:*	LINE NETWORKs in SERVICE FRAME.
«cntd»	***scheduledStop-Points***	*ScheduledStopPoint*	0:*	SCHEDULED STOP POINTs in SERVICE FRAME.
«cntd»	***serviceLinks***	*ServiceLink*	0:*	SERVICE LINKs in SERVICE FRAME.
«cntd»	***ServicePatterns***	*ServicePattern*	0:*	SERVICE PATTERN in SERVICE FRAME.
«cntd»	***stopAreas***	*StopArea*	0:*	STOP AREAs in SERVICE FRAME.
«cntd»	***connections***	*Connection*	0:*	CONNECTIONs in SERVICE FRAME.
«cntd»	***tariffZones***	*TariffZone*	0:*	TARIFF ZONEs in SERVICE FRAME.
«cntd»	***stopAssignments***	*StopAssignment*	0:*	STOP ASSIGNMENT in SERVICE FRAME.
«cntd»	***timingPoints***	*TimingPoint*	0:*	TIMING POINTs in SERVICE FRAME.
«cntd»	***timingLinks***	*TimingLink*	0:*	TIMING LINKs in SERVICE FRAME.
«cntd»	***timingPatterns***	*TimingPattern*	0:*	TIMING PATTERN in SERVICE FRAME.
«cntd»	***journeyPatterns***	*JourneyPattern*	0:*	JOURNEY PATTERNs in SERVICE FRAME.
«cntd»	***transferRestrictions***	*TtransferRestriction*	0:*	TRANSFER RESTRICTIONs in SERVICE FRAME.
«cntd»	***routingConstraint-Zones***	*RoutingConstraintZone*	0:*	ROUTING CONSTRAINT ZONEs in SERVICE FRAME.
«cntd»	***serviceExclusions***	*ServiceExclusion*	0:*	SERVICE EXCLUSIONs in SERVICE FRAME.
«cntd»	***timeDemandTypes***	*TimingDemandType*	0:*	TIME DEMAND TYPEs in SERVICE FRAME.
«cntd»	***logicalDisplays***	*LogicalDisplay*	0:*	LOGICAL DISPLAYs in SERVICE FRAME.
«cntd»	***display-Assignments***	*DisplayAssignment*	0:*	DISPLAY ASSIGNMENTs in SERVICE FRAME.
«cntd»	***groupsOfPoints***	*GroupOfPoints*	0:*	GROUP OF POINTs in SERVICE FRAME.

«cntd»	***passenger-Information-Equipments***	*PassengerInformation Equipment*	0:*	PASSENGER INFORMATION EQUIPMENT in SERVICE FRAME.

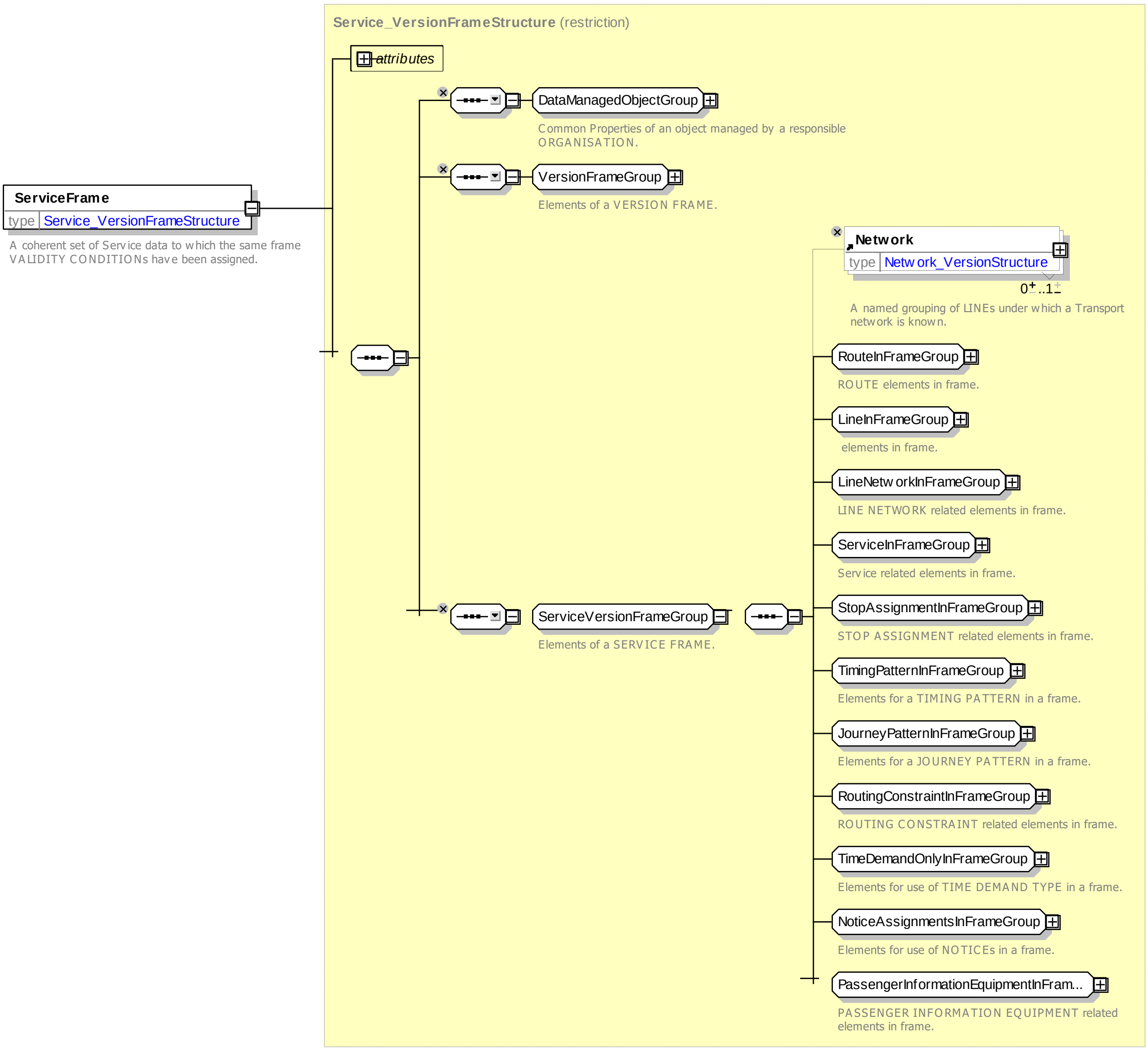


Figure 393 – Service Frame – XSD

8.3 Network Description – Subsystem

8.3.1 Network Infrastructure

NOTE The following explanations use excerpts from Transmodel.

The Network Infrastructure model describes the physical network on which the transport services run; a closely related Network Restriction model describes the physical restrictions on its use. This part does not concern the service aspects, i.e. vehicle work patterns described separately (e.g. by TIMING PATTERNs, JOURNEY PATTERNs, SERVICE PATTERNs).

8.3.1.1 NETWORK INFRASTRUCTURE – Conceptual MODEL

The Network Infrastructure model (Figure 394) describes the main components of the physical path network (rail, roads, etc.).

This modelling of the infrastructure is, however, very basic and simple and is used here to represent specific operational constraints (restrictions) for public transport operation resulting from the characteristics of the INFRASTRUCTURE POINTs and LINKs and of VEHICLE TYPEs. The spatial detailed organisation of the infrastructure itself is described by other models (GDF, INSPIRE, etc.) and the data is usually provided by GIS data sets.

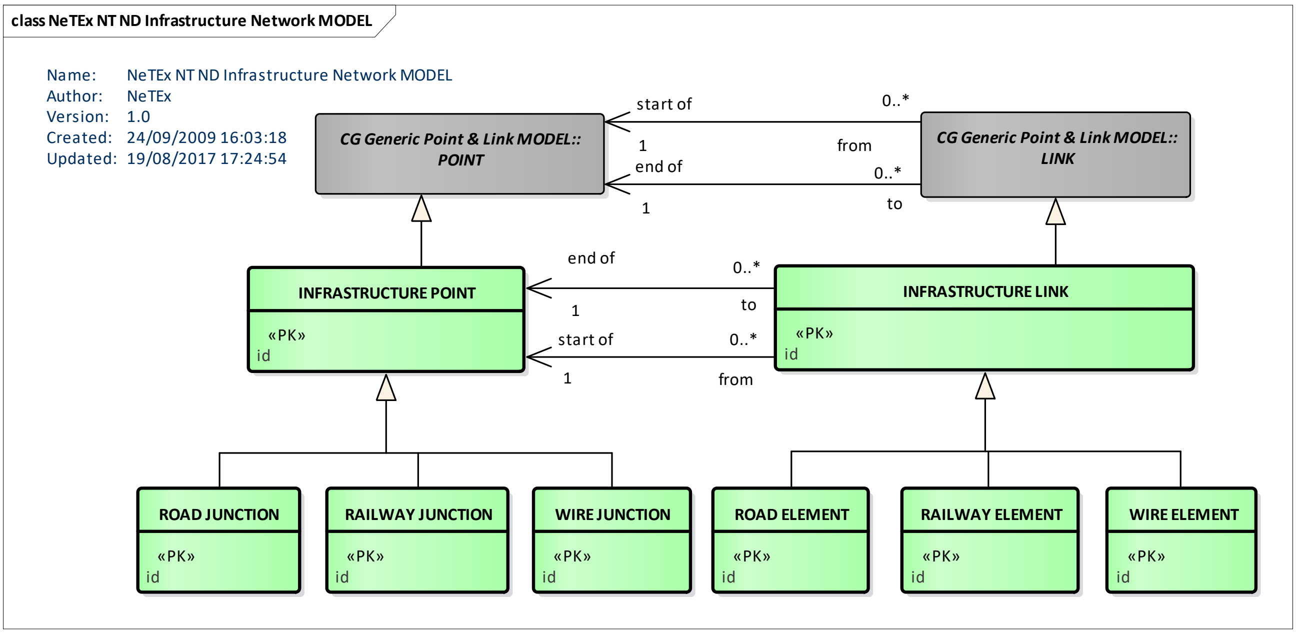


Figure 394 – Infrastructure – Conceptual MODEL (UML)

8.3.1.1.1 Infrastructure Points and Links

The PT network is described in Transmodel by POINTs and LINKs. This means that separate description of a network either as a set of points or a set of links, or both are possible and may be kept separately (see Section 7.6.5.1).

The approach of representing the network in terms of generic POINTs and/or LINKs and their specialisations (here: INFRASTRUCTURE POINT, INFRASTRUCTURE LINK) is used extensively in Transmodel to describe distinct functional layers as separate graphs.

In Transmodel terms, the Infrastructure Network builds a layer. A layer is a user-defined set of data, specified for a particular functional purpose, originating from a particular DATA SOURCE, associating data referring one particular LOCATING SYSTEM.

Examples of other layers are : Timing Pattern layer (defined through TIMING POINTs and TIMING LINKs), Service Pattern layer (defined through SCHEDULED STOP POINTS and SERVICE LINKs). Transmodel defines a correspondence mechanism between layers, called “projection” (see Section 7.6.9). It should be noted that the uniqueness of a LOCATING SYSTEM within a layer is an important parameter, in particular for the coherence of distances.

Each separate layer reflects different concerns and is deliberately kept independent of other layers. Thus for example the modelling of the objects necessary to describe the work patterns of vehicles (JOURNEY

PATTERNs) is represented separately, in the layers describing the operational planning (see Section 8.5) and not in the infrastructure layer.

The different functional layers may be projected (using the Transmodel projection mechanism) onto the infrastructure layer to represent how they are related to the physical paths represented by sequences of INFRASTRUCTURE LINKs.

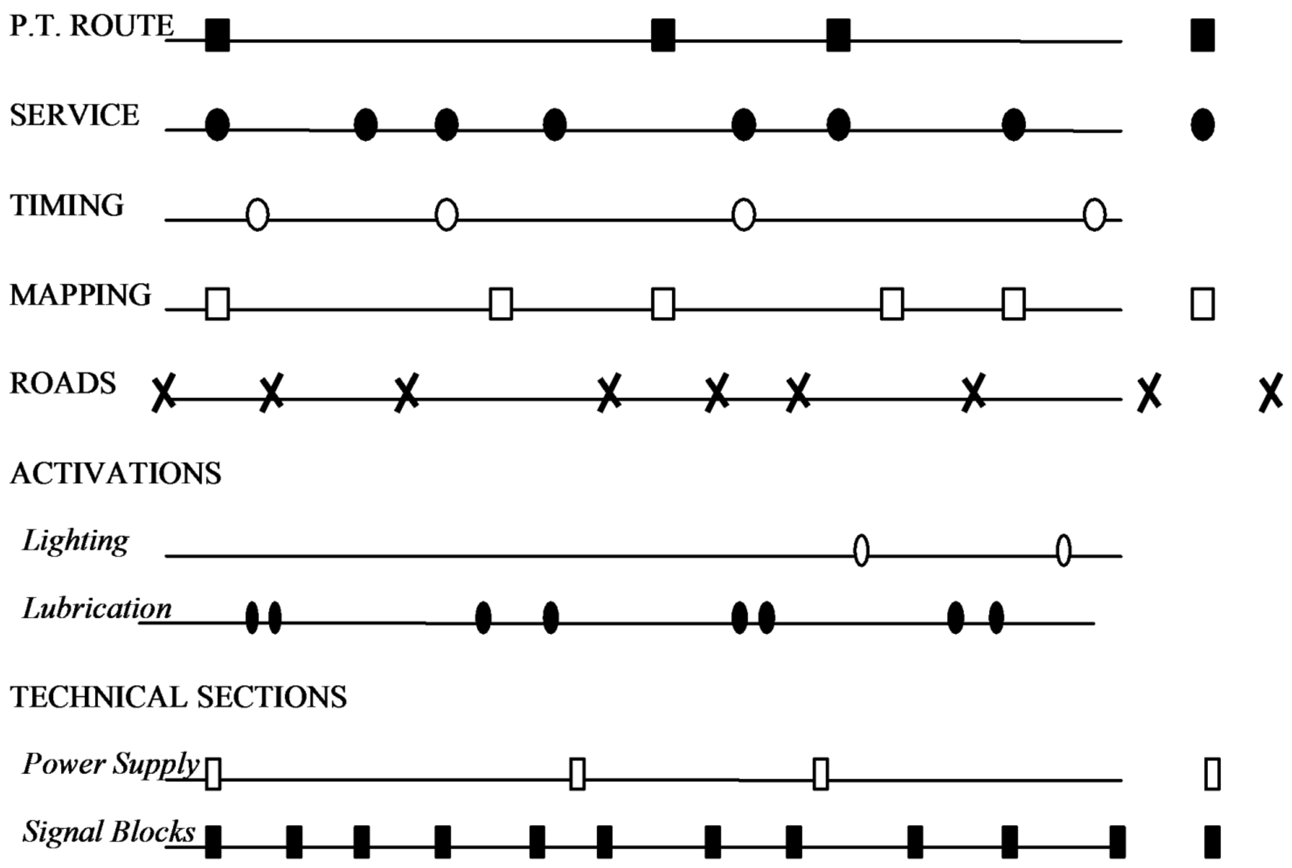


Figure 395 – Examples of layers – different functional aspects of a network (source Transmodel)

Any POINT necessary to describe the infrastructure network is defined as an INFRASTRUCTURE POINT, which is a generic entity including several subtypes (e.g. ROAD JUNCTION, RAILWAY JUNCTION). Similarly, the necessary LINKs between the POINTs are defined as INFRASTRUCTURE LINKs (e.g. ROAD ELEMENT, RAILWAY ELEMENT).

Any INFRASTRUCTURE LINK must be bordered by a start and an end INFRASTRUCTURE POINT. This orientation does not necessary refer to the direction of the traffic flow, but has to be interpreted as an arbitrary orientation (it may be "used" in one way or the other by the objects, like ROUTEs, JOURNEY PATTERNs, etc. referring to it through the PROJECTION mechanism).

8.3.1.1.2 Road Network: ROAD JUNCTION & ROAD ELEMENT

The physical road network represents all the carriageways available for buses, into which the bus line network can be embedded.

The corresponding road INFRASTRUCTURE POINTs are defined as ROAD JUNCTIONs, while the corresponding INFRASTRUCTURE LINKs are defined as ROAD ELEMENTs.

8.3.1.1.3 Rail Network: RAIL JUNCTION & RAIL ELEMENT

The rail network model represents the track network along which VEHICLEs (usually TRAINs) can physically proceed, without taking into account other operational aspects such as security, regulations or operational conventions followed by the company staff or other authorities. Railway elements are modelled in this data model for reference purposes and not for control functions.

The corresponding rail INFRASTRUCTURE POINTs are defined as RAILWAY JUNCTIONs, while the corresponding INFRASTRUCTURE LINKs are defined as RAILWAY ELEMENTs.

RAILWAY ELEMENTs will always have to be interpreted as non-overlapping parts of the rail network. This means that one railway section between two switches or crossings cannot be described alternatively, and in parallel, by two or more different subdivisions into chains of railway elements. Different sequences of railway elements between two switches will principally mean multiple connections, physically separated from each other.

The location where contiguous RAILWAY ELEMENTs are connected is represented by a RAILWAY JUNCTION. The two RAILWAY JUNCTIONs bounding a RAILWAY ELEMENT are specified by two relationships between these entities. The names of the relationship ends suggest a direction, which has to be interpreted as an arbitrary orientation, similar to the orientation of ROAD ELEMENTs described in the previous section.

8.3.1.1.4 Wire Network: WIRE JUNCTION & WIRE ELEMENT

The wire network for power supply of trolley buses (or trams) is modelled according to the same principles as applied for the rail network. WIRE ELEMENTs will be defined as the links between WIRE JUNCTIONs, which may be at places where three or more WIRE ELEMENTs are joined, at locations where only two adjacent WIRE ELEMENTs are connected or possibly at intermediate locations.

8.3.1.2 Network Infrastructure – Example

In the figure below, the Street Network is an example of an Infrastructure Network, that may be represented (in a GIS for instance) by ROAD JUNCTIONs and ROAD LINKs.

Other layers are represented by coloured graphs: green (Timing layer), blue (Route layer), red (Service layer).

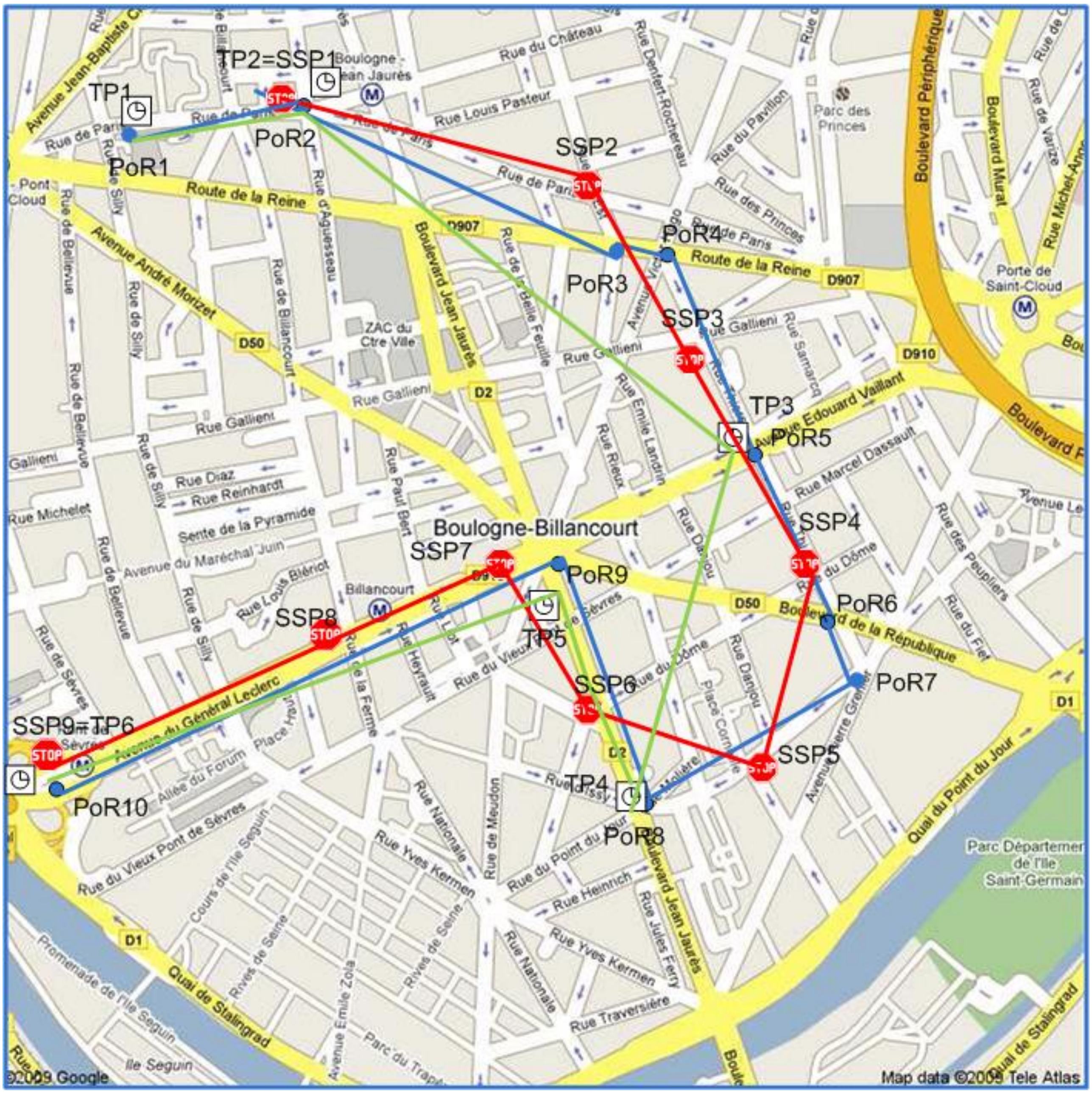


Figure 396 – Network Infrastructure example

8.3.1.3 Network Infrastructure – Physical Model

Figure 397 shows the Physical model for the Network Infrastructure. For each Conceptual model element there is a corresponding Physical model element.

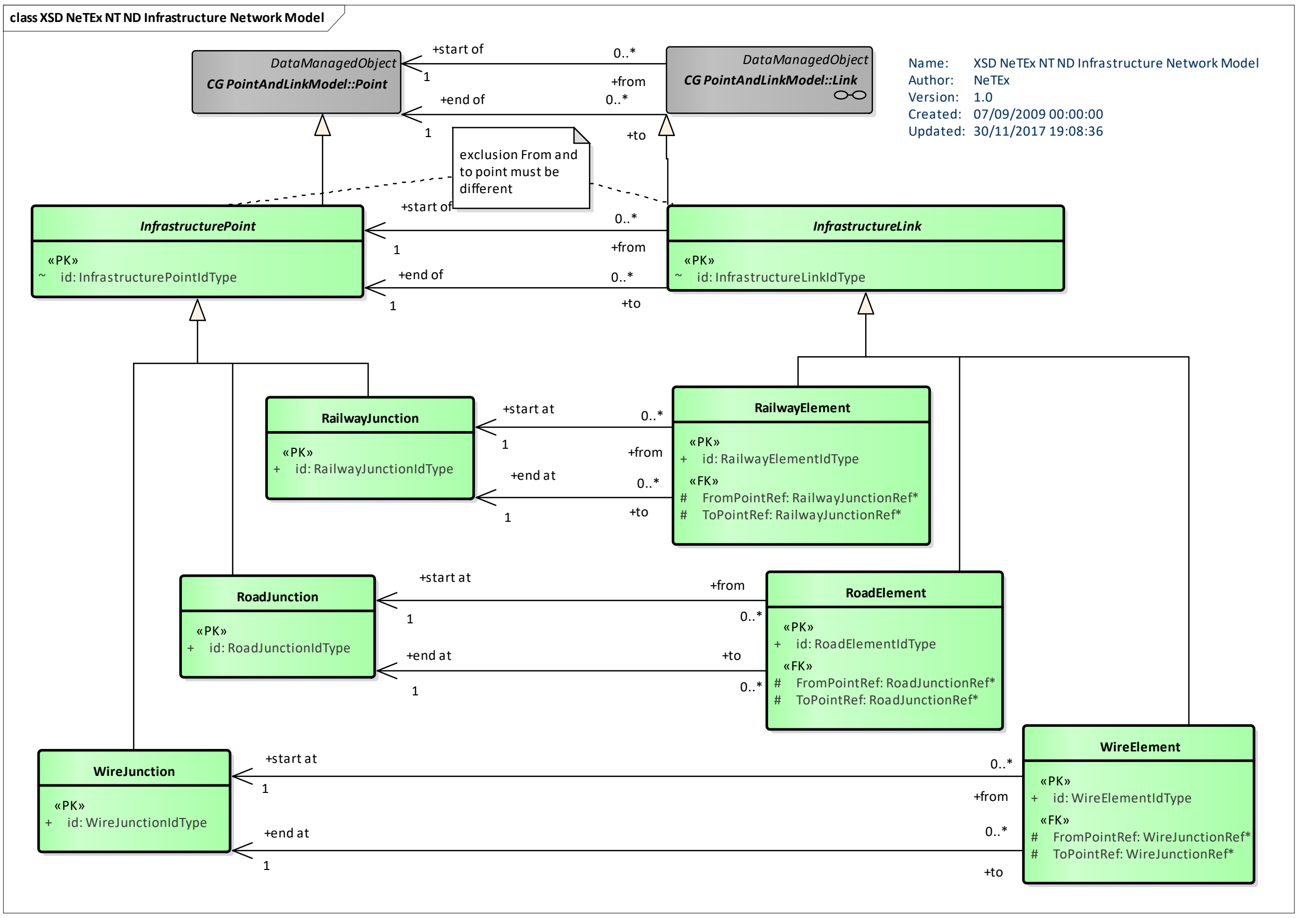


Figure 397 – Infrastructure – Physical Model (UML)

8.3.1.4 Network Infrastructure – Attributes and XSD

The physical model elements are described below.

8.3.1.4.1 InfrastructurePoint – Model Element

A common supertype for any POINT of the physical network (e.g. RAILWAY JUNCTION).

Table 336 – *InfrastructurePoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Point*	::>	INFRASTRUCTURE POINT inherits from POINT
«PK»	***id***	*InfrastructurePointIdType*	1:1	Identifier of INFRASTRUCTURE POINT.

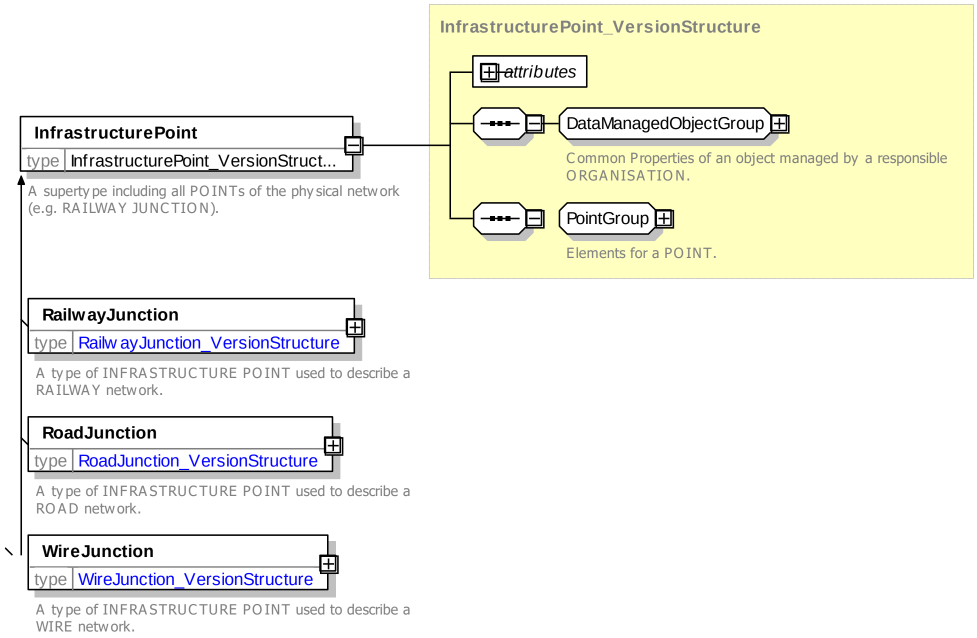


Figure 398 – *InfrastructurePoint* – XSD

8.3.1.4.2 RailwayJunction – Model Element

A type of INFRASTRUCTURE POINT used to describe a railway network.

Table 337 – *RailwayJunction*– Element

Classification	Name	Type	Cardinality	Description
::>	::>	*InfrastructurePoint*	::>	RAILWAY JUNCTION inherits from INFRASTRUCTURE POINT
«PK»	***id***	*RailwayJunctionIdType*	1:1	Identifier of RAILWAY JUNCTION.

8.3.1.4.3 RoadJunction – Model Element

A type of INFRASTRUCTURE POINT used to describe a road network.

Table 338 – *RoadJunction*– Element

Classification	Name	Type	Cardinality	Description
::>	::>	*InfrastructurePoint*	::>	ROAD JUNCTION inherits from INFRASTRUCTURE POINT
«PK»	***id***	*RoadJunctionIdType*	1:1	Identifier of ROAD JUNCTION.

8.3.1.4.4 WireJunction – Model Element

A type of INFRASTRUCTURE POINT used to describe a wire network.

Table 339 – *WireJunction*– Element

Classification	Name	Type	Cardinality	Description
::>	::>	*InfrastructurePoint*	::>	WIRE JUNCTION inherits from INFRASTRUCTURE POINT
«PK»	***id***	*WireJunctionIdType*	1:1	Identifier of WIRE JUNCTION.

8.3.1.4.5 InfrastructureLink – Model Element

A supertype including all LINKs of the physical network (e.g. RAILWAY ELEMENT).

Table 340 – *InfrastructureLink* – Element

Classification	Name	Type	Cardinality	Description
::>	***::>***	*Link*	::>	INFRASTRUCTURE LINK inherits from LINK
«PK»	***id***	*InfrastructureLinkIdType*	1:1	Identifier of INFRASTRUCTURE LINK.

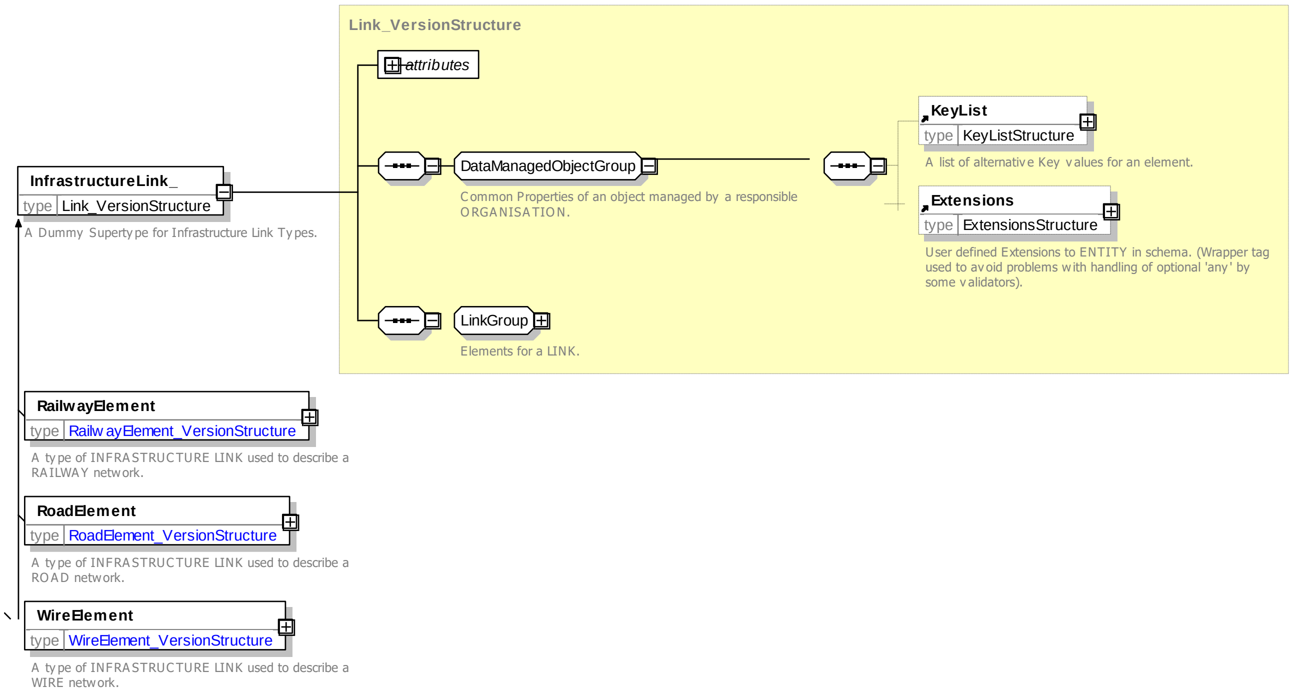


Figure 399 – *InfrastructureLink* – XSD

8.3.1.4.6 RailwayElement – Model Element

A specialisation of INFRASTRUCTURE LINK used to describe a railway network.

Table 341 – *RailwayElement* – Element

Classification	Name	Type	Cardinality	Description
::>	***::>***	*Link*	::>	RAILWAY ELEMENT inherits from INFRASTRUCTURE LINK
«PK»	***id***	*RailwayElementIdType*	1:1	Identifier of RAILWAY ELEMENT.
«FK»	***FromPointRef***	*RailwayJunctionRef*	1:1	Reference to RAILWAY POINT at which RAILWAY ELEMENT starts.
«FK»	***ToPointRef***	*RailwayJunctionRef*	1:1	Reference to RAILWAY POINT at which RAILWAY ELEMENT ends.

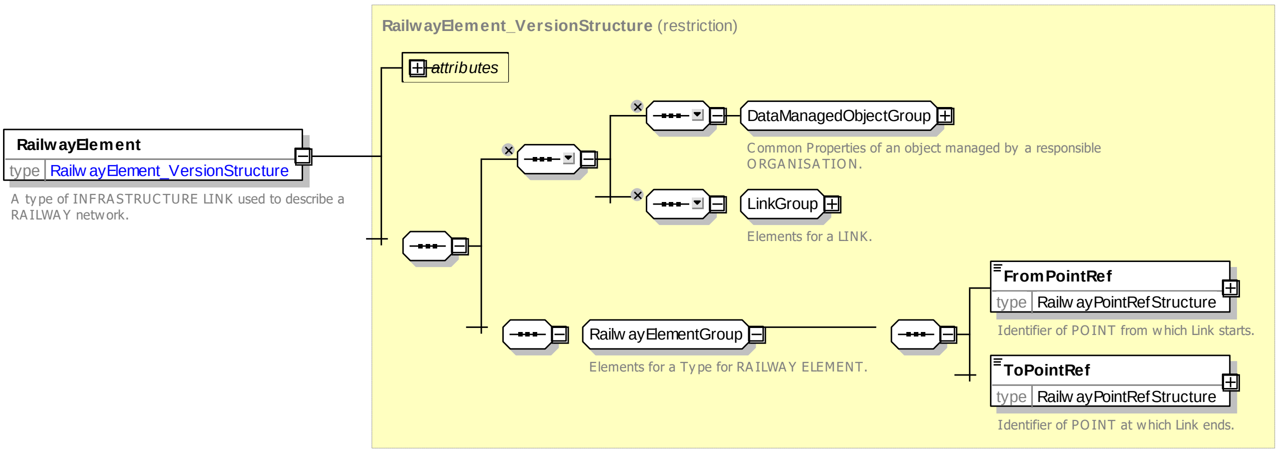


Figure 400 – *RailwayElement – XSD*

8.3.1.4.7 RoadElement – Model Element

A specialisation of INFRASTRUCTURE LINK used to describe a road network.

Table 342 – *RoadElement* – Element

Classification	Name	Type	Cardinality	Description
::>	***::>***	*Link*	::>	ROAD ELEMENT inherits from INFRASTRUCTURE LINK
«PK»	***id***	*RoadElementIdType*	1:1	Identifier of ROAD ELEMENT.
«FK»	***FromPointRef***	*RoadJunctionRef*	1:1	Reference to ROAD POINT at which ROAD ELEMENT starts.
«FK»	***ToPointRef***	*RoadJunctionRef*	1:1	Reference to ROAD POINT at which ROAD ELEMENT ends.

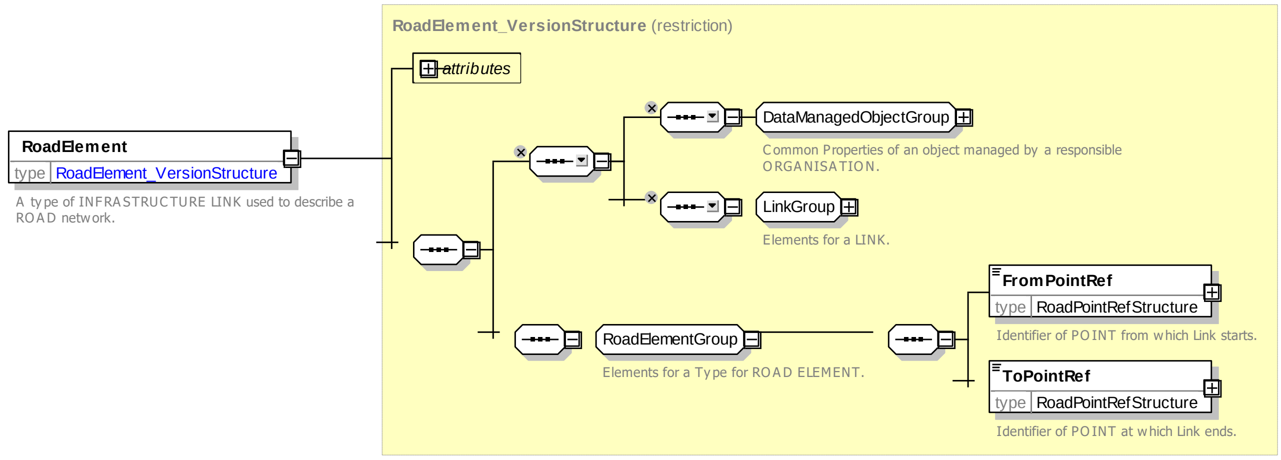


Figure 401 – *RoadElement* – XSD

8.3.1.4.8 WireElement – Model Element

A specialisation of INFRASTRUCTURE LINK used to describe a wire network.

Table 343 – *WireElement* – Element

Classification	Name	Type	Cardinality	Description
::>	***::>***	*Link*	::>	WIRE ELEMENT inherits from INFRASTRUCTURE LINK
«PK»	***id***	*WireElementIdType*	1:1	Identifier of WIRE ELEMENT.
«FK»	***FromPointRef***	*WireJunctionRef*	1:1	Reference to WIRE POINT at which WIRE ELEMENT starts.
«FK»	***ToPointRef***	*WireJunctionRef*	1:1	Reference to WIRE POINT at which WIRE ELEMENT ends.

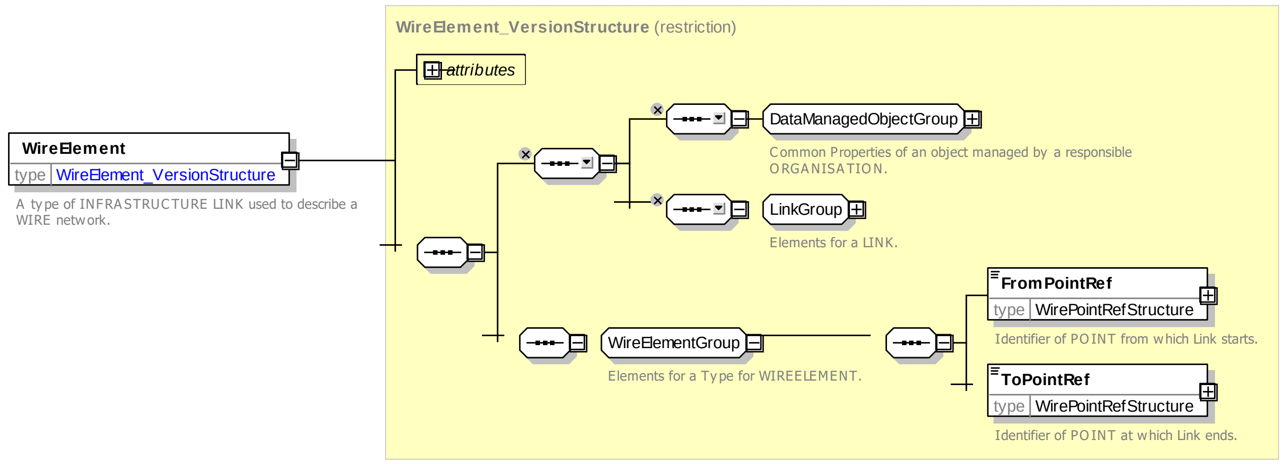


Figure 402 – *WireElement – XSD*

8.3.1.4.9 XML Example of Road Junctions and Road Links

The following XML code fragment shows two ROAD ELEMENT links between four ROAD JUNCTIONs.

EXAMPLE

```
<junctions>
    <RoadJunction id="gis:RoadJunction:rdJ001">
        <Name>Junc 1</Name>
        <Location>
            <Longitude>-0.2071397147</Longitude>
            <Latitude>51.4217482061</Latitude>
        </Location>
    </RoadJunction>
    <RoadJunction 1" id="gis:RoadJunction:rdJ002">
        <Name>Junc 2</Name>
        <Location>
            <Longitude>-0.2071397147</Longitude>
            <Latitude>51.4217482061</Latitude>
        </Location>
    </RoadJunction>
    <RoadJunction 1" id="gis:RoadJunction:rdJ003">
        <Name>Junc 3</Name>
        <Location>
            <Longitude>-0.2071397147</Longitude>
            <Latitude>51.4217482061</Latitude>
        </Location>
    </RoadJunction>
    <RoadJunction 1" id="gis:RoadJunction:rdJ004">
        <Name>Junc 4</Name>
        <Location>
            <Longitude>-0.2071397147</Longitude>
            <Latitude>51.4217482061</Latitude>
        </Location>
    </RoadJunction>

</junctions>
<elements>
    <RoadElement 1" id="gis:RoadElement:rdE001_1to2">
        <Name>Junc 1 -2</Name>
        <FromPointRef version="002" ref="gis:RoadJunction:rdJ001"/>
        <ToPointRef version="002" ref="gis:RoadJunction:rdJ002"/>
    </RoadElement>
    <RoadElement 1" id="gis:RoadElement:rdE002_2to3">
        <Name>Road between Junc 2 -3</Name>
        <FromPointRef version="002" ref="gis:RoadJunction:rdJ002"/>
        <ToPointRef version="002" ref="gis:RoadJunction:rdJ003"/>
    </RoadElement>
</elements>
```

8.3.1.4.10 Network Restriction

NOTE The following explanations use excerpts from Transmodel.

Constraints resulting from the physical characteristics of the network are represented in Transmodel by a range of restrictions. The Network Restriction model represents a number of the most relevant constraints (e.g. the OVERTAKING POSSIBILITY). Transmodel explains the approach as follows: the fact that trains cannot overtake each other or meet each other on the same track is obvious for railway systems, but similar restrictions apply for trolley buses and even conventional buses, under specific circumstances (depending on the number and width of lanes on the street). This type of restriction may be relevant for the scheduling process, because vehicle journeys must be scheduled in a way to avoid such conflicting events.

8.3.1.5 NETWORK RESTRICTION – Conceptual MODEL

The Network Restriction model is not aimed at describing the management of track and train movement management, for which the concepts to consider are far more complex. It fits with a use case often found in light train operation, which consists of an initial verification of the train movements planned in a schedule, in order to check whether there are situations in which the track constraints makes the schedule impossible to run. This function is usually operated with feedback to the scheduling process.

The model comprises a set of different types of Network Restriction elements (VEHICLE TYPE AT POINT, OVERTAKING POSSIBILITY, restricted MANOEUVRE and MEETING RESTRICTION) that apply to specific VEHICLE TYPEs.

Restrictions are explicit: if no NETWORK DESCRIPTION is described, it can be assumed that no limitation apply.

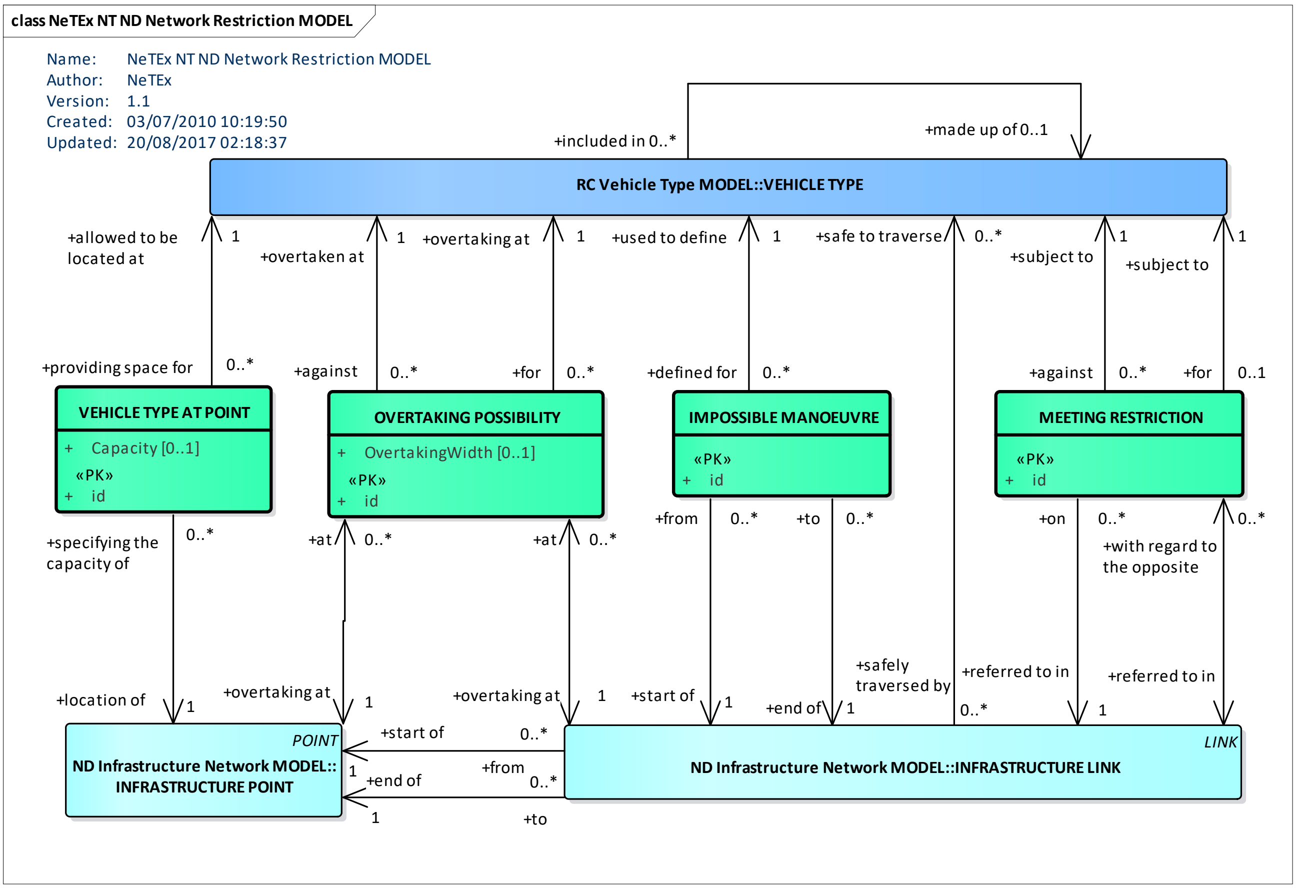


Figure 403 – Infrastructure Restriction – Conceptual MODEL (UML)

The following sections are an excerpt of Transmodel documentation, adapted as regards the references to this document or naming of some classes if appropriate.

8.3.1.5.1 Vehicle Types at Points

A VEHICLE TYPE characterises the common properties of a defined class of public transport vehicles (see Section 7.7.11). Vehicles of a certain VEHICLE TYPE may not be allowed, or physically not able, to stop for any length of time at particular INFRASTRUCTURE POINTs in the network. The entity VEHICLE TYPE AT POINT may be used to express how many vehicles of each type there is space for, at the concerned POINT. This will usually be a SCHEDULED STOP POINT. If the number is 0, then vehicles of that VEHICLE TYPE cannot stop at this INFRASTRUCTURE POINT at all. This restriction may sometimes be relevant for checking the timing of overtaking journeys during the scheduling process.

8.3.1.5.2 Availability of Links

Vehicles of a certain VEHICLE TYPE may not be able, allowed or safe to cross particular ROUTE LINKs in the network. For example, a double-decker bus may not be able to pass under a low bridge. The reference data model expresses this as a positive relationship: a VEHICLE TYPE is safe to traverse a particular ROUTE LINK.

There may be LINKs which are not available at all on certain DAY TYPEs (see Section 7.7.5.3.1). While these limitations generally depend only on the choice of the public transport company to offer or not to offer particular services, there may be physical restrictions that prevent particular LINKs to be used on a specific DAY TYPE. For instance, a street may be blocked because of a special event (e.g. market day) which occurs regularly on each day of that DAY TYPE. A relationship between the LINK and the DAY TYPE entity may be used to express this kind of limited availability on parts of the public transport network.

8.3.1.5.3 Overtaking Possibility

In rail or wire systems, overtaking is only possible if an appropriate overtaking track is available. In bus systems, the situation of two buses regularly planned to overtake each other while operating on the same ROUTE LINK can be practically neglected. Consequently, the places where it is possible to overtake can be described by particular POINTs, as far as the planning domain is concerned. Most often SCHEDULED STOP POINTs will be used for this purpose in operational practice.

The entity OVERTAKING POSSIBILITY is therefore related to, and identified by, the INFRASTRUCTURE POINT which allows a vehicle stopping at this POINT to be overtaken by another vehicle passing by. The OVERTAKING POSSIBILITY specifies that this INFRASTRUCTURE POINT provides means (for instance a bus bay, or an overtaking rail) for one vehicle overtaking the other. This possibility may depend on the characteristics of the VEHICLE TYPEs in question, so the VEHICLE TYPEs of both the overtaking and the overtaken vehicle are associated with the OVERTAKING POSSIBILITY, by means of identifying relationships. If no link is done with a VEHICLE TYPE, this defines a general overtaking possibility in the infrastructure for any vehicle.

8.3.1.5.4 Meeting Restrictions

The entity MEETING RESTRICTION expresses that vehicles of two specified VEHICLE TYPEs are not allowed to meet on a particular pair of INFRASTRUCTURE LINKs (e.g. opposite tracks). In practice, this will probably occur mainly in tram systems, where several generations of tram vehicles are operating on the same rail network, with different vehicle widths leading to conflicting clearance profiles along certain parts of the track network. In metro or light rail systems, such a situation may occur if the network comprises single-track sections.

If no link is done with a VEHICLE TYPE, this defines a general restriction in the infrastructure for any vehicle.

8.3.1.5.5 Restricted Manœuvre

A particular characteristic of railway networks (in contrast to road networks) is the fact that the railway geometry does not always allow vehicle movement between two adjacent railway elements, for instance in the case of switches or crossings. Railway elements may not be suitable to be passed through in any arbitrary sequence and some successions may physically be impossible. This kind of restrictions is expressed by the entity

MANOEUVRE, specifying from which INFRASTRUCTURE LINK to which other (adjacent) element a rail vehicle cannot proceed because of physical restrictions. Additional information can be attached, for example the VEHICLE TYPEs for which a restricted MANOEUVRE would apply (for instance, bi-directional rail vehicles may be able to perform a certain manoeuvre whereas one-directional vehicles are not capable of it).

8.3.1.6 Network Restriction – Example

The following figure provides an example of a meeting restriction: two vehicles run their journey on opposite tracks, but due to the narrowing of the track, they are not able to meet on the two opposite red links.

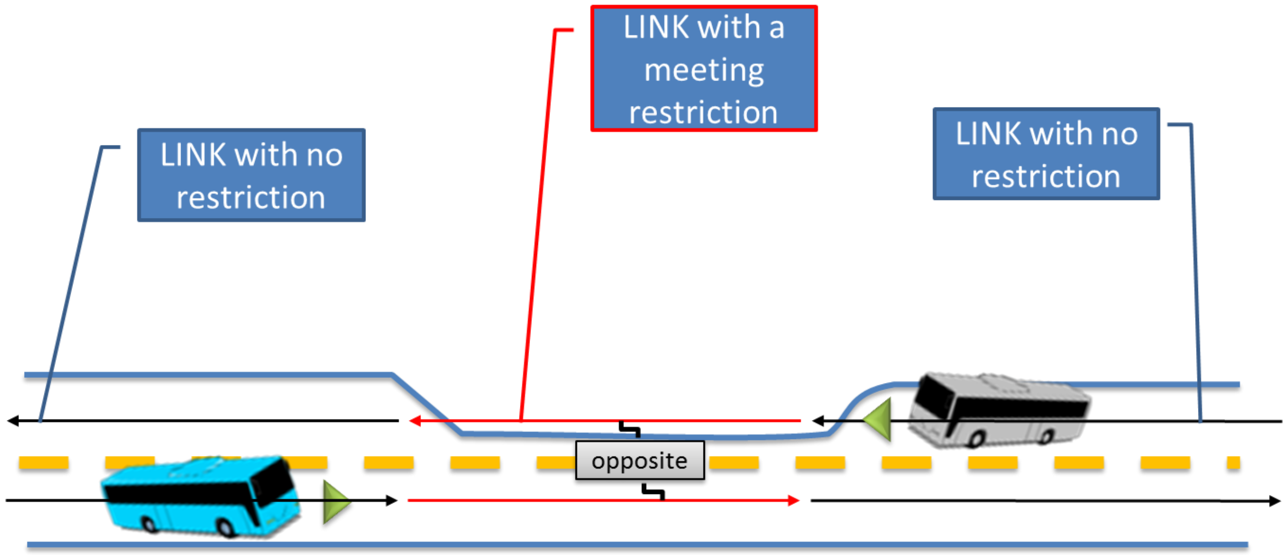


Figure 404 – Network Infrastructure example

8.3.1.7 Network Restriction – Physical Model

Figure 405 shows the physical model for the Network Restriction elements.

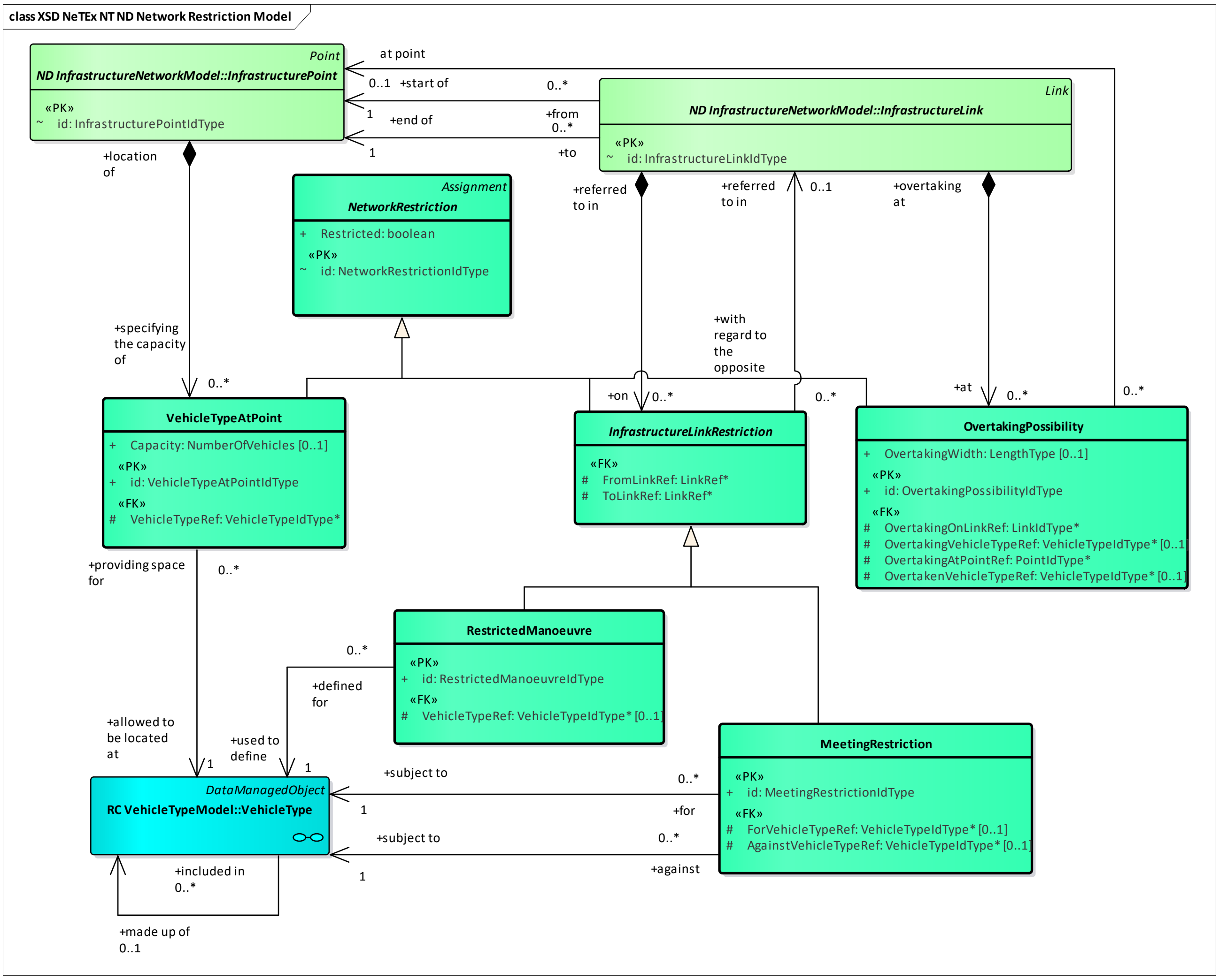


Figure 405 – Infrastructure Restriction – Physical Model (UML)

The physical model elements are as follows:

8.3.1.8 Network Restriction – Attributes and XSD

8.3.1.8.1 NetworkRestriction – Model Element

A constraint on use of elements of a network of INFRASTRUCTURE POINTs and INFRASTRUCTURE LINKs. This model introduces abstract classes (NETWORK RESTRICTION and INFRASTRUCTURE LINK RESTRICTION) that don't appear on the conceptual MODEL in order to simplify and enhance the representation of the different specialisations and their attributes.

Table 344 – *NetworkRestriction* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Assignment*	::>	NETWORK RESTRICTION inherits from ASSIGNMENT.
«PK»	***id***	*NetworkConstraintIdType*	1:1	Identifier of NETWORK RESTRICTION.

	Restricted	*xsd:boolean*	1:1	Whether the Restriction is allowed or forbidden. taken from frame.
	Description	*MultiingualString*	1:1	Description of NETWORK RESTRICTION.

MeetingRestriction and ***RestrictedManoeuvre*** are inheriting from ***NetworkRestriction*** and are connected to ***VehicleType*** in order to express limitations specific to some VEHICLE TYPEs. If the limitation applies to any vehicle, this is expressed by having no link with ***VehicleType.*** This is a work around to keep things simple (no relation with ***VehicleType*** means a relation with all ***VehicleType***). Alternatively an "*AnyVehicle*" ***VehicleType*** can be used.

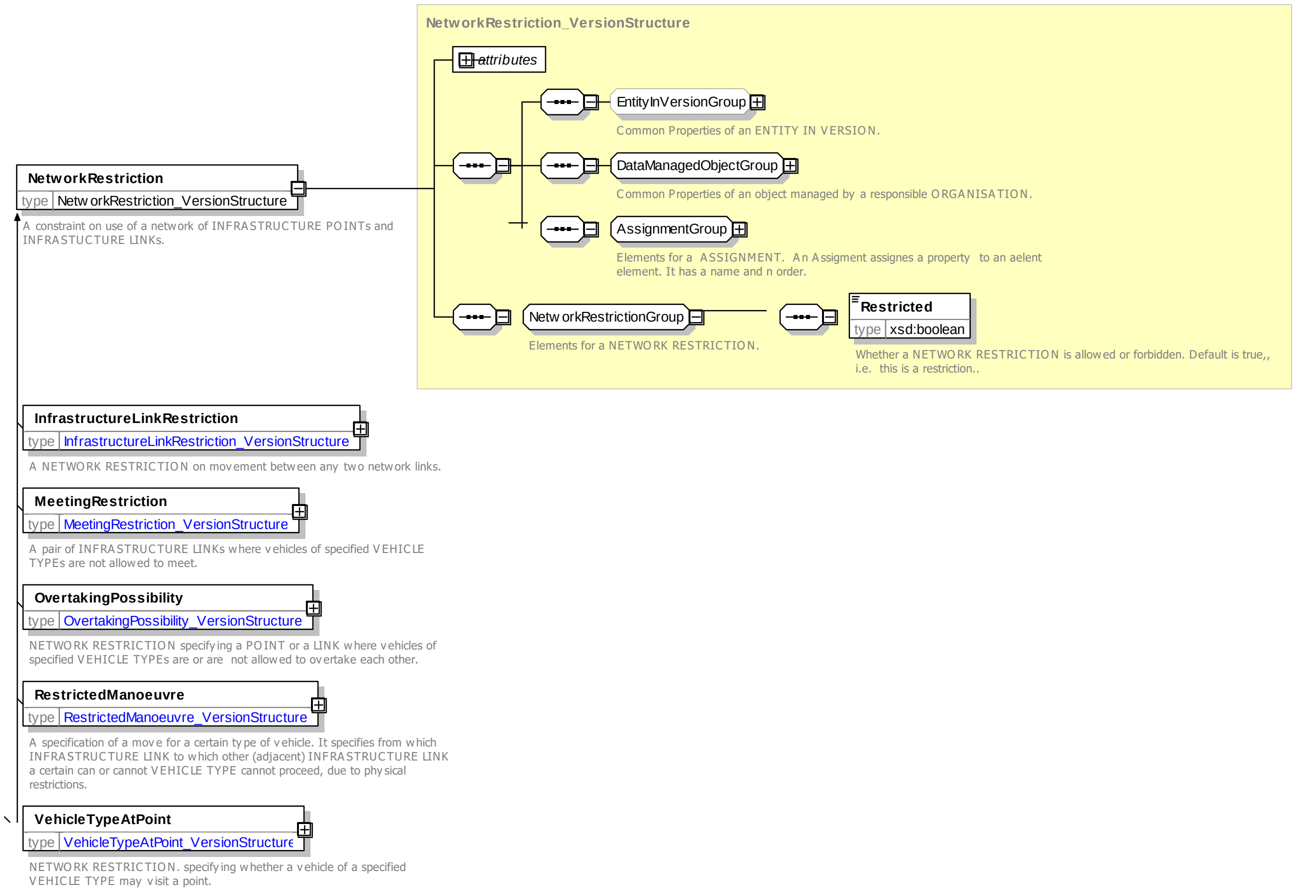


Figure 406 – *NetworkRestriction* – XSD

8.3.1.8.1.1 InfrastructureLinkRestriction – Model Element

A constraint on use of an INFRASTRUCTURE LINK.

Table 345 – *InfrastructureLinkRestriction* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*NetworkRestriction*	*::>*	INFRASTRUCTURE LINK RESTRICTION inherits from NETWORK RESTRICTION.

«FK»	***FromLinkRef***	*(LinkRef)* WireElement | RoadElement | RailwayElement	1:1	Reference to LINK at which NETWORK RESTRICTION starts.
«FK»	***ToLinkRef***	*(LinkRef)* WireElement | RoadElement | RailwayElement	1:1	Reference to LINK at which NETWORK RESTRICTION ends.

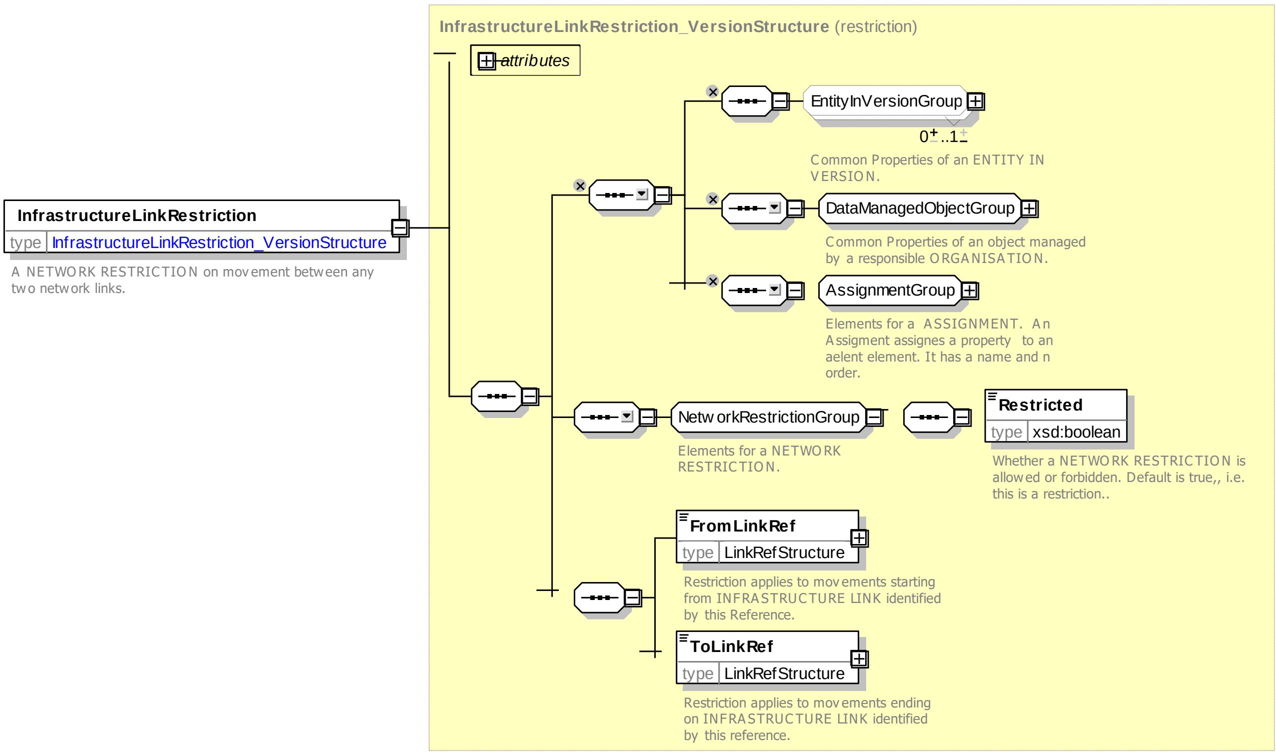


Figure 407 – *InfrastructureLinkRetriction*– XSD

8.3.1.8.1.2 RestrictedManoeuvre – Model Element

A specification of an impossible move for a certain type of vehicle. It specifies from which INFRASTRUCTURE LINK to which other (adjacent) INFRASTRUCTURE LINK a certain VEHICLE TYPE can or cannot proceed, due to physical restrictions.

Table 346 – *RestrictedManoeuvre* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*InfrastructureLinkRestriction*	::>	RESTRICTED MANOEUVRE inherits from INFRASTRUCTURE LINK RESTRICTION.
«PK»	***id***	Restricted*Manoeuvre-IdType*	1:1	Identifier of RESTRICTED MANOEUVRE.

«FK»	***VehicleTypeRef***	*VehicleTypeRef*	1:1	Reference to VEHICLE TYPE for RESTRICTED MANOEUVRE.

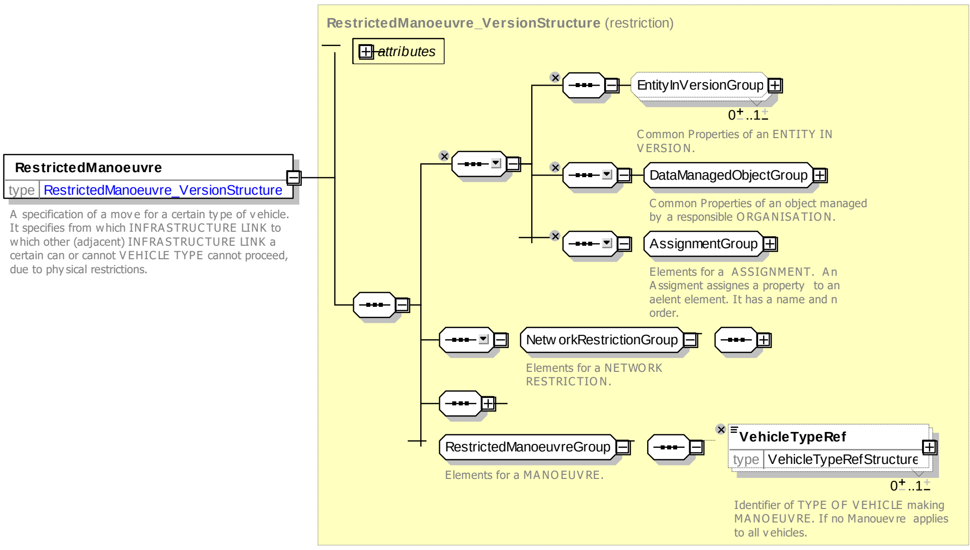


Figure 408 – *RestrictedManoeuvre* – XSD

8.3.1.8.1.3 MeetingRestriction – Model Element

A pair of INFRASTRUCTURE LINKs where vehicles of specified VEHICLE TYPEs are not allowed to meet.

Table 347 – *MeetingRestriction* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*InfrastructureLinkRestriction*</u>	::>	MEETING RESTRICTION inherits from INFEASTRUCTURE LINK RESTRICTION.
«PK»	***id***	*MeetingRestrictionIdType*	1:1	Identifier of MEETING RESTRICTION.
«FK»	***ForVehicleRef***	*VehicleTypeRef*	1:1	Reference to VEHICLE TYPE for RESTRICTION – forwards direction of LINK.
«FK»	***AgainstVehicleRef***	*VehicleTypeRef*	1:1	Reference to VEHICLE TYPE for RESTRICTION – backwards direction of LINK.

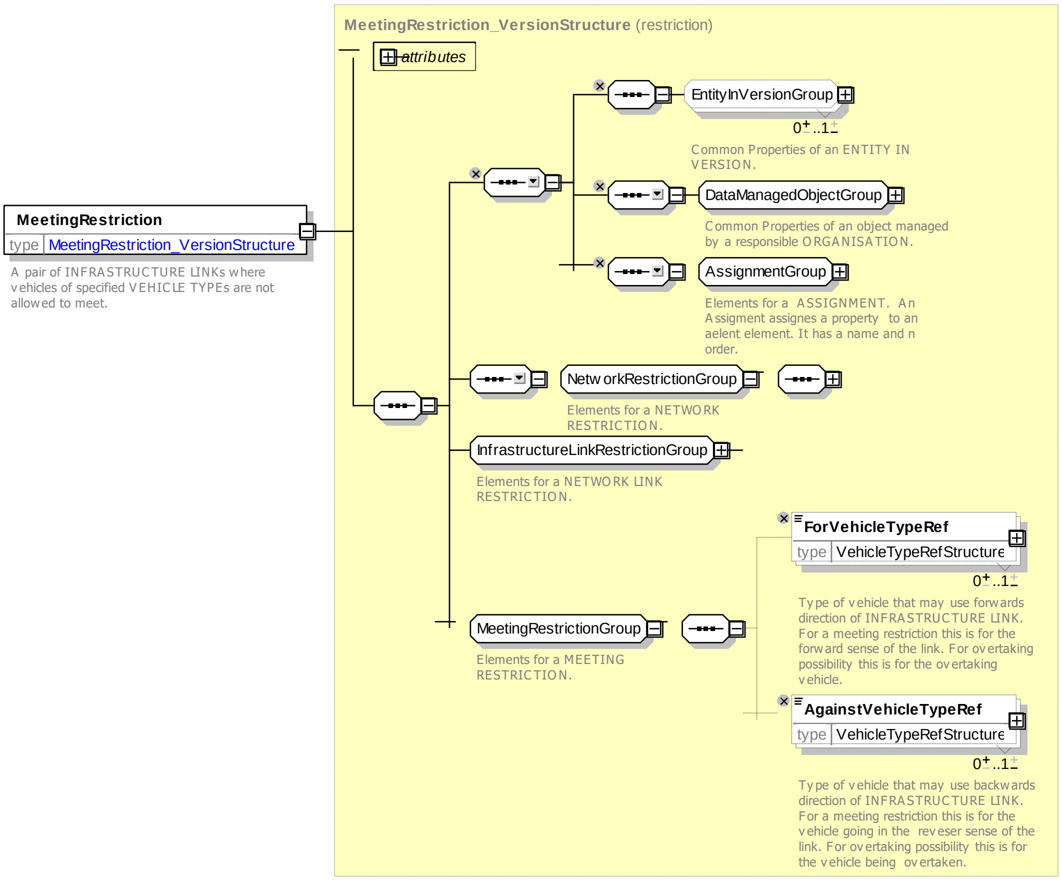


Figure 409 – *MeetingRestriction – XSD*

8.3.1.8.1.4 OvertakingPossibility – Model Element

Table 348 – *OvertakingPossibility* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*MeetingRestriction*</u>	::>	OVERTAKING POSSIBILITY inherits from MEETING RESTRICTION.
«PK»	***id***	*OvertakingPossibility-IdType*	1:1	Identifier of OVERTAKING RESTRICTION.
	OvertakingWidth	*LengthType*	0:1	Width at overtaking point.
«FK»	***OvertakingOnLinkRef***	*(LinkRef) WireElement | RoadElement | RailwayElement*	1:1	Reference to LINK at which OVERTAKING POSSIBILITY applies.
«FK»	***OvertakingAtPointRef***	*(PointRef) WirePoint | RoadPoint | RailwayPoint*	1:1	Reference to POINT at which OVERTAKING POSSIBILITY applies.
«FK»	***OvertakingVehicleTypeRef***	*VehicleTypeRef*	01:1	Type of VEHICLE that may overtake. If Blank any

«FK»	***OvertakenVehicle TypeRef***	*VehicleTypeRef*	0:1	Type of VEHICLE that may be overtaken. If Blank any

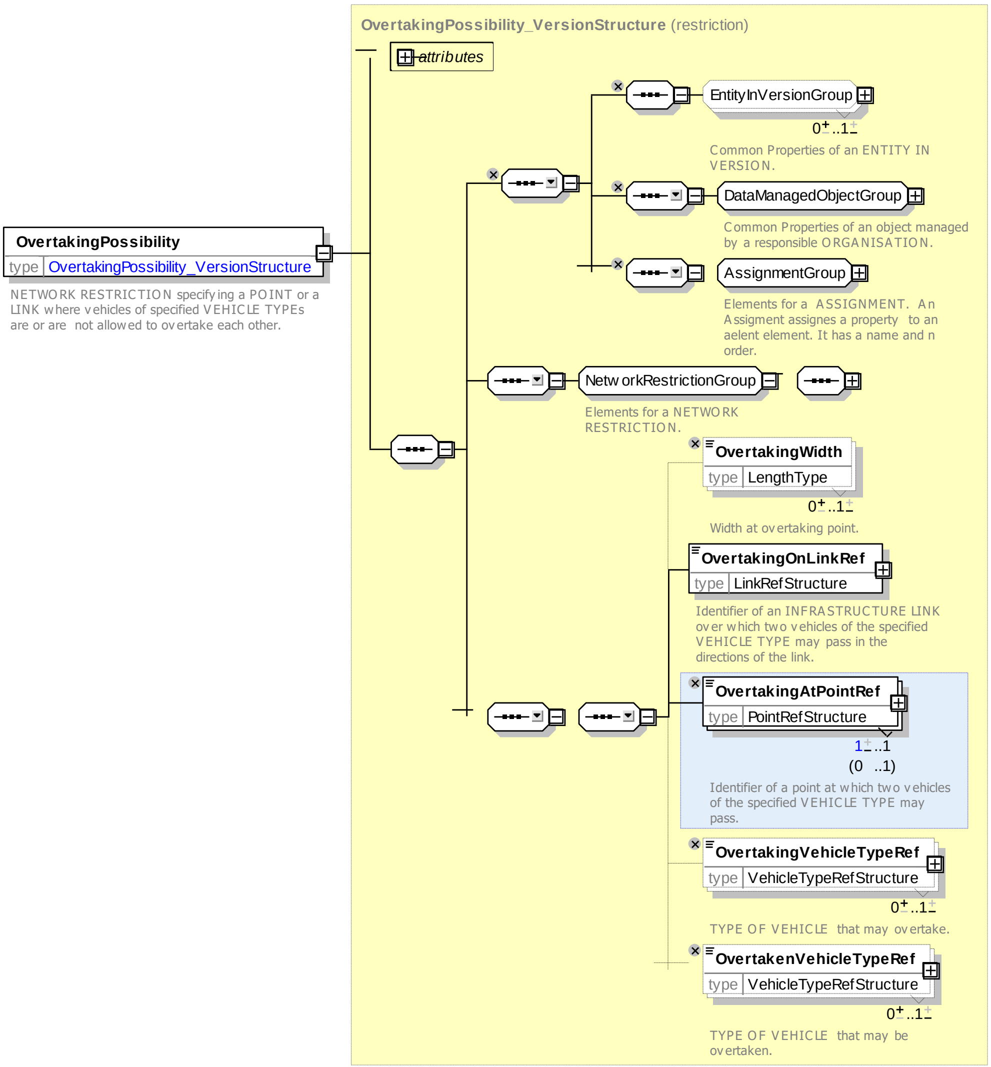


Figure 410 – *OvertakingPossibility* – XSD

8.3.1.9 XML Examples of Infrastructure Elements

8.3.1.9.1 XML Example of Road Junctions and Road Links

The following XML code fragment shows restrictions on the two ROAD ELEMENT links from the earlier example.

EXAMPLE

```
<restrictions>
    <RestrictedManoeuvre version="0022" id="hde:RestrictedManoeuvre :RM_01">
        <Restricted>true</Restricted>
        <FromLinkRef ref="gis:RoadElement:rdE001">EXTERNAL</FromLinkRef>
        <ToLinkRef ref="gis:RoadElement:rdE002">EXTERNAL</ToLinkRef>
        <VehicleTypeRef version="002" ref="hde:VehicleType:VT001"/>
    </RestrictedManoeuvre >
    <MeetingRestriction version="002" id="hde:MeetingRestriction:RMT_02">
```

```xml
                <Restricted>false</Restricted>
                <ForVehicleTypeRef version="002" ref="hde:VehicleType:VT001"/>
                <AgainstVehicleTypeRef version="002" ref="hde:VehicleType:VT001"/>
                <FromLinkRef ref="gis:RoadElement:rdE001">EXTERNAL</FromLinkRef>
                <ToLinkRef ref="gis:RoadElement:rdE002">EXTERNAL</ToLinkRef>
            </MeetingRestriction>
            <OvertakingPossibility version="002" id="hde:OvertakingPossibility:ROP_03">
                <Restricted>true</Restricted>
                <OvertakingOnLinkRef version="any" ref="gis:RoadElement:rdE001/>
                <OvertakingAtPointRef version="any" ref="gis:pt0021"/>
                <OvertakingVehicleTypeRef version="002" ref="hde:VehicleType:VT001"/>
                <OvertakenVehicleTypeRef version="002" ref="hde:VehicleType:VT001"/>
            </OvertakingPossibility>
        </restrictions>
        <vehicleTypes>
            <VehicleType version="002" id="hde:VehicleType:VT001">
                <Name>Big red bus</Name>
                <PassengerCapacity>
                    <SeatingCapacity>50</SeatingCapacity>
                    <StandingCapacity>20</StandingCapacity>
                    <WheelchairPlaceCapacity>2</WheelchairPlaceCapacity>
                </PassengerCapacity>
                <Length>20</Length>
            </VehicleType>
        </vehicleTypes> ::>…
```

8.3.2 Activation

The ACTIVATION MODEL relates the points in the network at which monitoring equipment may interact with VEHICLEs - possibly with on-board equipment. Such equipment may be relevant for real-time control. Uses of ACTIVATED EQUIPMENT can include:

— Detectors for vehicle locating systems.

— Traffic light priority systems.

— Sign cleardown: the use of vehicle to infrastructure wireless communication to trigger sign content update based on proximity (providing a faster and more reliable link than a hub based signal).

8.3.2.1 ACTIVATION – Conceptual MODEL

The Activation model (Figure 411) describes a geometry made up of ACTIVATION POINTs, ACTIVATION LINKs, and TRAFFIC CONTROL POINTs. ACTIVATED EQUIPMENT may be assigned to the points of the network using ACTIVATION ASSIGNMENTs.

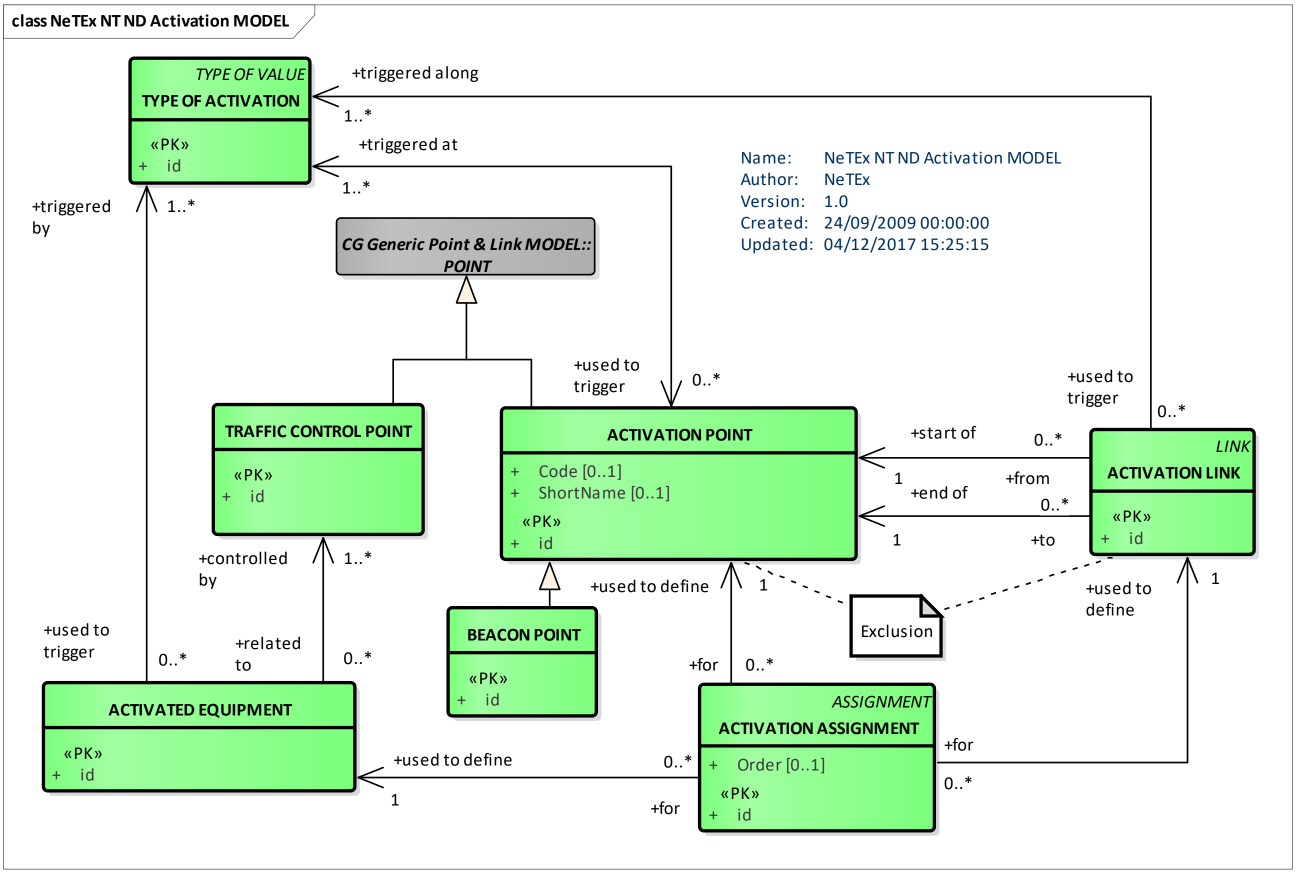


Figure 411 – ACTIVATION – Conceptual MODEL (UML)

8.3.2.2 Activation – Physical Model

Figure 412 shows the physical representation of the ACTIVATION model.

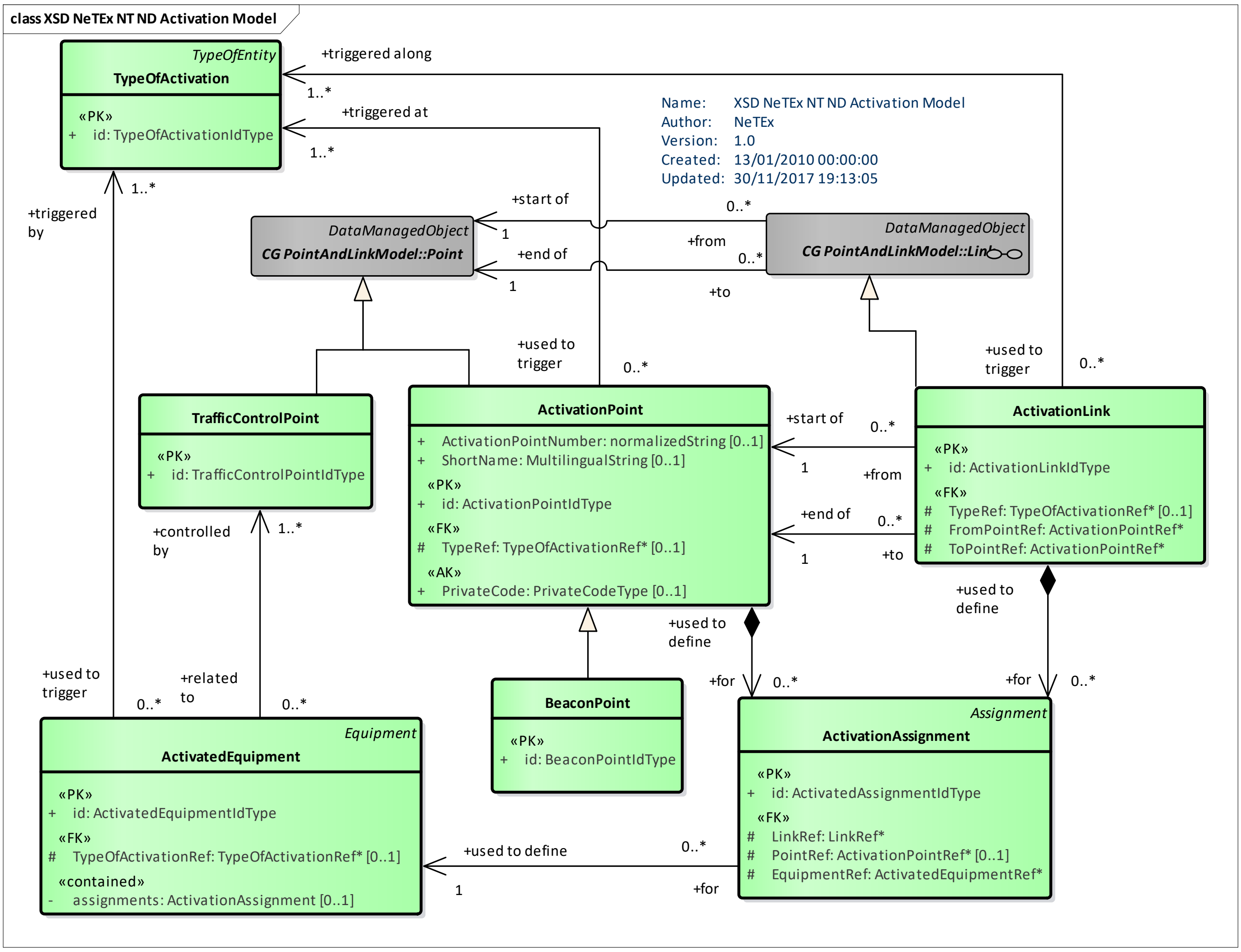


Figure 412 – Activation – Physical Model (UML)

8.3.2.3 Activation – Attributes and XSD

8.3.2.3.1 ActivationPoint – Model Element

A POINT where a control process is activated when a vehicle passes.

Table 349 – *ActivationPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Point*</u>	::>	ACTIVATION POINT inherits from POINT
«PK»	***id***	*ActivationPointIdType*	1:1	Identifier of ACTIVATION POINT.
	Code	*anyType*	0:1	Code of ACTIVATION POINT.

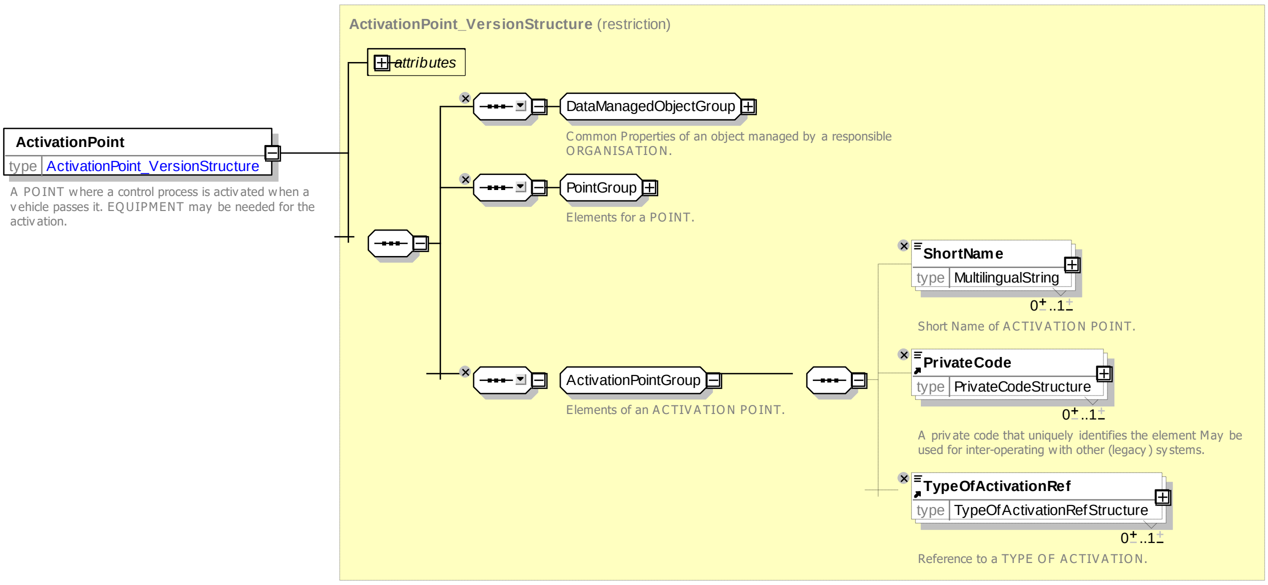


Figure 413 – *ActivationPoint – XSD*

8.3.2.3.2 ActivationLink – Model Element

A LINK where a control process is activated when a vehicle passes it.

Table 350 – *ActivationLink* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Link*	::>	ACTIVATION LINK inherits from LINK
«PK»	***id***	*ActivationLinkIdType*	1:1	Identifier of ACTIVATION LINK.
«FK»	***TypeRef***	*TypeOfActivationRef*	0:1	Type of ACTIVATION LINK.
«FK»	***FromPointRef***	*ActivationPointRef*	1:1	Reference to ACTIVATION POINT at which ACTIVATION LINK starts.
«FK»	***ToPointRef***	*ActivationPointRef*	1:1	Reference to ACTIVATION POINT at which ACTIVATION LINK ends.

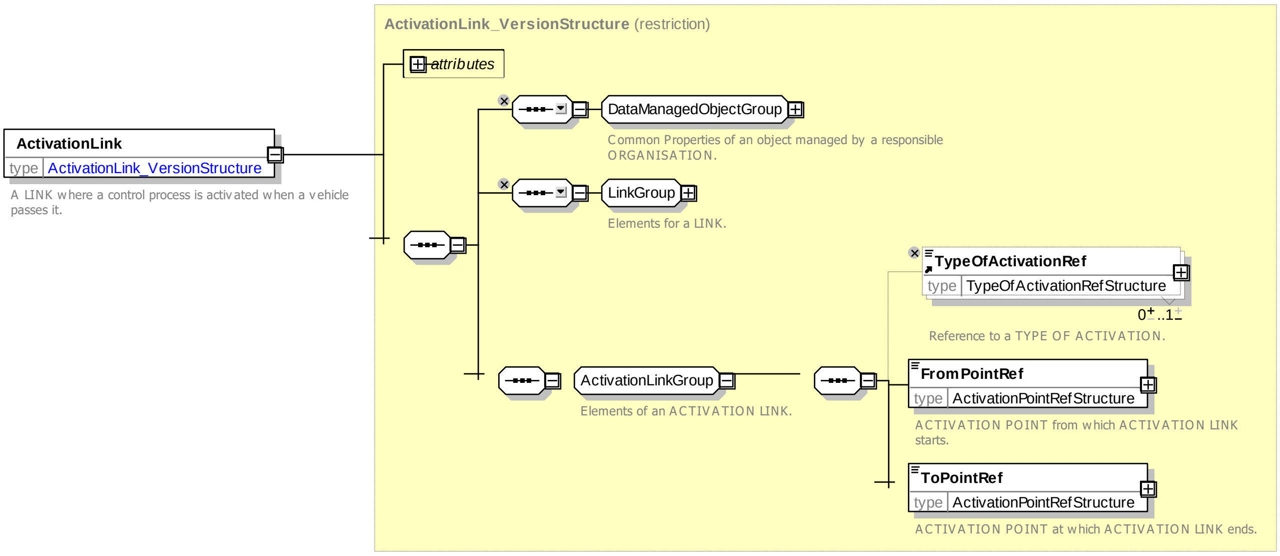


Figure 414 – *ActivationLink* – XSD

8.3.2.3.3 BeaconPoint – Model Element

A POINT where a beacon or similar device to support the automatic detection of vehicles passing by is located.

Table 351 – *BeaconPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*ActivationPoint*</u>	::>	BEACON LINK inherits from ACTIVATION POINT.
«PK»	***id***	*BeaconPointIdType*	1:1	Identifier of BEACON POINT.

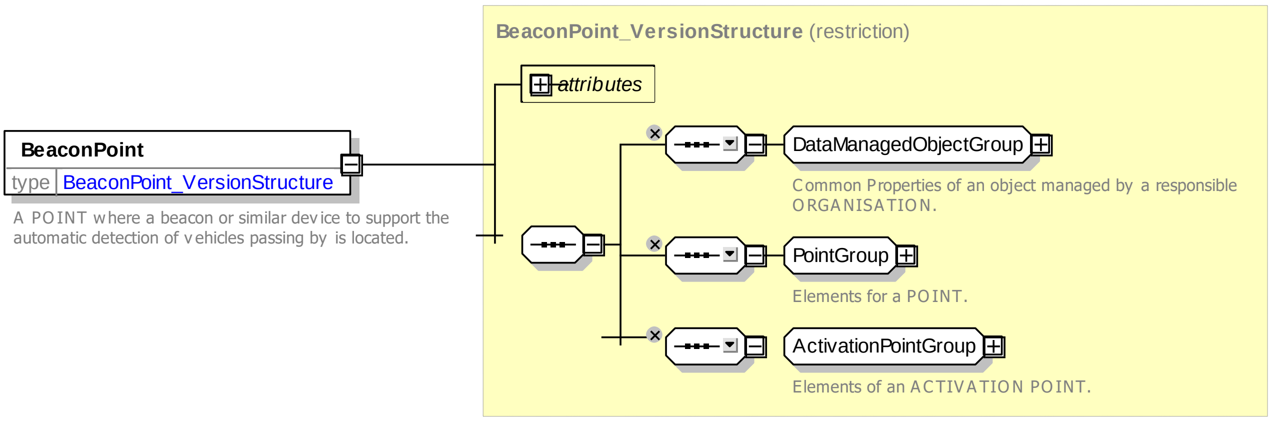


Figure 415 – *BeaconPoint – XSD*

8.3.2.3.4 TrafficControlPoint – Model Element

A POINT where the traffic flow can be influenced. Examples are: traffic lights (lanterns), barriers.

Table 352 – *TrafficControlPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Point*	::>	TRAFFIC CONTROL POINT inherits from POINT.
«PK»	***id***	*TrafficControlPointIdType*	1:1	Identifier of TRAFFIC CONTROL POINT.

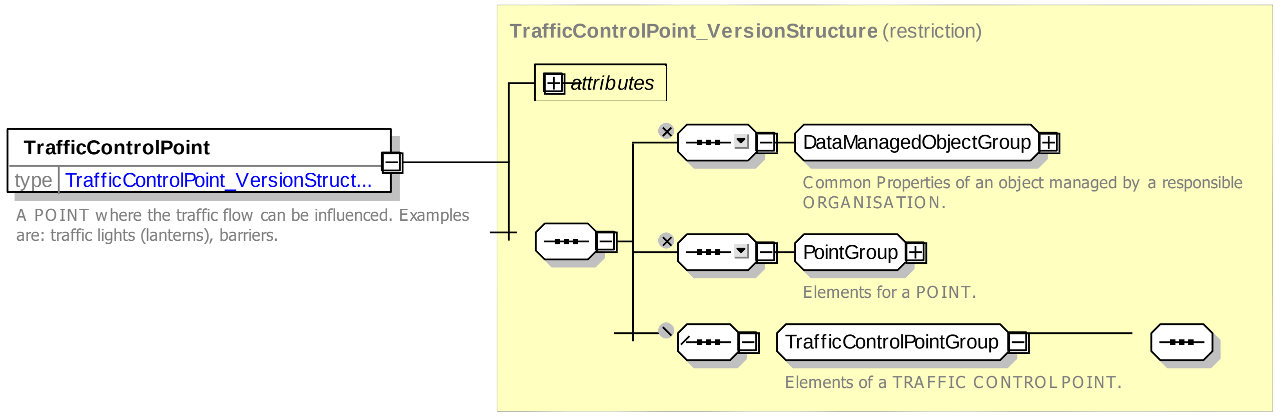


Figure 416 – *TrafficControlPoint* – XSD

8.3.2.3.5 ActivationAssignment – Model Element

An assignment of an ACTIVATION POINT/LINK to an ACTIVATED EQUIPMENT related on its turn to a TRAFFIC CONTROL POINT. The considered ACTIVATION POINT/LINK will be used to influence the control process for that TRAFFIC CONTROL POINT (e.g. to fix priorities as regards the processing of competing requests from different ACTIVATION POINTs/LINKs).

Table 353 – *ActivationAssignment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Assignment*	::>	ACTIVATION ASSIGNMENT inherits from ASSIGNMENT.
«PK»	***id***	*ActivatedAssignmentIdType*	1:1	Identifier of ACTIVATION ASSIGNMENT.
«FK»	***LinkRef***	(*LinkRef*)	1:1	Reference to LINK associated with of ACTIVATION ASSIGNMENT.
«FK»	***PointRef***	*ActivationPointRef* | *BeaconPointRef*	0:1	Reference to POINT associated with ACTIVATION ASSIGNMENT.
«FK»	***EquipmentRef***	*ActivatedEquipmentRef*	1:1	Reference to EQUIPMENT for ACTIVATION ASSIGNMENT.

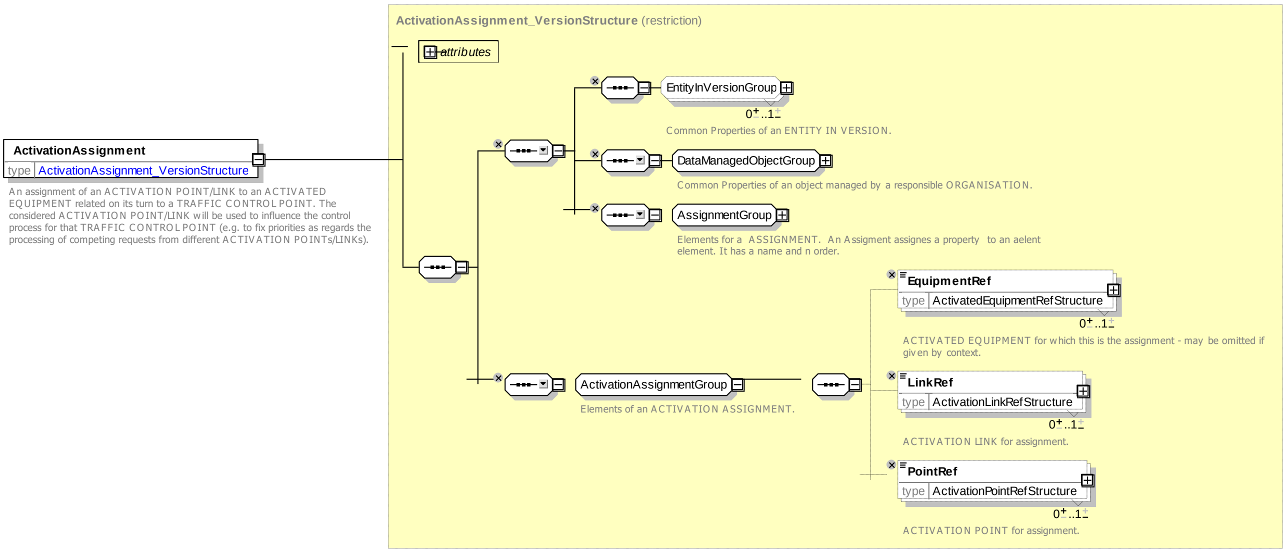


Figure 417 – *ActivationAssignment* – *XSD*

8.3.2.3.6 *ActivatedEquipment – Model Element*

An equipment activated by the passage of a vehicle at an ACTIVATION POINT or on an ACTIVATION LINK.

Table 354 – *ActivatedEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Equipment*	::>	ACTIVATED EQUIPMENT inherits from EQUIPMENT.
«PK»	***id***	*ActivatedEquipment-IdType*	1:1	Identifier of ACTIVATED EQUIPMENT.
«FK»	***TypeOfActivationRef***	*TypeOfActivationRef*	0:1	Reference to TYPE OF ACTIVATION.
«cntd»	***assignments***	*ActivationAssignment*	0:1	ACTIVATION ASSIGNMENTs for equipment.

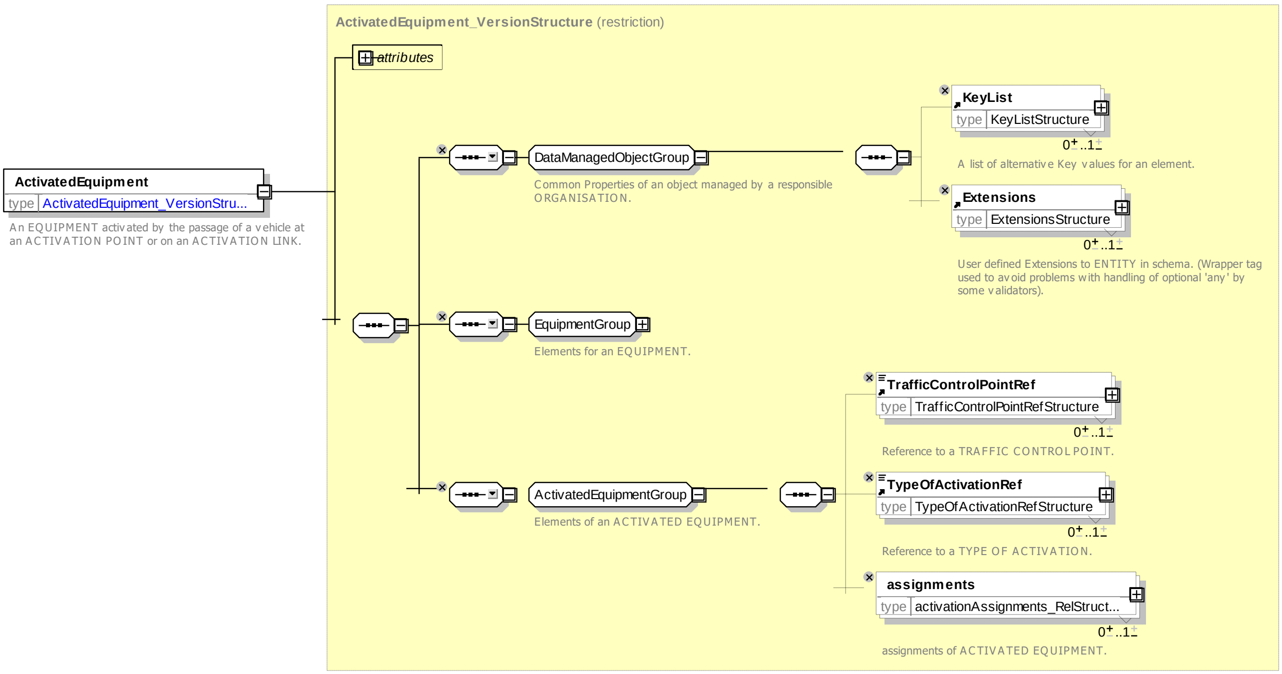


Figure 418 – *ActivatedEquipment* – *XSD*

8.3.2.3.7 TypeOfActivation – Model Element

A classification of real-time processes that are activated when vehicles passes an ACTIVATION POINT or an ACTIVATION LINK. See TYPE OF VALUE for XSD diagram.

Table 355 – *TypeOfActivation* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF ACTIVATION inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfActivationIdType*	1:1	Identifier of TYPE OF ACTIVATION.

8.3.3 Vehicle & Crew Point

The Vehicle & Crew Point model describes the location of the vehicle and crew changeover points that are referenced by vehicle and duty schedules.

8.3.3.1 VEHICLE & CREW POINT – Conceptual MODEL

Figure 419 shows the Conceptual MODEL for VEHICLE & CREW POINTs. There are three types of POINT: RELIEF POINT, PARKING POINT and GARAGE POINT at which VEHICLEs may be located. These may be associated with organisation entities for staff – CREW BASE – and for vehicle scheduling – GARAGE.

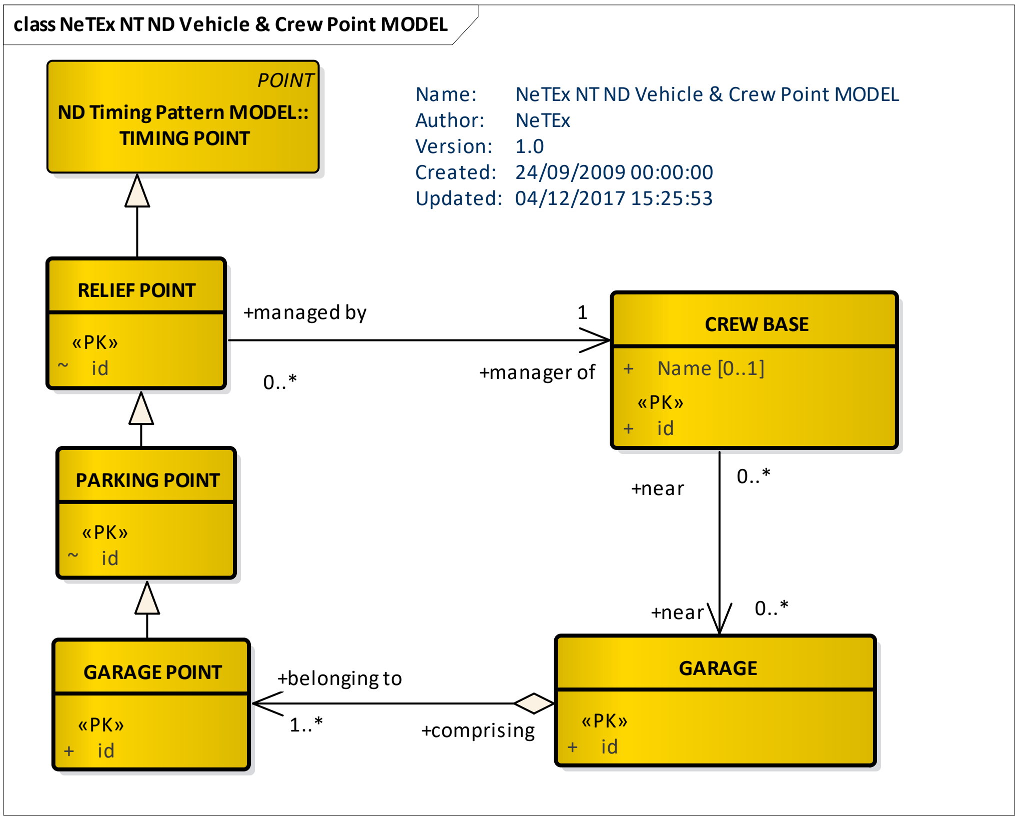


Figure 419 – VEHICLE & CREW POINT – Conceptual MODEL (UML)

8.3.3.2 Vehicle & Crew Point – Physical Model

Figure 420 shows the physical model for VEHICLE & CREW POINTs.

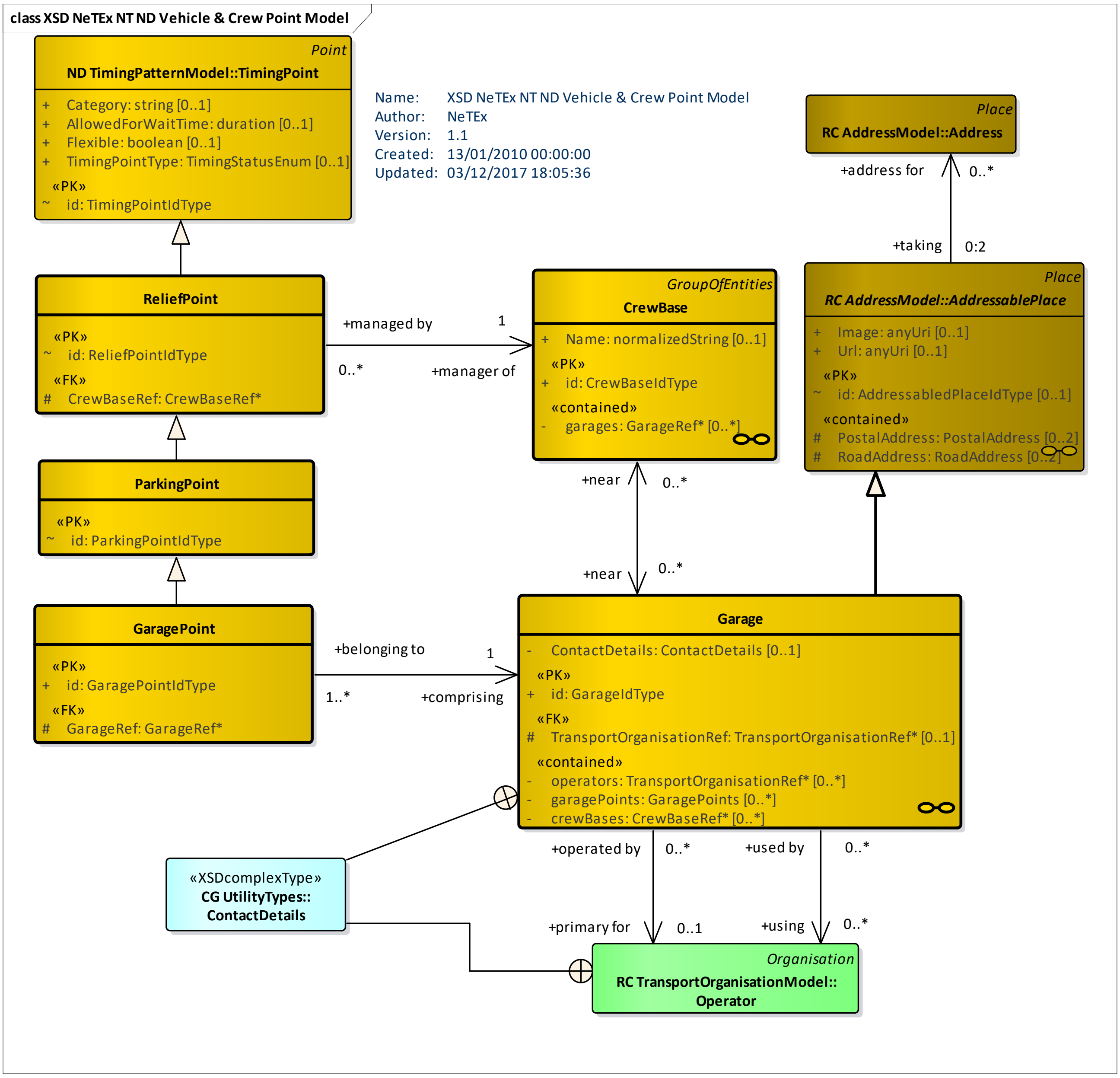


Figure 420 – Vehicle & Crew Point – Physical Model (UML)

8.3.3.3 Vehicle & Crew Point – Attributes and XSD

8.3.3.3.1 CrewBase – Model Element

A place where operating Employees (e.g. drivers) report on and register their work.

Table 356 – *CrewBase* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*DataManagedObject*	::>	CREW BASE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*CrewBaseIdType*	1:1	Identifier of CREW BASE.
	Name	*xsd:normalizedString*	0:1	Name of CREW BASE.
«cntd»	***garages***	*GarageRef*	0:*	Garages associated with CREW BASE.

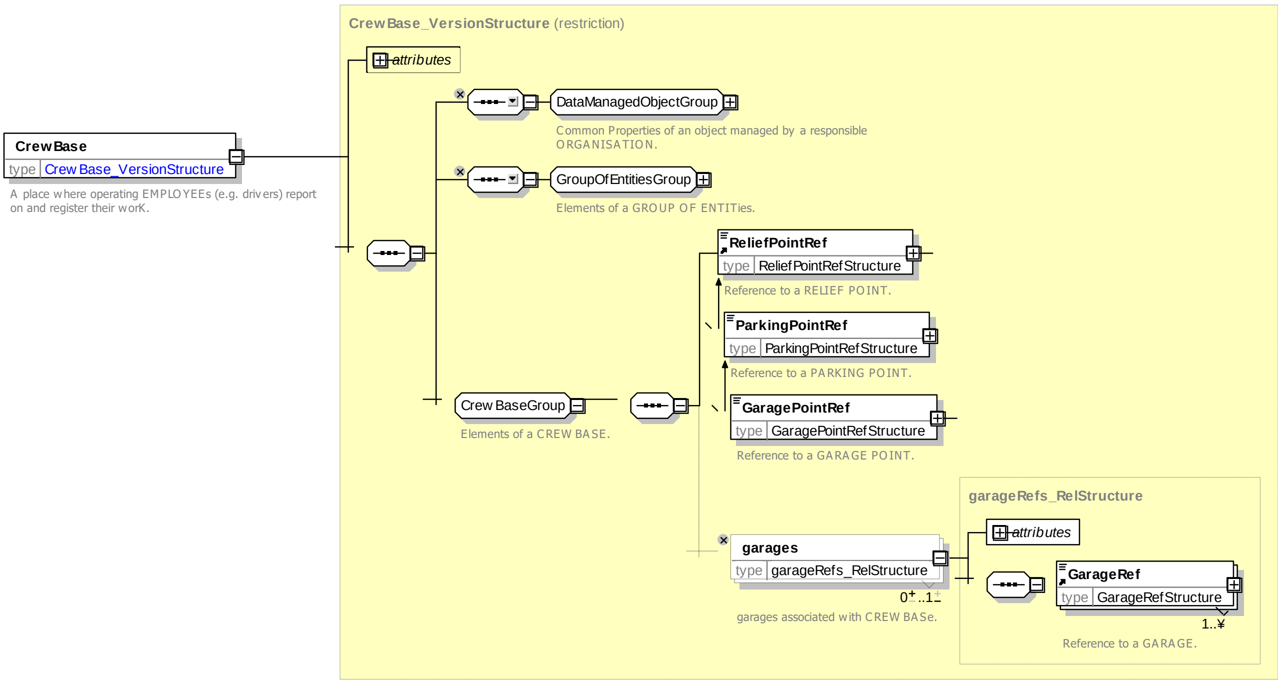


Figure 421 – *CrewBase* – XSD

8.3.3.3.2 Garage – Model Element

A facility used for parking and maintaining vehicles. PARKING POINTs in a GARAGE are called GARAGE POINTs.

Table 357 – *Garage* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*AddressablePlace*	::>	GARAGE inherits from ADDRESSABLE PLACE.
«PK»	***id***	*GarageIdType*	1:1	Identifier of GARAGE.
«cntd»	***ContactDetails***	*ContactDetails*	0:1	Contact details for GARAGE. +V1.1
«FK»	***Transport-OrganisationRef***	*(TransportOrganisationRef) AuthorityRef | OperatorRef*	0:1	Reference to OPERATOR of GARAGE.
«cntd»	***operators***	*(TransportOrganisationRef) AuthorityRef | OperatorRef*	0:*	Other TRANSPORT OPERATORs associated with GARAGE. +v1.1

«cntd»	***garagePoints***	*GaragePoint* | *GaragePointRef*	0:*	GARAGE POINTs associated with GARAGE.
«cntd»	***crewBases***	*CrewBaseRef*	0:*	CREW BASEs associated with GARAGE.

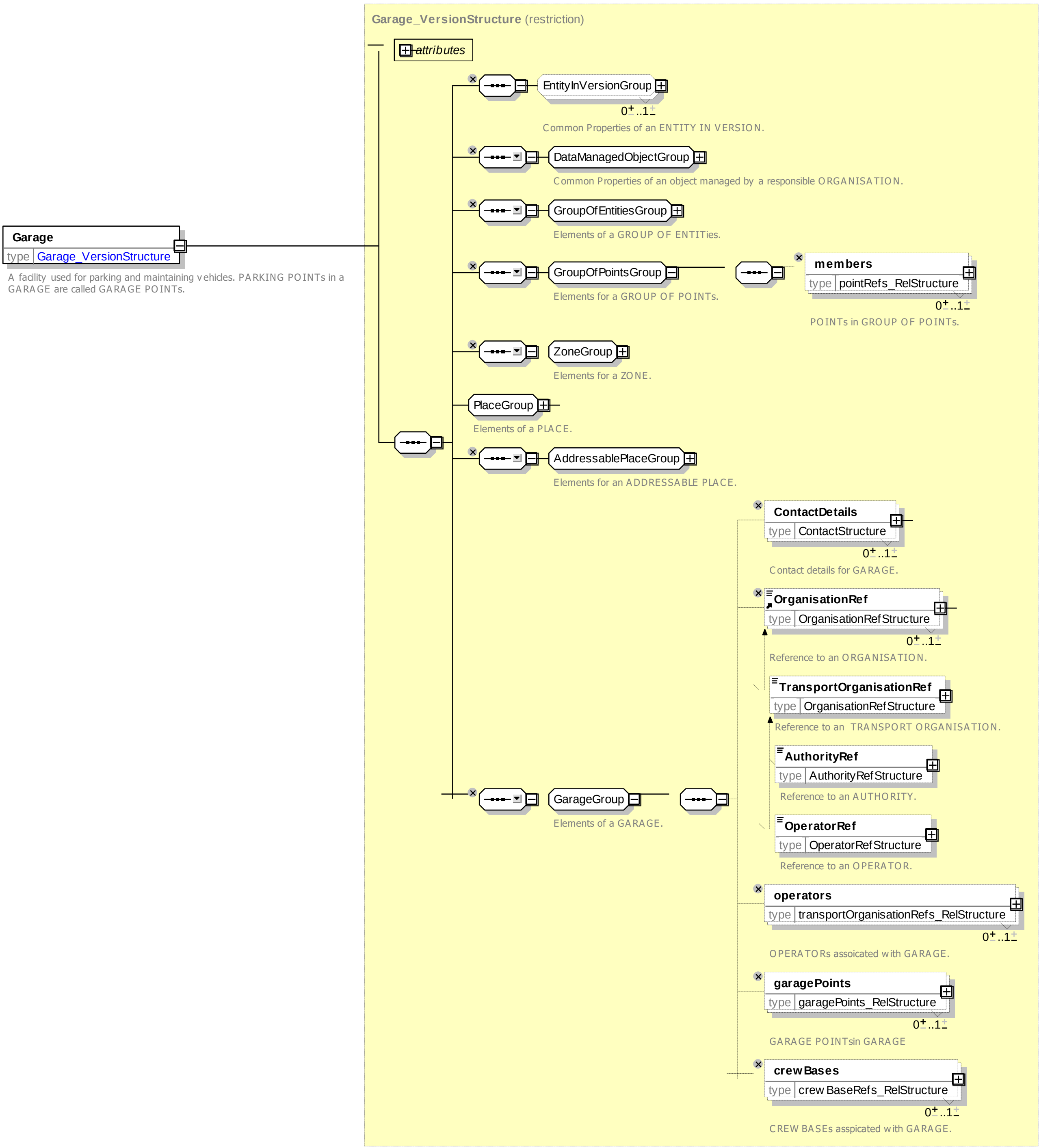


Figure 422 – *Garage* – XSD

8.3.3.3.3 ReliefPoint – Model Element

A TIMING POINT where a relief is possible, i.e. a driver may take on or hand over a vehicle. The vehicle may sometimes be left unattended.

Table 358 – *ReliefPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TimingPoint*	::>	RELIEF POINT inherits from TIMING POINT.
«PK»	***id***	*ReliefPointIdType*	1:1	Identifier of RELIEF POINT.
«FK»	***CrewBaseRef***	*CrewBaseRef*	0:1	Reference to a CREW BASE associated with RELIEF POINT.

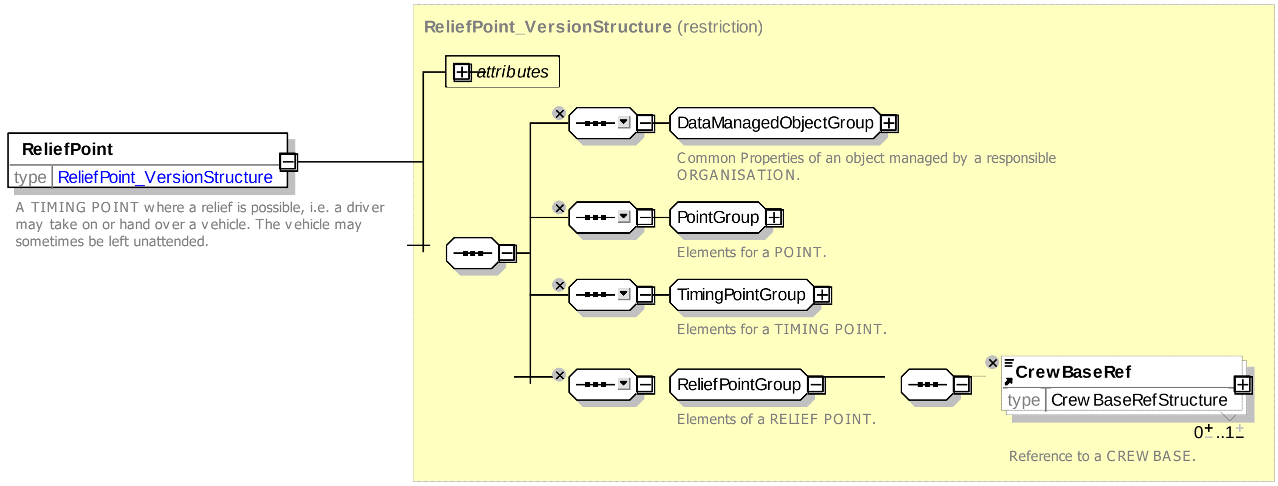


Figure 423 – *ReliefPoint* – XSD

8.3.3.3.4 *ParkingPoint – Model* Element

A TIMING POINT where vehicles may stay unattended for a long time. A vehicle's return to park at a PARKING POINT marks the end of a BLOCK.

Table 359 – *ParkingPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*ReliefPoint*	::>	PARKING POINT inherits from RELIEF POINT.
«PK»	***id***	*ParkingPointIdType*	1:1	Identifier of PARKING POINT.

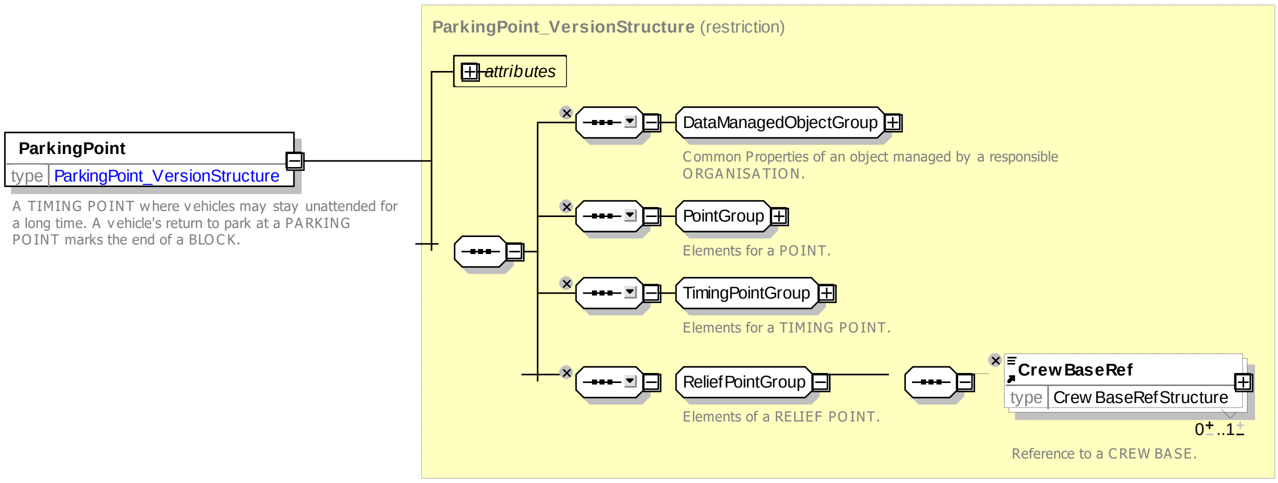


Figure 424 – *ParkingPoint* – XSD

8.3.3.3.5 GaragePoint – Model Element

A subtype of PARKING POINT located in a GARAGE.

Table 360 – *GaragePoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*GaragePoint*	::>	GARAGE POINT inherits from RELIEF POINT.
«PK	***id***	*GaragePointIdType*	1:1	Identifier of GARAGE POINT.
«FK»	***GarageRef***	*GarageRef*	1:1	Reference to a GARAGE associated with GARAGE POINT.

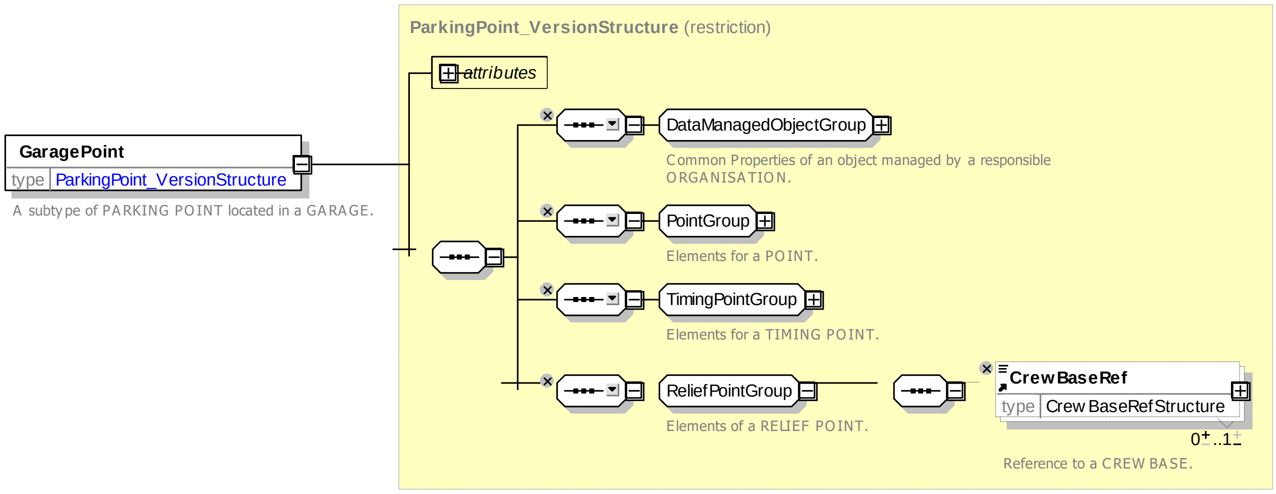


Figure 425 – *GaragePoint* – XSD

8.3.4 Lines and Routes

8.3.4.1 ROUTE – Conceptual MODEL

NOTE The following explanations use excerpts from Transmodel.

The ROUTE submodel is the same as the one described by Transmodel: the ROUTE entity represents a conventional way of describing a path through the network, to be used by regular PT services. A ROUTE is a linear feature composed of points and links specifically defined for that purpose. This sequence of points and links must be built in a way that identifies a path without any ambiguity.

The ROUTE entity represents an abstract concept that has in itself no real operational meaning. Its purpose is to describe a path independently of both the infrastructure pattern (e.g. ROAD ELEMENTs or RAILWAY ELEMENTs) and the operational pattern (e.g. sequence of SCHEDULED STOP POINTs). ROUTE is classically used as an interfacing object between operational planning and infrastructure description. The independence of the ROUTE definition serves to separate the concerns of the different layers allowing a modular exchange of data.

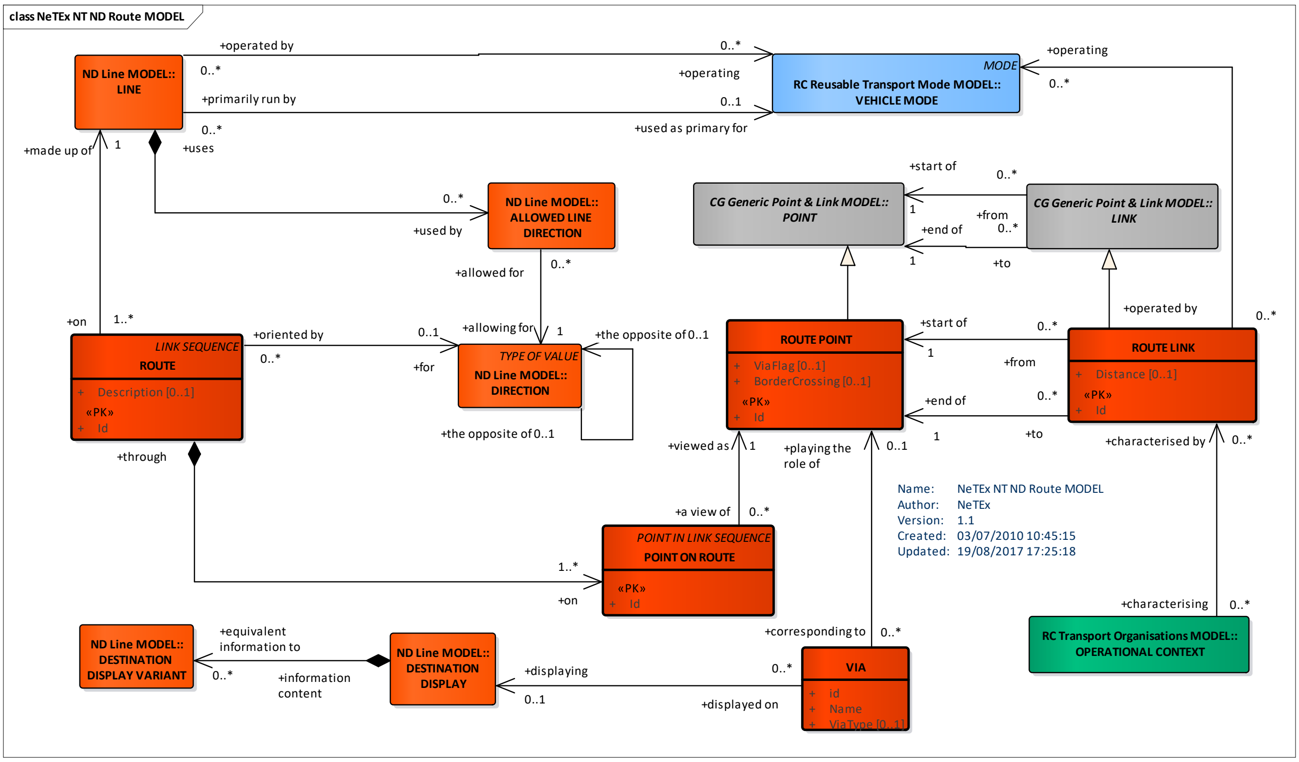


Figure 426 –Route – Conceptual MODEL (UML)

A ROUTE is made up of ROUTE LINKs, which are LINKs defined between two ROUTE POINTs. A ROUTE LINK is restricted to be identifiable by its end ROUTE POINTs, which means that there cannot normally be any alternative ROUTE LINK between the same pair of ROUTE POINTs. This restriction corresponds to most practices, but if necessary can be qualified by the use of an OPERATIONAL CONTEXT (see TRANSPORT ORGANISATION Model), which allows separate links for separate designated purposes.

A ROUTE is thus a LINK SEQUENCE, defined by an ordered sequence of (two or more) POINTs ON ROUTE. A ROUTE may pass through the same ROUTE POINT more than once, as in the case of a loop. The POINT ON ROUTE entity is accordingly used to describe the ordered list of ROUTE POINTs defining the path of a ROUTE, with an attribute ‘order’ as identifier.

It should be noted that a ROUTE – as a single path through a network in one direction - corresponds to only one of the possible senses of ‘route’ in colloquial English. In particular the wider sense of a set of paths including branches and conditional variants given a common name for marketing to the public, is represented by the concept of a LINE,

The LINE & ROUTE model above gives an overview on all the relevant concepts in this context. They will be explained in the following sections.

8.3.4.2 Route Topologies

A number of different geometries for routes are typically found in transport networks, all of which may be described using POINT and LINK representations:

- **Linear**: A simple linear path from an origin stop to a destination stop. It may be exactly symmetric i.e. be traversed to matching stop pairs in the outbound and inbound direction. Or asymmetric – with differences in the stop sequences in each direction.

- **Circular** : A path that returns to the origin stop as the destination. It then may continue round repeatedly. There may be symmetric or asymmetric services in the clockwise or anticlockwise direction. The destinations shown for such routes may vary along the way.

- **Lollipop**: A path that goes round a loop one way at the outbound destination end and then returns past the same stops on the inbound path.

- **Cloverleaf**: A path that returns repeatedly to the same stop.

- **Branching**: Alternate paths that go one or other alternative way at either end of the journey.

- **Eye**: Alternate paths that go one or other alternative way round an intermediate section of the route.

There must be a valid ROUTE LINK between each pair of consecutive POINTs ON ROUTE.

The general orientation of a ROUTE (a ROUTE is of course oriented) may be described by an expression like “outwards”, “backwards” etc., often referring approximately to the city centre. This classification may lead to the definition of arbitrarily chosen DIRECTIONs, which may be used for passenger information, but may also be relevant for scheduling or fare collection. Two DIRECTIONs may be defined as being opposite to each other.

8.3.4.3 Route – Example

The following figure shows a ROUTE determined through a sequence of POINTs ON ROUTE. It should clarify in particular, the difference between the Infrastructure Network (streets) and the schematic representation of the physical path for vehicles: the ROUTE.

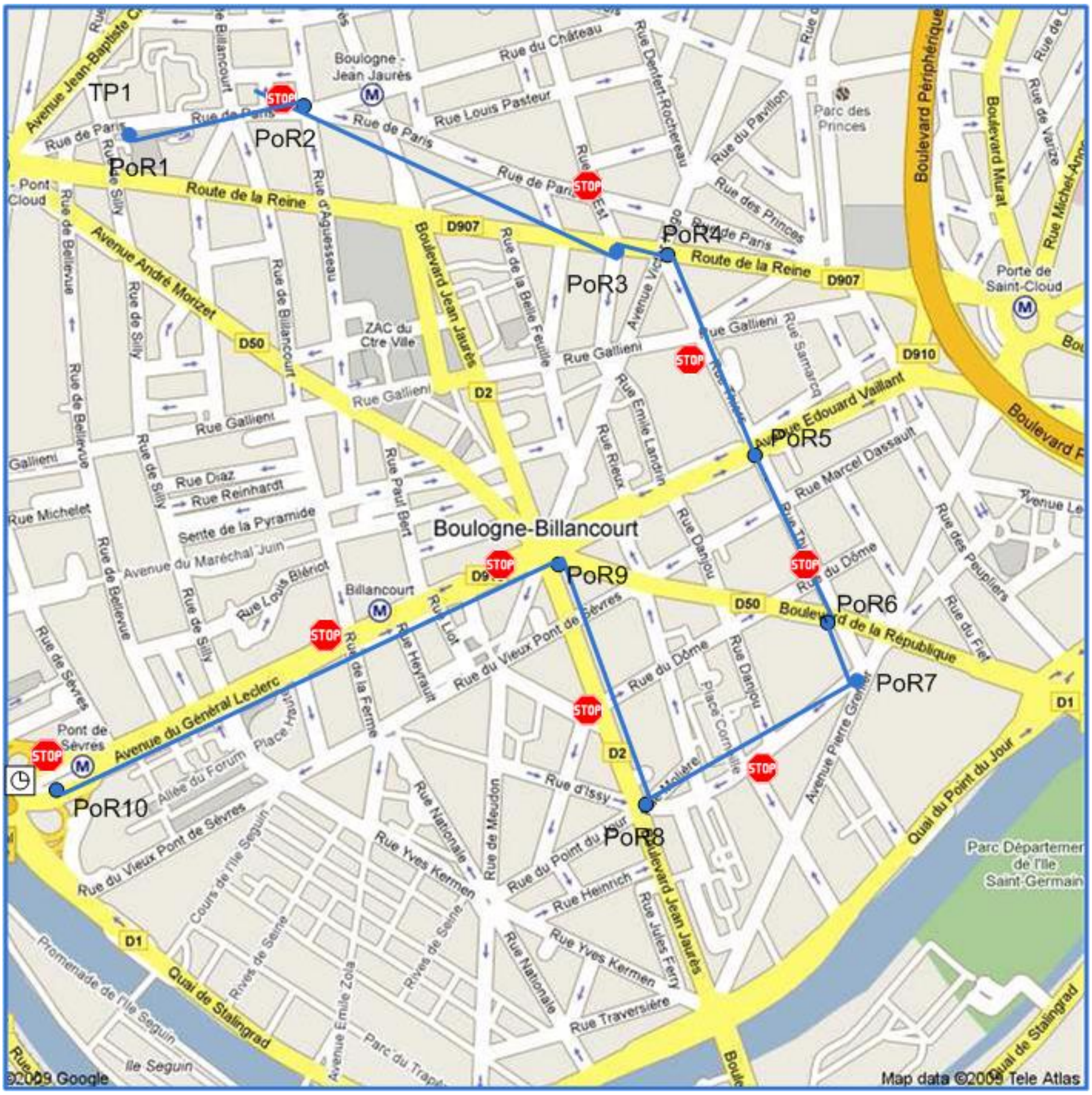


Figure 427 – ROUTE POINT & POINT ON ROUTE – Example

Figure 428 below shows an example of ROUTE POINTs used by two ROUTEs, with ROUTE 1 (the green one) passing several times at the same ROUTE POINT. Each time the ROUTE passes through a ROUTE POINT, it "creates" a POINT ON ROUTE.

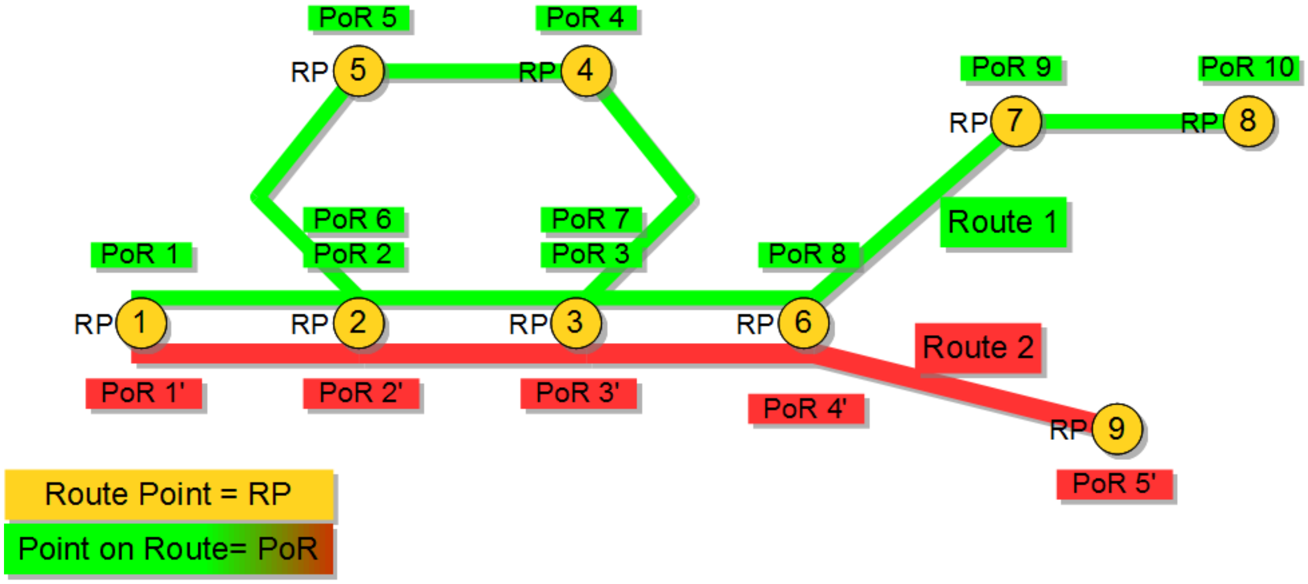


Figure 428 – ROUTE POINT & POINT ON ROUTE – Example

8.3.4.4 Route – Physical Model

Figure 429 shows the Physical model for a ROUTE.

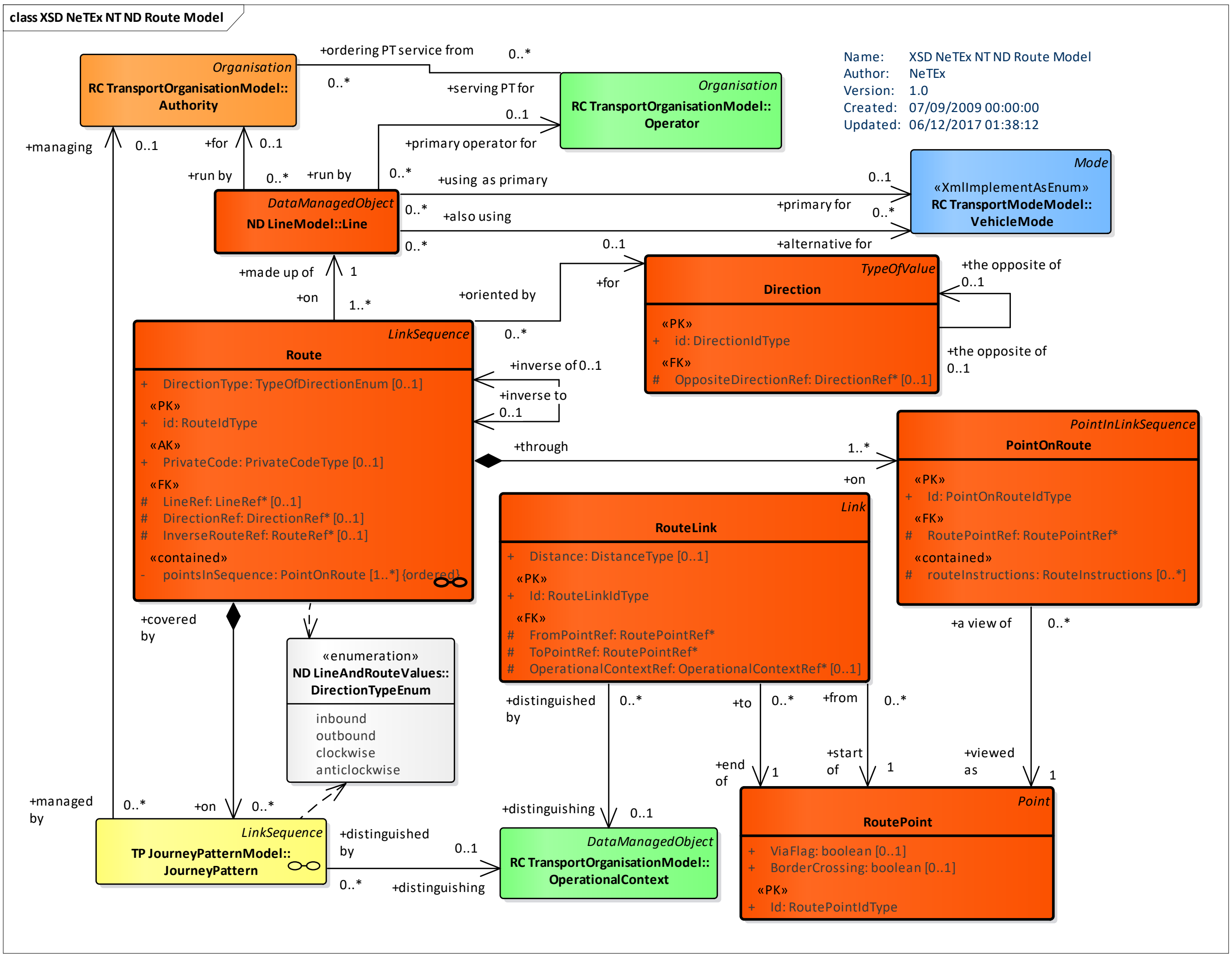


Figure 429 – Route – Physical Model (UML)

8.3.4.5 Route – Attributes and XSD

8.3.4.5.1 Direction – Model Element

A classification for the general orientation of ROUTEs.

Table 361 – *Direction* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	DIRECTION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*DirectionIdType*	1:1	Identifier of DIRECTION.
«AK»	***ExternalDirectionRef***	*ExternalObjectRef*	0:1	An alternative code that uniquely identifies the DIRECTION specifically for use in AVMS systems. For VDV compatibility.

«enum»	***DirectionType***	*DirectionTypeEnum*	0:1	A fixed value e.g. '*Outbound*', '*Inbound*', '*Clockwise*', associated with this direction. See allowed values below.
«FK»	***OppositeDirection Ref***	*DirectionRef*	0:1	Reference to a DIRECTION that runs counter to this DIRECTION.

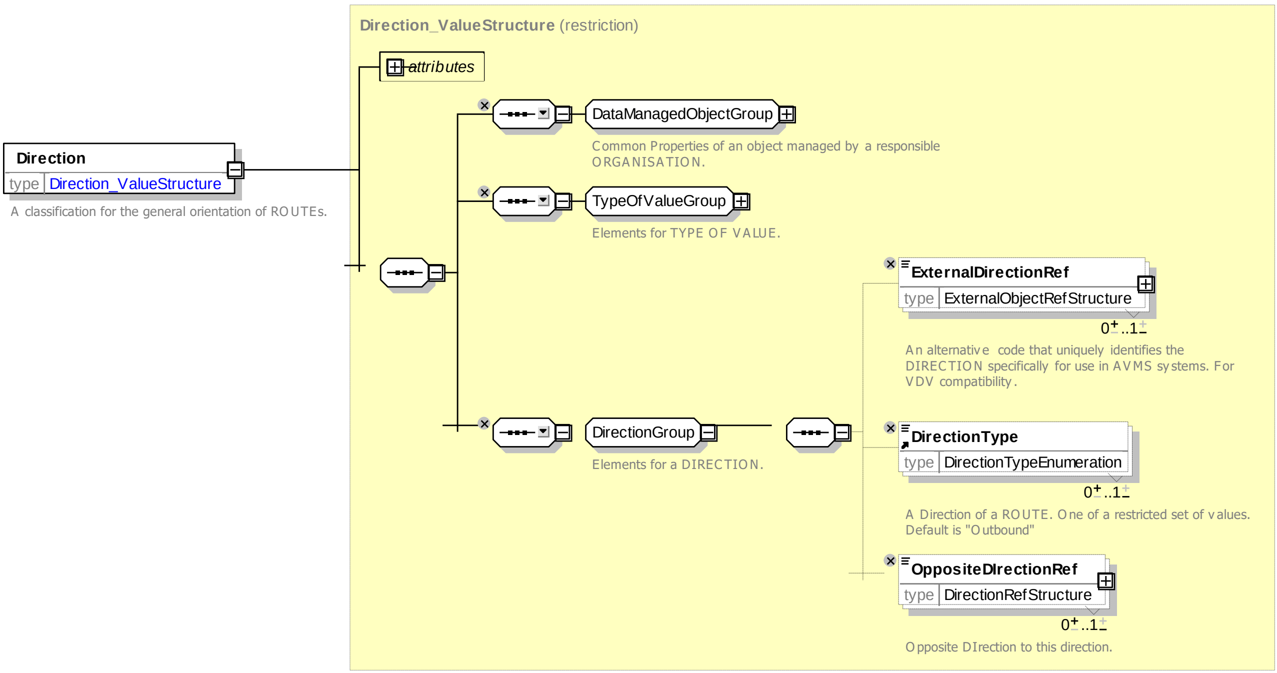


Figure 430 – *Direction* – XSD

8.3.4.5.2 RoutePoint – Model Element

A POINT used to define the shape of a ROUTE through the network.

Table 362 – *RoutePoint* – Element

Classifi cation	Name	Type	Cardina lity	Description
::>	::>	*Point*	::>	ROUTE POINT inherits from POINT.
«PK»	***id***	*RoutePointIdType*	1:1	Identifier of ROUTE POINT.
	ViaFlag	*xsd:boolean*	1:1	Whether ROUTE POINT is flagged as a via point.
	BorderCrossing	*xsd:boolean*	0:1	Whether points is a border crossing (defined for country border).

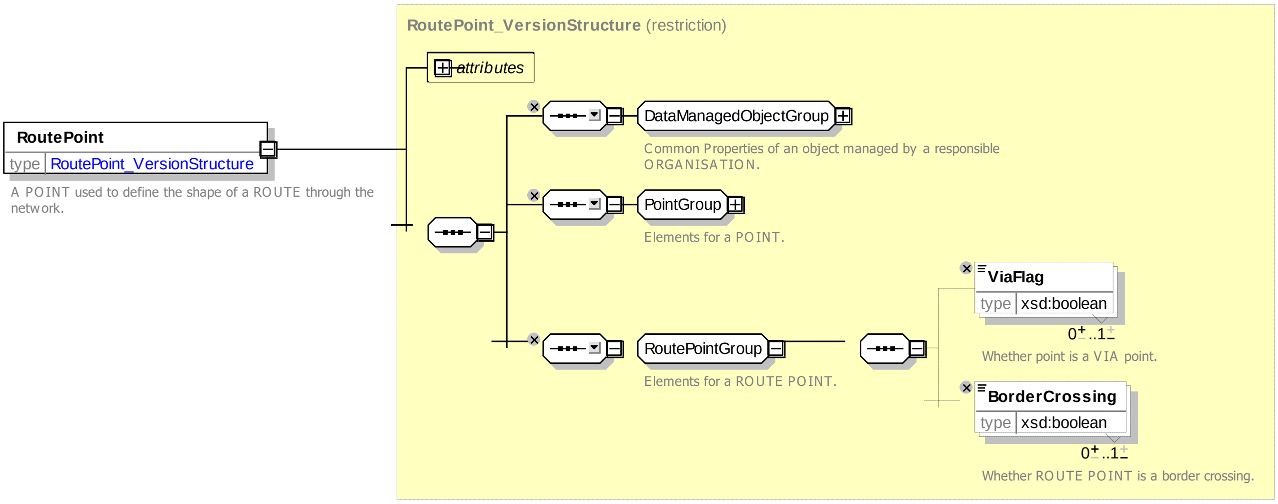


Figure 431 – *RoutePoint* – XSD

8.3.4.5.3 *RouteLink – Model Element*

An oriented link between two ROUTE POINTs allowing the definition of a unique path through the network.

Table 363 – *RouteLink* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Link*	::>	ROUTE LINK inherits from LINK.
«PK»	***id***	*RouteLinkIdType*	1:1	Identifier of ROUTE LINK.
	Distance	*DistanceType*	1:1	Distance of ROUTE LINK. Units are as specified for Frame. (Distance is inherited from LINK, but, by convention, mandatory in ROUTE LINK).
«FK»	***FromPointRef***	*RoutePointRef*	1:1	Reference to ROUTE POINT at which ROUTE LINK starts.
«FK»	***ToPointRef***	*RoutePointRef*	1:1	Reference to ROUTE POINT at which ROUTE LINK ends.
«FK»	***Operational-ContextRef***	*OperationalContextRef*	0:1	Reference to OPERATIONAL CONTEXT for ROUTE LINK.

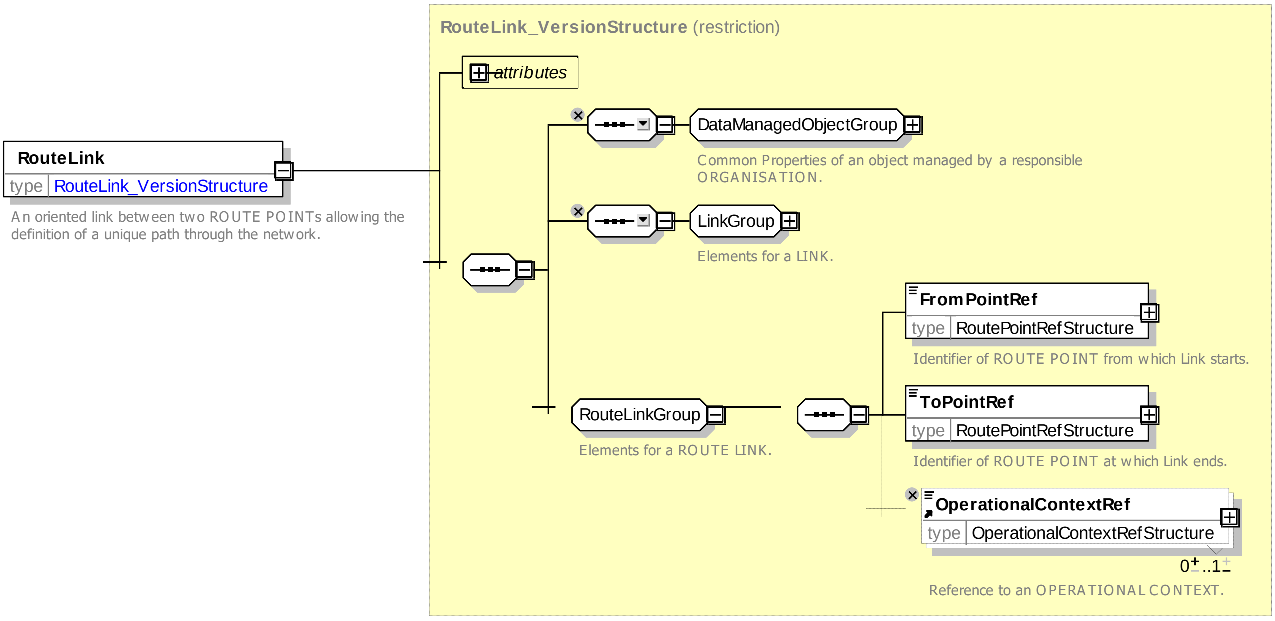


Figure 432 – *RouteLink – XSD*

8.3.4.5.4 Route – Model Element

An ordered list of located POINTs defining one single path through the road (or rail) network. A ROUTE may pass through the same POINT more than once.

Table 364 – *Route* – Element

Classification	Name	Type	Cardinality	Description
::>	***::>***	*LinkSequence*	::>	ROUTE inherits from LINK SEQUENCE.
«PK»	***id***	*RouteIdType*	1:1	Identifier of ROUTE.
«FK»	***LineRef***	*LineRef*	0:1	Reference to LINE to which ROUTE belongs.
«enum	***DirectionType***	*TypeOfDirectionEnum*	0:1	Type of Direction of ROUTE (outbound, inbound). See allowed values below.
«FK»	***DirectionRef***	*DirectionRef*	0:1	Reference to DIRECTION of ROUTE.
«cntd»	***pointsInSequence***	*PointOnRoute*	2:*	Points making up ROUTE.
«FK»	***InverseRouteRef***	*RouteRef*	0:1	Reference to the corresponding matching ROUTE in the opposite direction, if any.

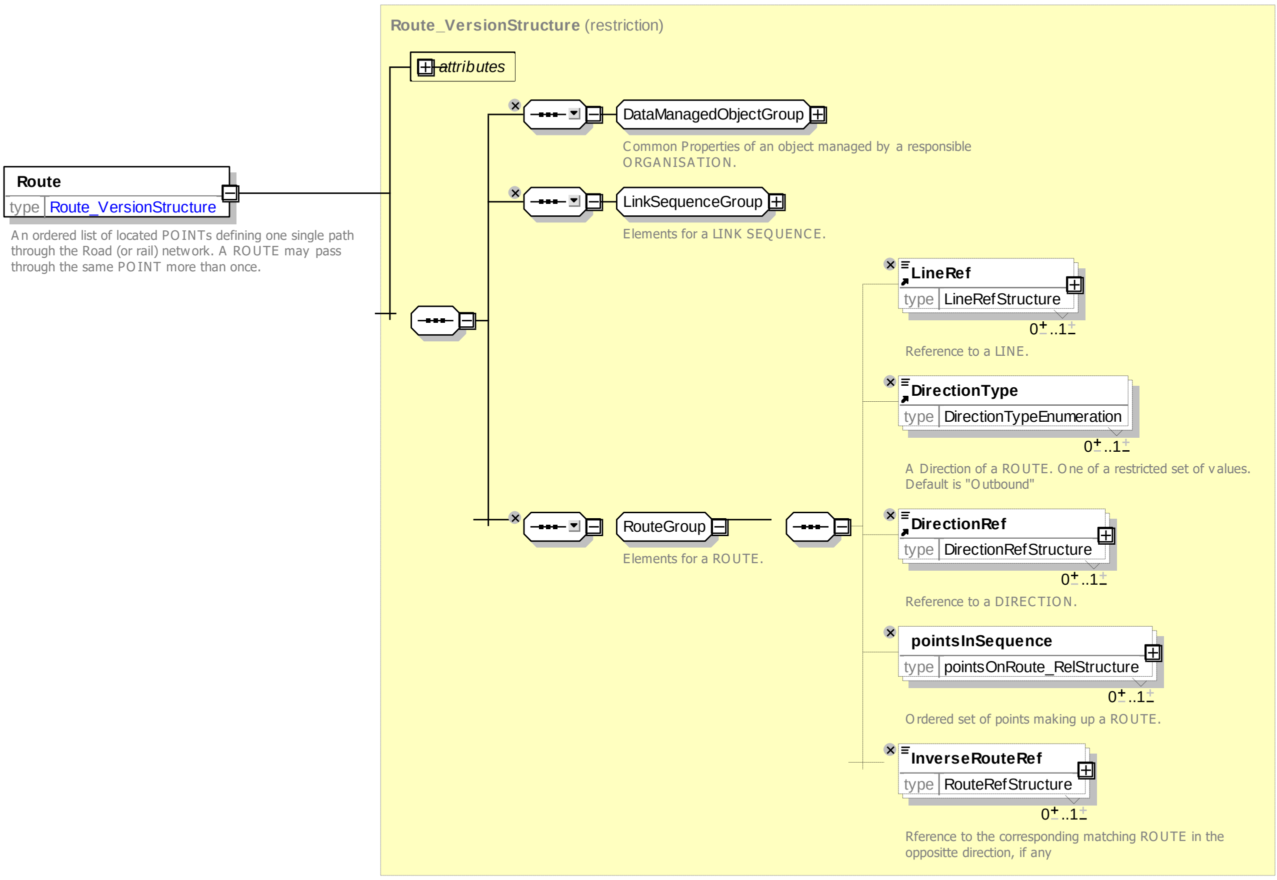


Figure 433 – *Route* – XSD

8.3.4.5.4.1 DirectionType – Allowed values

The following table shows the allowed values for ***DirectionType*** ***(****DirectionTypeEnumeration****).***

Table 365 – *DirectionType* – Allowed values

Value	Description
outbound	Outbound direction
inbound	Inbound direction
clockwise	Clockwise direction
anticlockwise	Anticlockwise direction

8.3.4.5.5 PointOnRoute – Model Element

A ROUTE POINT used to define a ROUTE with its order on that ROUTE.

Table 366 – *PointOnRoute* – Element

Classification	Name	Type	Cardinality	Description
::>	***::>***	*PointInLinkSequence*	::>	POINT ON ROUTE inherits from POINT IN LINK SEQUENCE.
«PK»	***id***	*PointOnRouteIdType*	1:1	Identifier of POINT ON ROUTE.

«FK»	***LinkSequenceRef***	*(LinkSequenceRef)*	0:1	Reference to a LINK SEQUENCE.
«cntd»	***projections***	*Projection*	0:1	Projection on the POINT OF ROUTE (on the infrastructure).
	Description	*MultilingualString*	0:1	Description of POINT ON ROUTE.
«FK»	***RoutePointRef***	*RoutePointRef*	1:1	Reference to ROUTE POINT that POINT ON ROUTE uses.
«FK»	***OnwardRouteLink Ref***	*RouteLinkRef*	0:1	Optional Reference to onward link to use - can be used to disambiguate where there are multiple links between the same point.
«cntd»	***routeInstructions***	*RouteInstruction*	0:*	Step instructions for point. +V1.1

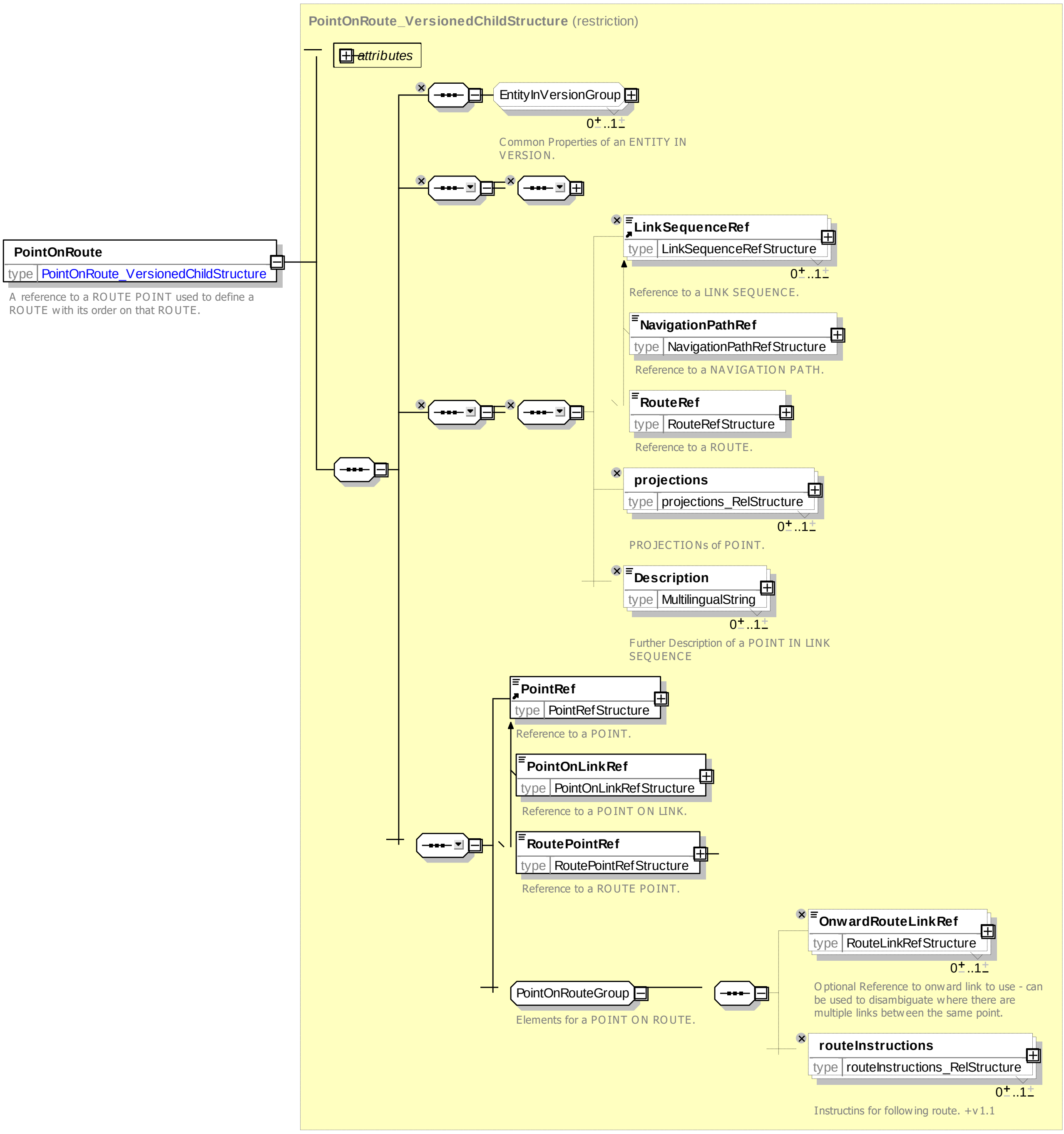


Figure 434 – *PointOnRoute* – XSD

8.3.4.5.6 Route Instruction –Model Element

Some operators require a step by step instruction for following the route that can be provided to drivers. This can be represented by an additional attachment to the POINT ON ROUTE, the PATH INSTRUCTION.

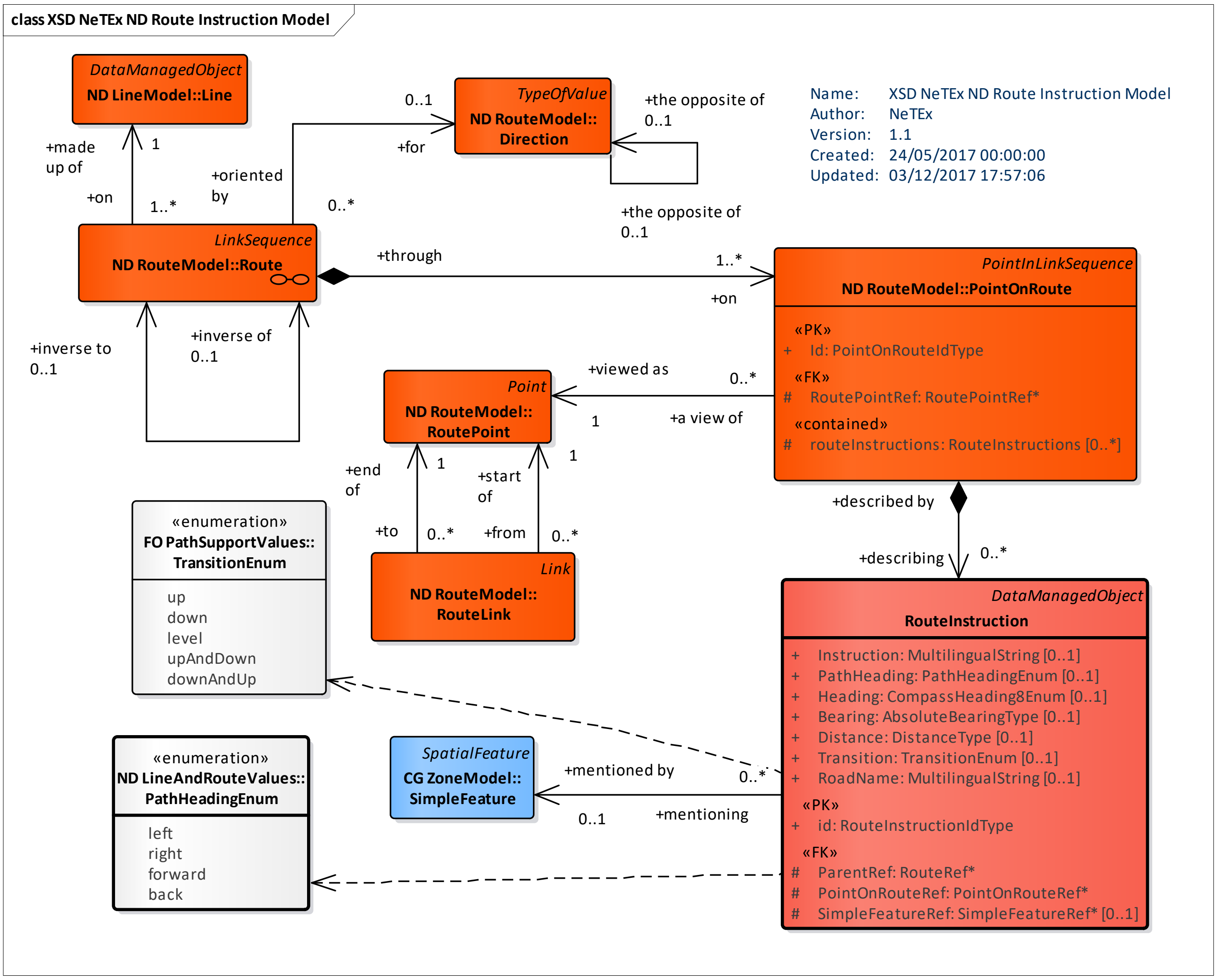


Figure 435 – Route Instruction – Physical Model (UML)

8.3.4.5.7 Route Instructions – Attributes and XSD

The physical model elements are as follows:

Route Instructions is a view providing Information for a human readable instruction as to how to follow a step in the route; data may be derived from underlying route links and a spatial PROJECTION of ROUTE LINK and ROUTE POINTs.

.

Table 367 – *RouteInstruction* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	ROUTE INSTRUCTION inherits from DATA MANAGED OBJECT. +V1.1
«PK»	***id***	*RouteInstructionIdType*	1:1	Identifier of ROUTE INSTRUCTION.
«FK»	***ParentRef***	*RouteRef*	1:1	Reference to ROUTE of POINT ON ROUTE.

«FK»	***PointOnRouteRef***	*PointOnRouteRef*	1:1	Reference to POINT ON ROUTE to which the instructions apply.
	Instruction	*MultilingualString*	0:1	Instruction for PATH LINK IN SEQUENCE.
«enum»	***PathHeading***	*PathHeadingEnum*	0:1	Heading instruction relative to point declared *'left'*, *'right'* onwards, etc. See allowed values below.
«enum»	***Heading***	*CompassHeading8Enum*	0:1	Heading instruction relative to point declared as compass bearing *'N', 'S',* etc. See allowed value above.
«FK»	***Bearing***	*AbsoluteBearingType*	0:1	Heading instruction in degrees.
	Distance	*DistanceType*	0:1	Distance of ROUTE LINK. Units are as specified for Frame.
«enum»	***Transition***	*TransitionEnum*	0:1	Transition for ROUTE LINK: '*up', 'down', 'level'*. See PATH LINK model for allowed values
	RoadName	*MultilingualString*	0:1	Name of road to follow.
«FK»	***SimpleFeatureRef***	*SimpleFeatureRef*	0:1	Reference to SIMPLE FEATURE associated with step.

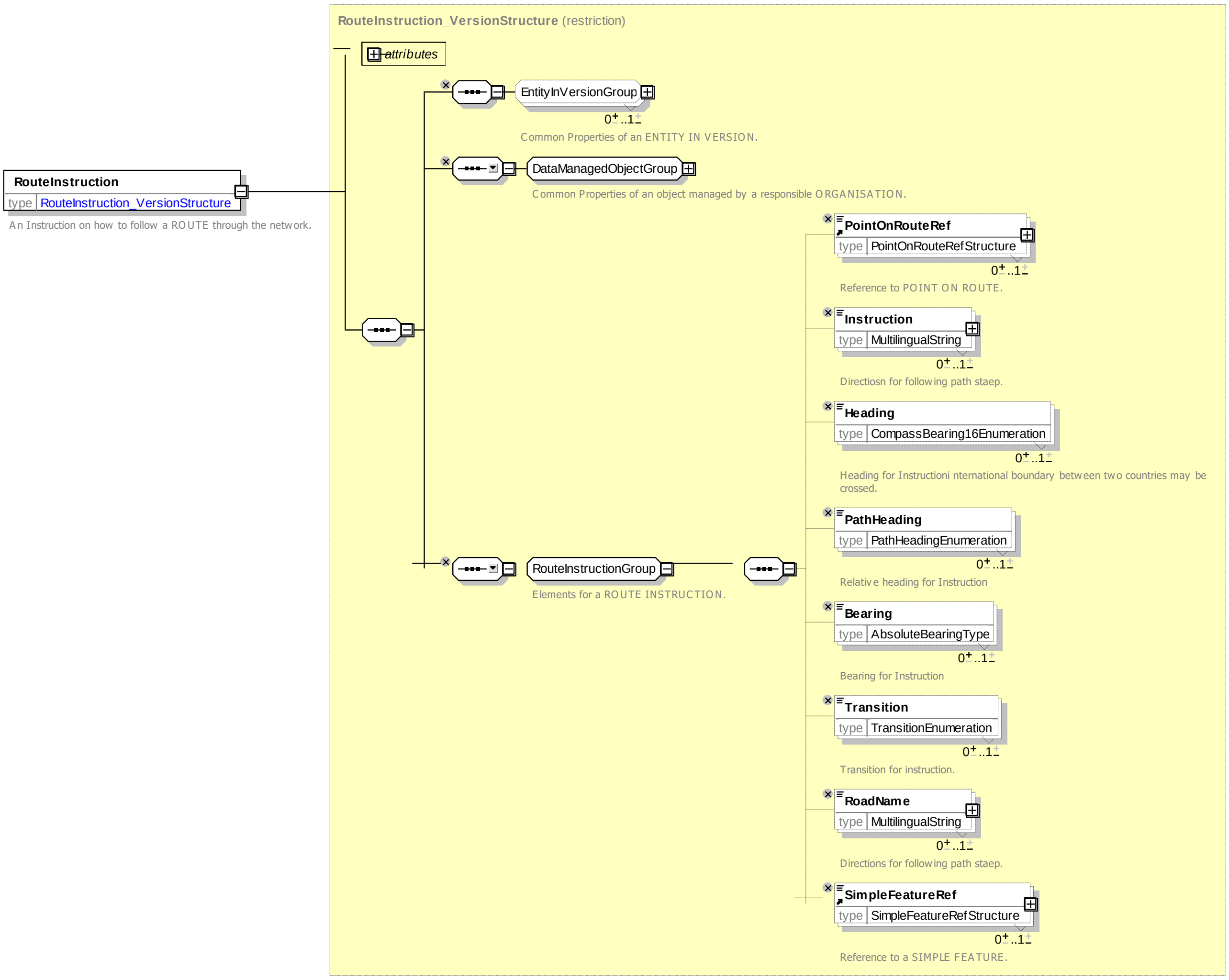


Figure 436 – *RouteInstruction* – XSD

8.3.4.6 LINE – Conceptual MODEL

Transmodel defines a LINE (Figure 437) as a grouping of ROUTEs that is generally known to the public by a similar name or number. These ROUTEs are usually very similar to each other from the topological point of view, being variants of a core route with some deviations on certain parts only. Often the vehicle journeys on these ROUTEs are scheduled jointly with tight synchronisation, in order to provide a regular service on this specific LINE. They are often grouped together for presentation of the timetable to the public.

Two ROUTEs using the same infrastructure path (or parallel tracks), but with opposite DIRECTIONs, will generally belong to the same LINE.

LINEs may be grouped into GROUPs OF LINE for particular purposes, such as fare harmonisation, day type assignment, or to group some kind of service categories (night buses, etc.). Grouping can also be used to define several kinds of PT networks and sub-networks: what is usually called 'public transport network' is in fact only a specific GROUP OF LINEs and a LINE may belong to several of them. For example in Île-deFrance, a LINE may belong not only to the STIF network (the network for all of the Île-de-France), but also to the *Nocitlien* network (night buses) and to the *PHEBUS* network (the bus network for the town of Versailles).

Each GROUP OF LINEs must be defined for only one purpose, which is expressed by a PURPOSE OF GROUPING. A LINE may be in different groups for different purposes, and may also belong to more than one GROUP OF LINEs for one single purpose. It is the responsibility of users to assure consistent groupings, suitable for the specific purpose in question.

A NETWORK is a specialisation of a GROUP OF LINEs that additionally associates OPERATORs and TARIFF ZONEs as a named transport netwrok.

Information about services running on a line will indicate the destinations to which the service run. A set of standardised names for these destination can be given as DESTINATION DISPLAYs; different values can be used at different poinst on the route and there can be different DESTINATION DISPLAY VARIANTs for use in different media and different display format.

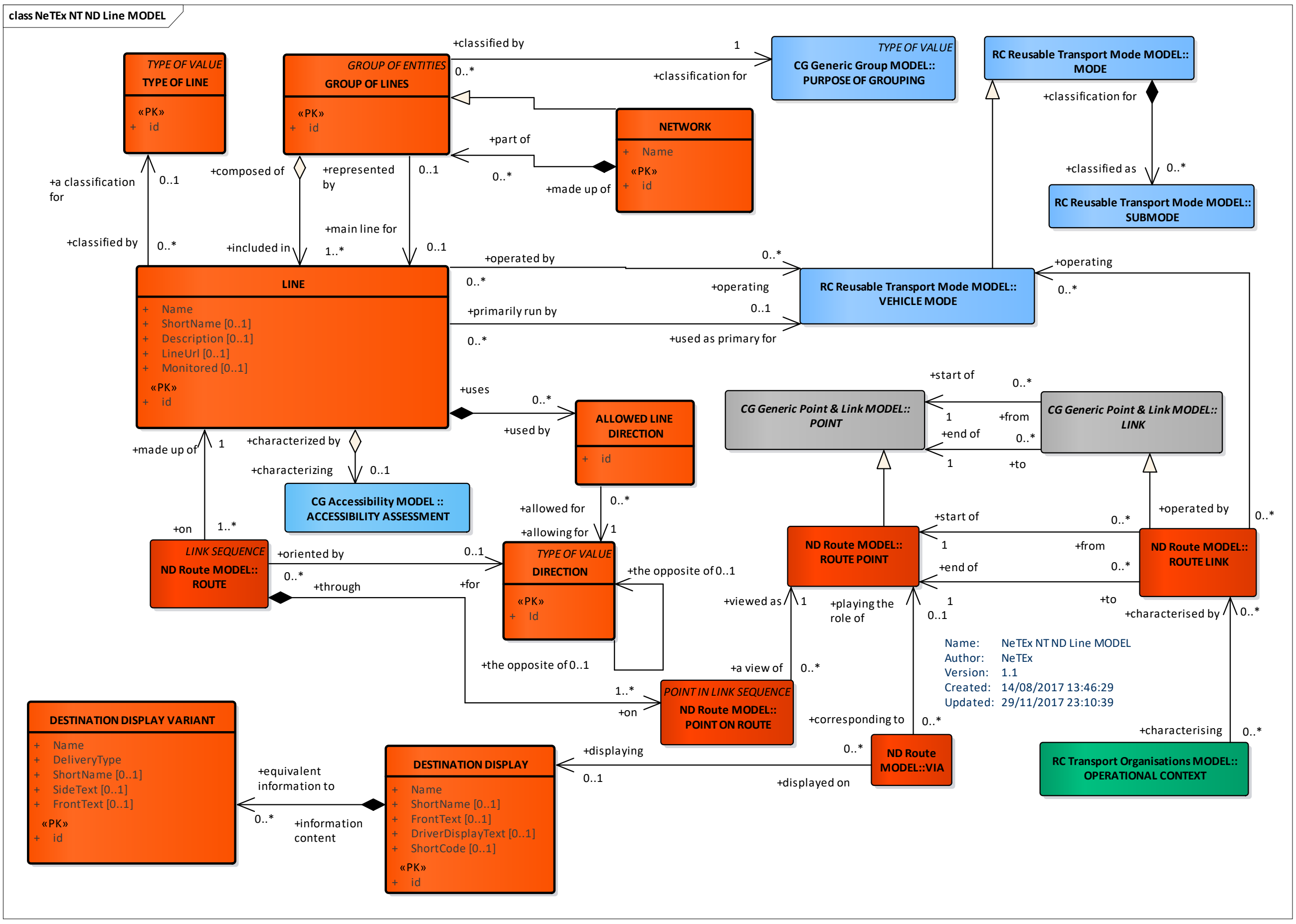


Figure 437 – LINE – Conceptual MODEL (UML)

8.3.4.7 Line – Physical Model

Figure 438 shows the Physical model for a LINE.

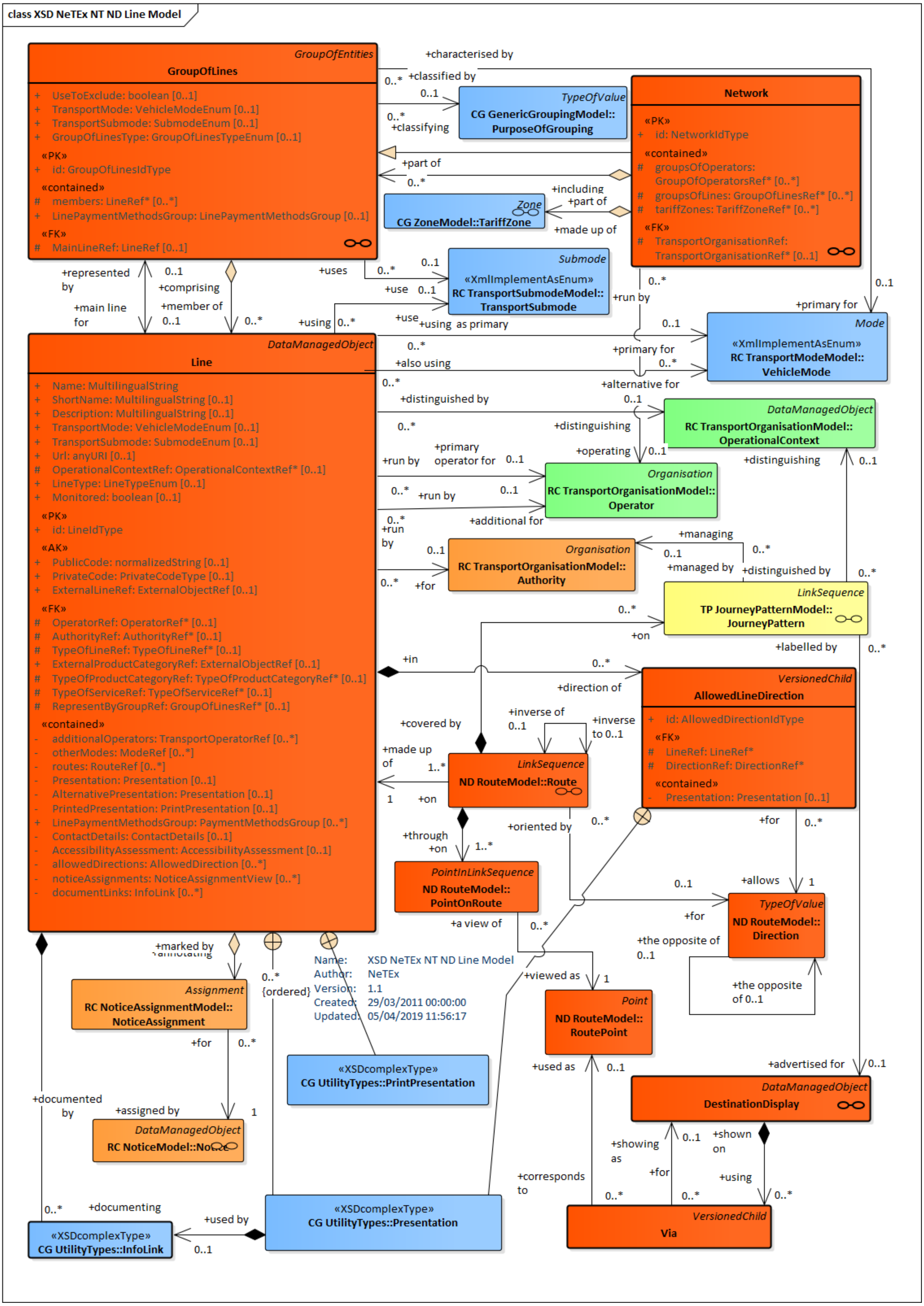

class XSD NeTEx NT ND Line Model
GroupOfEntities
GroupOfLines
+ UseToExclude: boolean [0..1]
+ TransportMode: VehicleModeEnum [0..1]
+ TransportSubmode: SubmodeEnum [0..1]
+ GroupOfLinesType: GroupOfLinesTypeEnum [0..1]
«PK»
+ id: GroupOfLinesIdType
«contained»
members: LineRef* [0..*]
+ LinePaymentMethodsGroup: LinePaymentMethodsGroup [0..1]
«FK»
MainLineRef: LineRef [0..1]
+characterised by
0..* +classified by
0..1
TypeOfValue
CG GenericGroupingModel::
PurposeOfGrouping
0..*
+classifying
Network
«PK»
+ id: NetworkIdType
«contained»
groupsOfOperators:
GroupOfOperatorsRef* [0..*]
groupsOfLines: GroupOfLinesRef* [0..*]
tariffZones: TariffZoneRef* [0..*]
«FK»
TransportOrganisationRef:
TransportOrganisationRef* [0..1]
+part of
0..*
+including
+part of
Zone
CG ZoneModel::TariffZone
+made up of
+represented by
0..1
+comprising
+uses
0..*
0..1
Submode
«XmlImplementsAsEnum»
RC TransportSubmodeModel::
TransportSubmode
+use 0..1
0..*
+run by
0..1
+main line for
+member of
0..1
0..*
+using
0..*
+use
+using as primary
+primary for
0..1
Mode
«XmlImplementsAsEnum»
RC TransportModeModel::
VehicleMode
DataManagedObject
Line
+ Name: MultilingualString
+ ShortName: MultilingualString [0..1]
+ Description: MultilingualString [0..1]
+ TransportMode: VehicleModeEnum [0..1]
+ TransportSubmode: SubmodeEnum [0..1]
+ Url: anyURI [0..1]
# OperationalContextRef: OperationalContextRef* [0..1]
+ LineType: LineTypeEnum [0..1]
+ Monitored: boolean [0..1]
«PK»
+ id: LineIdType
«AK»
+ PublicCode: normalizedString [0..1]
+ PrivateCode: PrivateCodeType [0..1]
+ ExternalLineRef: ExternalObjectRef [0..1]
«FK»
OperatorRef: OperatorRef* [0..1]
AuthorityRef: AuthorityRef* [0..1]
TypeOfLineRef: TypeOfLineRef* [0..1]
+ ExternalProductCategoryRef: ExternalObjectRef [0..1]
# TypeOfProductCategoryRef: TypeOfProductCategoryRef* [0..1]
# TypeOfServiceRef: TypeOfServiceRef* [0..1]
# RepresentByGroupRef: GroupOfLinesRef* [0..1]
«contained»
- additionalOperators: TransportOperatorRef [0..*]
- otherModes: ModeRef [0..*]
- routes: RouteRef [0..*]
- Presentation: Presentation [0..1]
- AlternativePresentation: Presentation [0..1]
- PrintedPresentation: PrintPresentation [0..1]
+ LinePaymentMethodsGroup: PaymentMethodsGroup [0..*]
- ContactDetails: ContactDetails [0..1]
- AccessibilityAssessment: AccessibilityAssessment [0..1]
- allowedDirections: AllowedDirection [0..*]
- noticeAssignments: NoticeAssignmentView [0..*]
- documentLinks: InfoLink [0..*]
0..*
+also using
+primary for
0..*
0..*
+alternative for
+distinguished by
0..1
DataManagedObject
RC TransportOrganisationModel::
OperationalContext
0..*
+distinguishing
+primary operator for
+run by
0..1
+operating
0..1
+distinguishing
0..1
Organisation
RC TransportOrganisationModel::
Operator
0..*
+run by
0..1
0..*
+run by
+additional for
+managing
0..1
Organisation
RC TransportOrganisationModel::
Authority
0..1
+managed by
0..*
+distinguished by
0..*
0..*
+for
LinkSequence
TP JourneyPatternModel::
JourneyPattern
0..*
+on
+labelled by
0..*
+in
0..*
VersionedChild
AllowedLineDirection
+ id: AllowedDirectionIdType
«FK»
LineRef: LineRef*
DirectionRef: DirectionRef*
«contained»
- Presentation: Presentation [0..1]
+direction of
+covered by
+inverse of
0..1
+inverse to 0..1
+made up of
1..*
LinkSequence
ND RouteModel::Route
1
+on
+oriented by
0..*
+for
0..*
+through
+on
1..*
PointInLinkSequence
ND RouteModel::
PointOnRoute
0..1
+allows
1
TypeOfValue
ND RouteModel::
Direction
+for
+a view of
0..*
+the opposite of
0..1
+the opposite of 0..1
+marked by
+annotating
0..*
{ordered}
Name: XSD NeTEx NT ND Line Model
Author: NeTEx
Version: 1.1
Created: 29/03/2011 00:00:00
Updated: 05/04/2019 11:56:17
+viewed as
1
Point
ND RouteModel::
RoutePoint
Assignment
RC NoticeAssignmentModel::
NoticeAssignment
+for
0..*
«XSDcomplexType»
CG UtilityTypes::PrintPresentation
+used as
0..1
+advertised for
0..1
DataManagedObject
DestinationDisplay
+documented by
+assigned by
1
DataManagedObject
RC NoticeModel::Notice
+showing as
0..1
+shown on
+for
+using
+corresponds to
0..*
0..*
0..*
0..*
+documenting
+used by
«XSDcomplexType»
CG UtilityTypes::InfoLink
0..1
«XSDcomplexType»
CG UtilityTypes::Presentation
VersionedChild
Via

Figure 438 – Line – Physical Model (UML)

8.3.4.7.1 Line Details – Physical Model

Figure 439 shows futher details of the Physical model for a LINE.

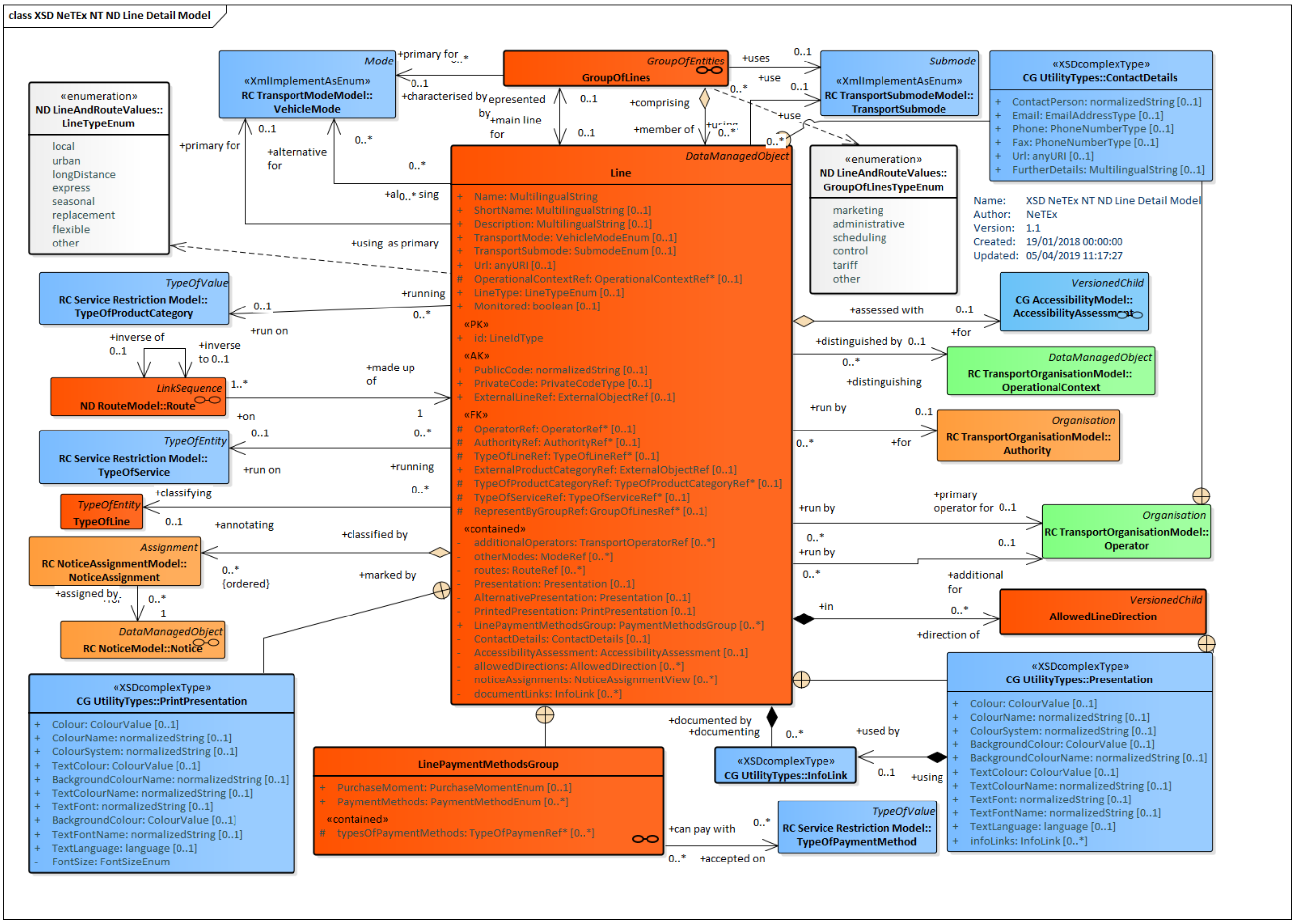


Figure 439 – Line Details – Physical Model (UML)

8.3.4.7.2 Destination Display– Physical Model

Figure 439 shows details of the Physical model for a DESTINATION DISPLAY.

There may be different variants of text to use in different cintext (SECONDARY NAME) or on different media (DESTINATION DISPLAY VARIANT)

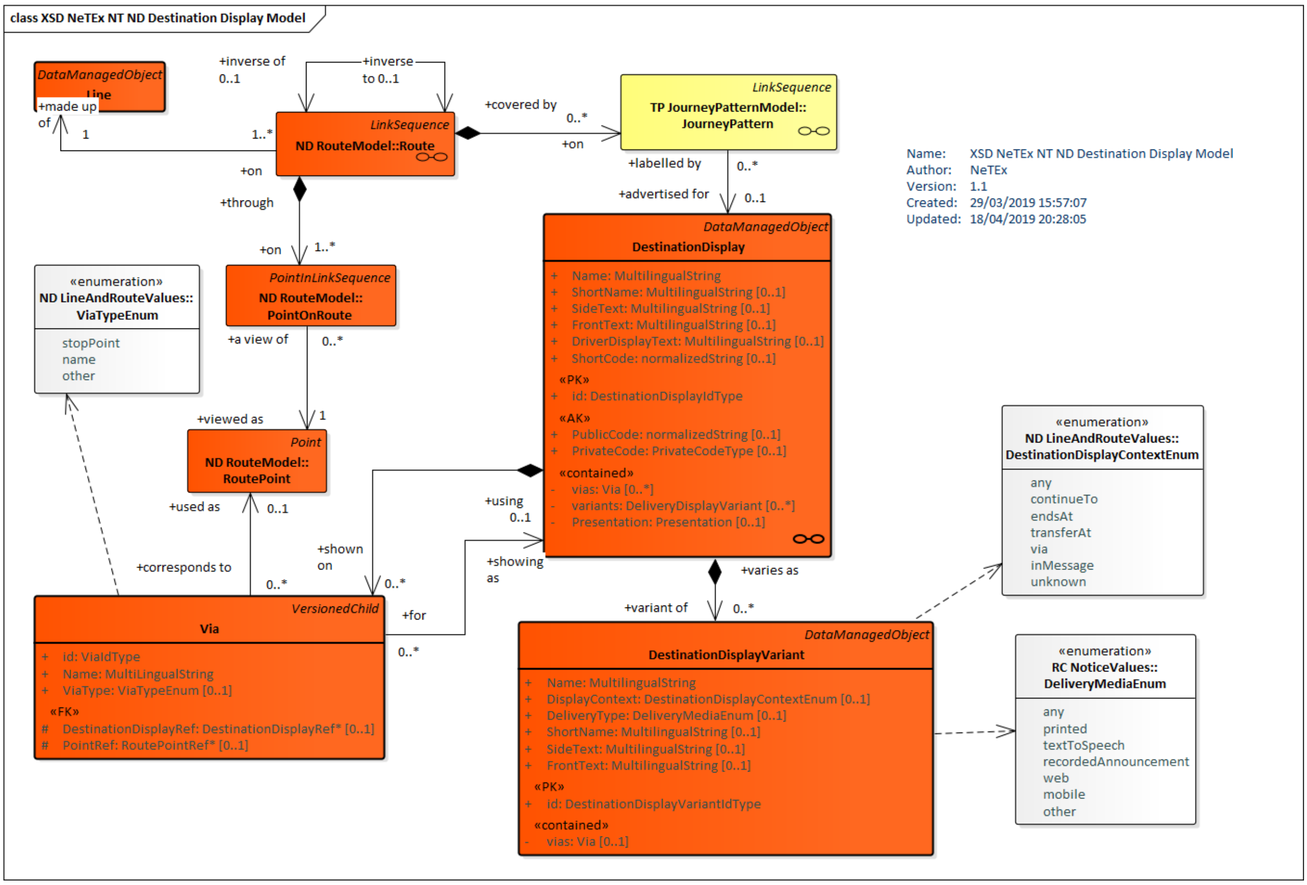


Figure 440 – Destination Display – Physical Model (UML)

8.3.4.7.3 Line Classifications – Physical Model

Figure 441 shows stadardised classification values used in the Physical model for a LINE.

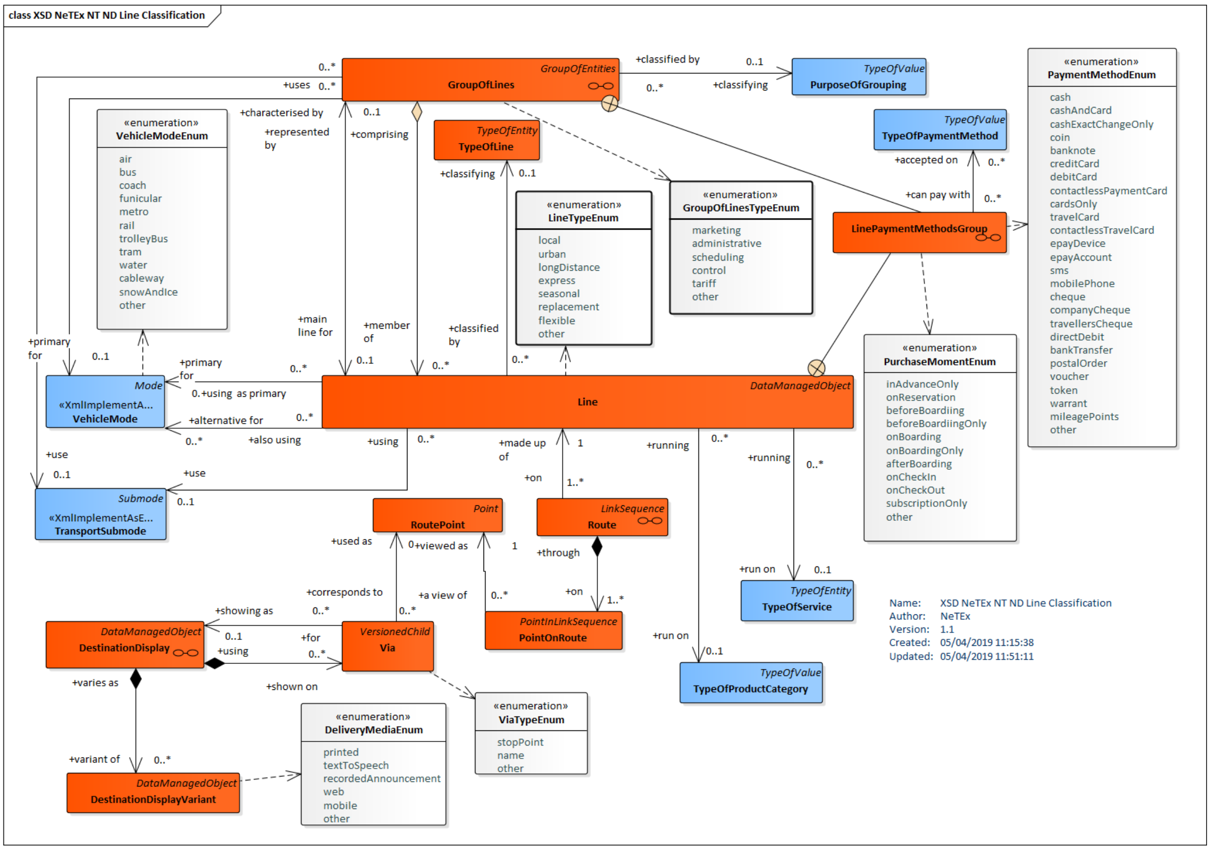


Figure 441 – Line Classifications – Physical Model (UML)

8.3.4.7.4 Network – Physical Model

Figure 442 shows the Physical model for a NETWORK and GROUP OF LINEs.

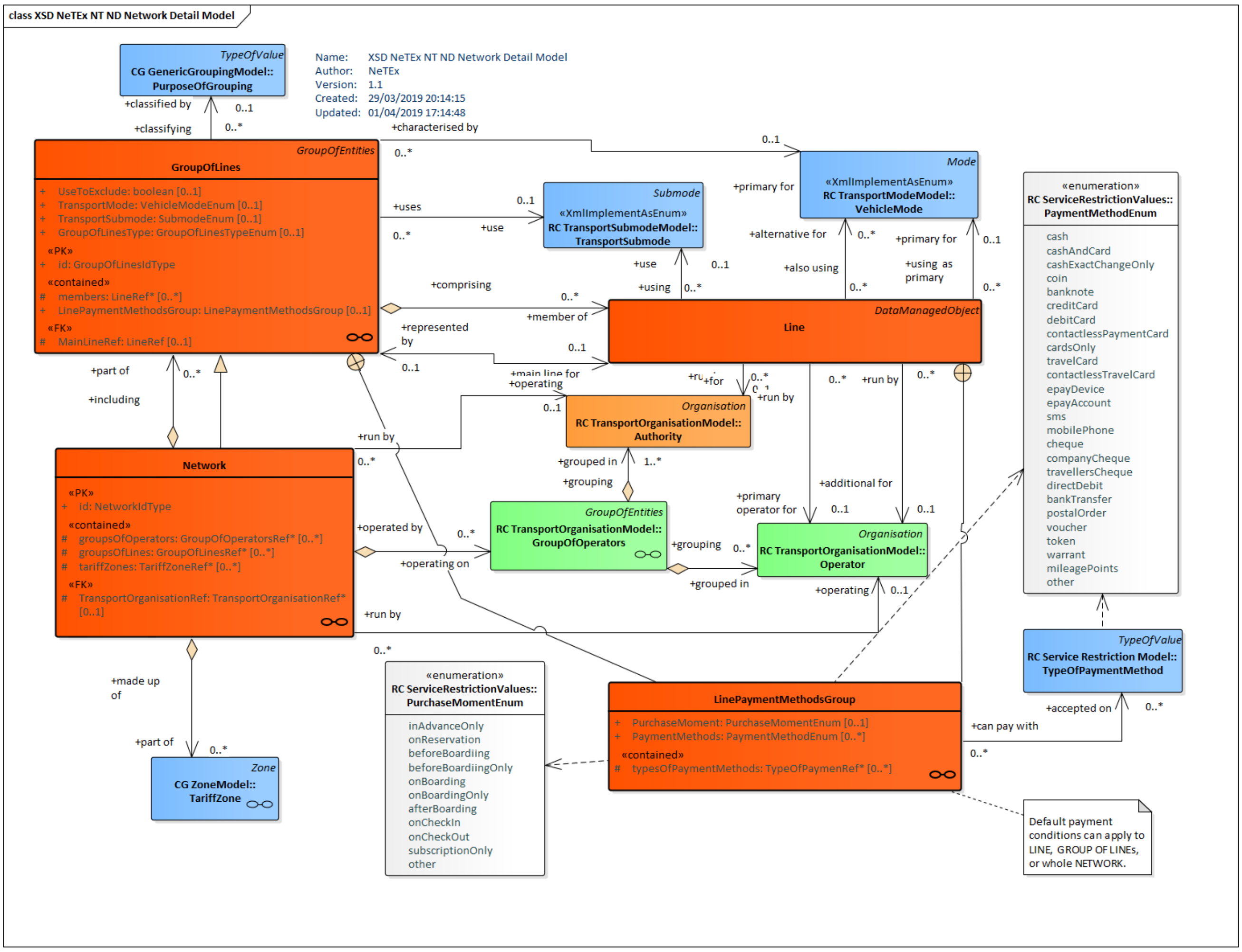


Figure 442 – Network Details – Physical Model (UML)

8.3.4.8 Line – Attributes and XSD

8.3.4.8.1 Line – Model Element

A group of ROUTEs which is generally known to the public by a similar name or number.

Table 368 – *Line* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	LINE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*LineIdType*	1:1	Identifier of LINE.
XGRP	***LineDescriptionGroup***	***xmlGroup***	0:1	Elements describing LINE. See below.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Public identifier of LINE.

«AK»	***PrivateCode***	*PrivateCode*	0:1	Alternative identifier of LINE.
«AK»	***ExternalLineRef***	*ExternalObjectRef*	0:1	An alternative code that uniquely identifies the LINE specifically for use in AVMS systems. For VDV compatibility.
XGRP	***LineProperties-Group***	***xmlGroup***	0:1	Elements specifying general properties of LINE. See below.

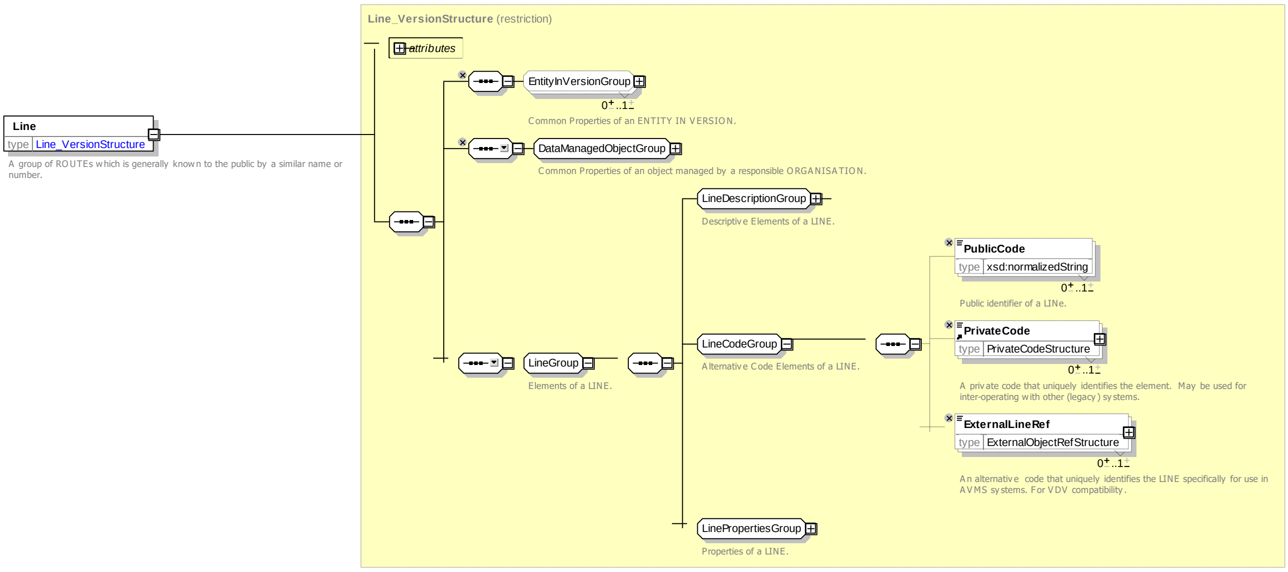


Figure 443 – Line – XSD

8.3.4.8.1.1 LineDescriptionGroup – XML Group

Table 369 – *Line* – Group

Classification	Name	Type	Cardinality	Description
	Name	*MultilingualString*	1:1	Name of LINE.
	ShortName	*MultilingualString*	0:1	Short Name of LINE.
	Description	*MultilingualString*	0:1	Description of LINE.
«enum»	***TransportMode***	*AllVehicleModesOf-TransportEnum*	0:1	Identifier of Primary TRANSPORT MODE of LINE. See TRANSPORT MODE model for allowed values.
«enum»	***Transport-Submode***	*TransportSubmodeEnum*	0:1	Submode of the TRANSPORT MODE. See TRANSPORT MODE model for allowed values.
	Url	*any*	0:1	A URL associated with the LINE.

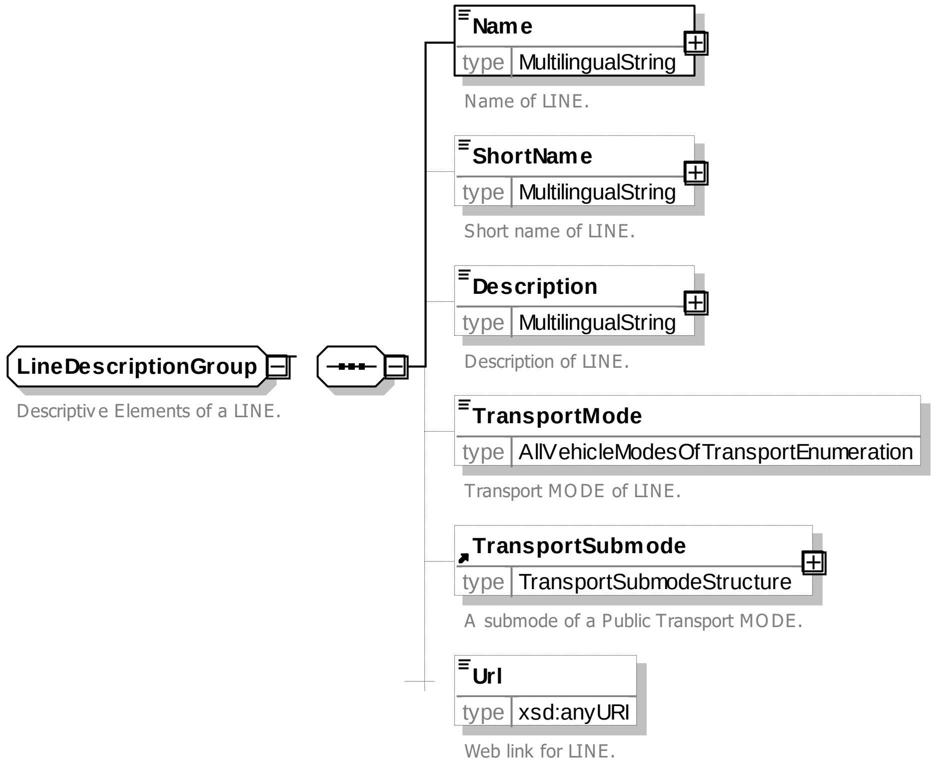


Figure 444 – LineDescriptionGroup – XSD

8.3.4.8.1.2 LinePropertiesGroup – XML Group

Table 370 – *LinePropertiesGroup* – Group

Classification	Name		Type	Cardinality	Description
			CHOICE		Primary TRANSPORT OPERATOR
«FK»	a	***OperatorRe***	*OperatorRef*	0:1	Reference to OPERATOR of LINE.
«FK»		***AuthorityRef***	*AuthorityRef*	0:1	Reference to AUTHORITY of LINE.
«cntd»	***additional-Operators***		*(TransportOrganisationRef) AuthorityRef* | *OperatorRef*	0:*	References to additional OPERATOR of LINE.
«cntd»	***otherModes***		*ModeRef*	0:*	Additional transport MODEs for LINE.
«FK»	***Operational-ContextRef***		*OperationalContextRef*	0:1	Reference to OPERATIONAL CONTEXT of LINE.
«enum»	***LineType***		*LineTypeEnum*	0:1	Classification of LINE. See allowed values below.
«FK»	***TypeOfLineRef***		*TypeOfLineRef*	0:1	Reference to a TYPE of LINE. +v1.1
«FK»	***ExternalProduct-CategoryRef***		*ExternalObjectRef*	0:1	An external product classification for all journeys of the LINE for use in AVMS systems. For VDV compatibility. +v1.1

«FK»	***TypeOfProduct-CategoryRef***	*TypeOfProductCategoryRef*	0:1	Reference to a TYPE of PRODUCT CATEGORY. +v1.1
«FK»	***TypeOfServiceRef***	*TypeOfServiceRef*	0:1	Reference to a TYPE of SERVICE. +v1.1
	Monitored	*xsd:boolean*	0:1	Whether real-time data is normally available for LINE.
«cntd»	***routes***	*RouteRef*	0:*	Routes that follow the LINE.
«FK»	***RepresentBy-GroupRef***	*GroupOfLinesRef*	0:1	Identifier of GROUP OF LINEs that should be used to represent this LINE.
«cntd»	***Presentation***	*Presentation*	0:1	Preferred presentation attributes to use when rendering LINE on maps, etc.
«cntd»	***Alternative-Presentation***	*Presentation*	0:1	Alternative presentation attributes to use when rendering LINE on maps, etc.
«cntd»	***Printed-Presentation***	*PrintedPresentation*	0:1	Preferred presentation attributes to use when rendering LINE on maps, etc. for printing. +v1.1
«enum»	***PaymentMethods***	*PaymentMethodEnum*	0:*	PAYMENT METHODs accepted for this LINE. See Service Restriction Model for allowed values. +v1.1
«cntd»	***typesOfPayment-Method***	*TypeOfPaymentMethod*	0:*	TYPEs OF PAYMENT METHOD accepted for this LINE. +v1.1.
«enum»	***Purchase-Moments***	*PurchaseMomentEnum*	0:*	LINE. See Service Restriction Model for allowed values. +v1.1
«cntd»	***ContactDetails***	*ContactDetails*	0:1	Contact details for OPERATOR of the LINE.
«cntd»	***Accessibility-Assessment***	*AccessibilityAssessment*	0:1	The overall accessibility characteristics of the LINE. See accessibility model.
«cntd»	***allowedDirections***	*AllowedDirection*	0:*	ALLOWED DIRECTIONs allowed for this LINE.
«cntd»	***notice-Assignments***	*noticeAssignment*	0:*	NOTICES assigned to the LINE.
«cntd»	***documentLinks***	*InfoLinks*	0:*	Timetable documents associated with the LINE, e.g pdf files +v1.1

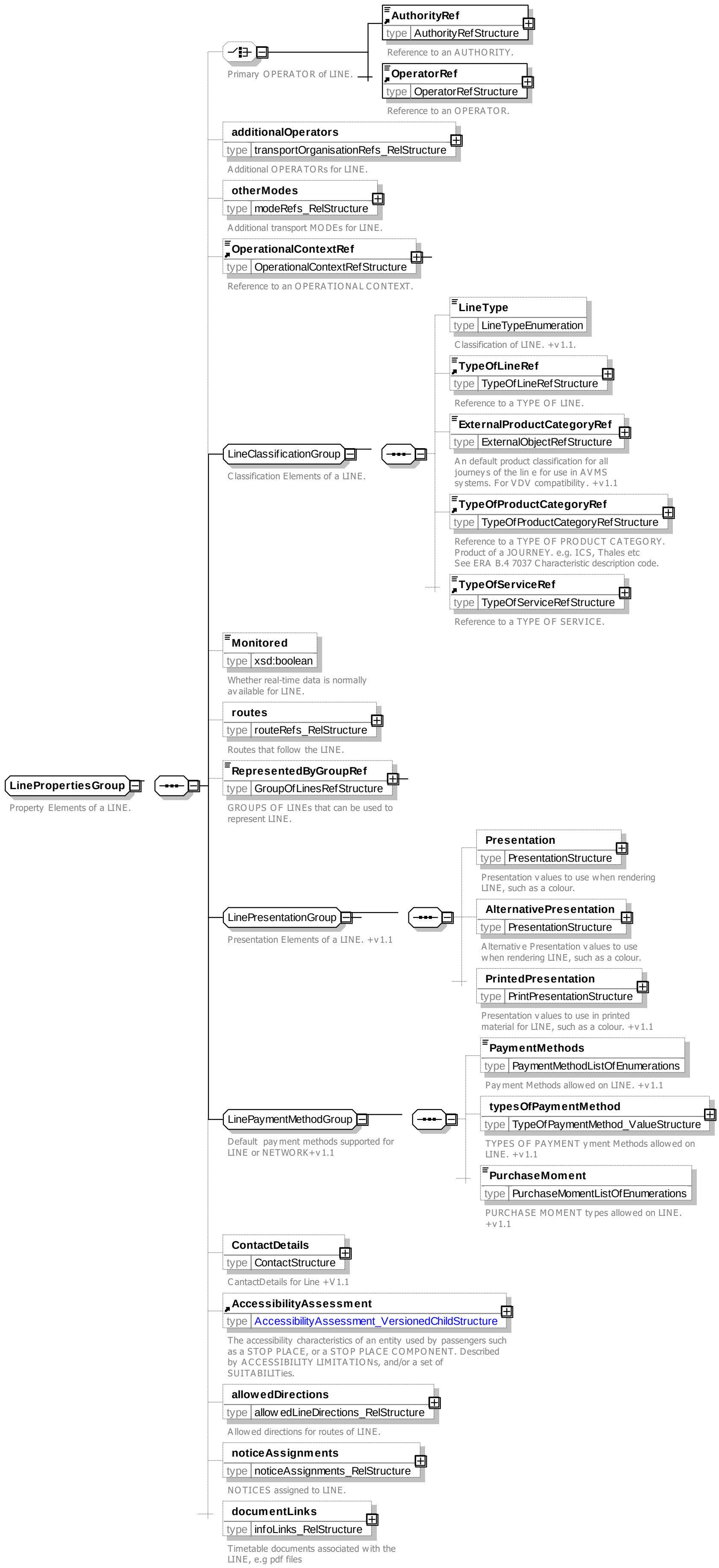

LinePropertiesGroup
Property Elements of a LINE.
Primary OPERATOR of LINE.
AuthorityRef
type AuthorityRefStructure
Reference to an AUTHORITY.
OperatorRef
type OperatorRefStructure
Reference to an OPERATOR.
additionalOperators
type transportOrganisationRefs_RelStructure
Additional OPERATORs for LINE.
otherModes
type modeRefs_RelStructure
Additional transport MODEs for LINE.
OperationalContextRef
type OperationalContextRefStructure
Reference to an OPERATIONAL CONTEXT.
LineClassificationGroup
Classification Elements of a LINE.
LineType
type LineTypeEnumeration
Classification of LINE. +v1.1.
TypeOfLineRef
type TypeOfLineRefStructure
Reference to a TYPE OF LINE.
ExternalProductCategoryRef
type ExternalObjectRefStructure
An default product classification for all journeys of the lin e for use in AVMS systems. For VDV compatibility. +v1.1
TypeOfProductCategoryRef
type TypeOfProductCategoryRefStructure
Reference to a TYPE OF PRODUCT CATEGORY. Product of a JOURNEY. e.g. ICS, Thales etc See ERA B.4 7037 Characteristic description code.
TypeOfServiceRef
type TypeOfServiceRefStructure
Reference to a TYPE OF SERVICE.
Monitored
type xsd:boolean
Whether real-time data is normally available for LINE.
routes
type routeRefs_RelStructure
Routes that follow the LINE.
RepresentedByGroupRef
type GroupOfLinesRefStructure
GROUPS OF LINEs that can be used to represent LINE.
LinePresentationGroup
Presentation Elements of a LINE. +v1.1
Presentation
type PresentationStructure
Presentation values to use when rendering LINE, such as a colour.
AlternativePresentation
type PresentationStructure
Alternative Presentation values to use when rendering LINE, such as a colour.
PrintedPresentation
type PrintPresentationStructure
Presentation values to use in printed material for LINE, such as a colour. +v1.1
LinePaymentMethodGroup
Default payment methods supported for LINE or NETWORK+v1.1
PaymentMethods
type PaymentMethodListOfEnumerations
Payment Methods allowed on LINE. +v1.1
typesOfPaymentMethod
type TypeOfPaymentMethod_ValueStructure
TYPES OF PAYMENT yment Methods allowed on LINE. +v1.1
PurchaseMoment
type PurchaseMomentListOfEnumerations
PURCHASE MOMENT types allowed on LINE. +v1.1
ContactDetails
type ContactStructure
CantactDetails for Line +V1.1
AccessibilityAssessment
type AccessibilityAssessment_VersionedChildStructure
The accessibility characteristics of an entity used by passengers such as a STOP PLACE, or a STOP PLACE COMPONENT. Described by ACCESSIBILITY LIMITATIONs, and/or a set of SUITABILITIes.
allowedDirections
type allowedLineDirections_RelStructure
Allowed directions for routes of LINE.
noticeAssignments
type noticeAssignments_RelStructure
NOTICES assigned to LINE.
documentLinks
type infoLinks_RelStructure
Timetable documents associated with the LINE, e.g pdf files

Figure 445 – LinePropertiesGroup – XSD

LineType – Allowed values

The following table shows the allowed values for *LineType* (*fLineTypeEnum*eration***).***

Table 371 –*fLineType* – Allowed Values

Value	Description
local	Local
urban	Urban
longDistance	Long distance
express	Express
seasonal	Seasonal
replacement	Temporary replacement
flexible	Flexible
other	Other

8.3.4.8.2 DestinationDisplay – Model Element

An advertised destination of a specific LINE or JOURNEY PATTERN, usually displayed on a headsign or at other on-board locations.

Table 372 – *DestinationDisplay* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	DESTINATION DISPLAY inherits from DATA MANAGED OBJECT.
«PK»	***id***	*DestinationDisplay-IdType*	1:1	Identifier of DESTINATION DISPLAY.
	Name	*MultilingualString*	0:1	Name of DESTINATION DISPLAY.
	ShortName	*MultilingualString*	0:1	Short Name of DESTINATION DISPLAY.
	SideText	*MultilingualString*	0:1	Text to display on side of vehicle associated with DESTINATION DISPLAY.
	FrontText	*MultilingualString*	0:1	Front of vehicle text associated with DESTINATION DISPLAY.
	DriverDisplay-Text	*MultilingualString*	0:1	Text to display to DRIVER.
«AK»	***ShortCode***	*xsd:normalizedString*	0:1	PUBLIC CODE associated with DESTINATION DISPLAY.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Private CODE associated with DESTINATION DISPLAY.
«AK»	***PrivateCode***	*xsd:normalizedString*	0:1	Additional short CODE associated with DESTINATION DISPLAY.
«cntd»	***Presentation***	Presentation	0:1	Preferred presentation attributes to use when rendering destination name on maps, etc. +v1.1
«cntd»	***vias***	*Via*	0:*	Text to show for VIA display.

«cntd»	***variants***	*DestinationDisplay-Variant*	0:*	Variant DESTINATION DISPLAY texts for different media and in different contexts.

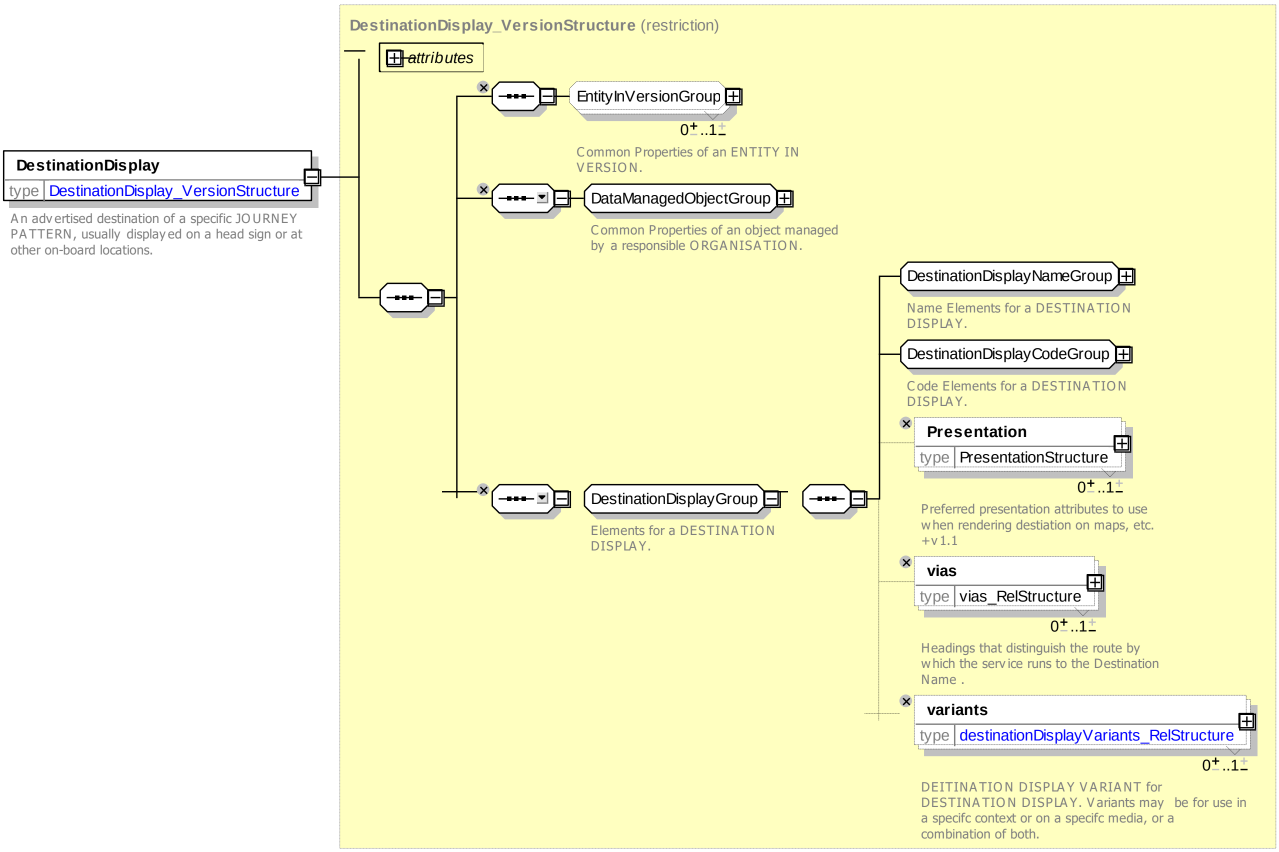


Figure 446 – *DestinationDisplay* – XSD

8.3.4.8.3 DestinationDisplayVariant – Model Element

A variant text for DESTINATION DISPLAY.

Table 373 – *DestinationDisplayVariant* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	DESTINATION DISPLAY VARIANT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*DestinationDisplay-VariantIdType*	1:1	Identifier of DESTINATION DISPLAY VARIANT.
	Name	*MultilingualString*	1:1	Name of DESTINATION DISPLAY VARIANT.
«enum»	***Destination-DisplayContext***	*Destination-DisplayContextEnum*	0:1	Context in which to use the variant name. Default is '*any*'. See allowed values below. + v1.1

«enum»	***DeliveryType***	*DeliveryMediaEnum*	1:1	Way a NOTICE is delivered, characterized by the delivery medium (voice, printed material, etc.). Default is '*any*'. See allowed values below.
	ShortName	*MultilingualString*	0:1	Short Name of DESTINATION DISPLAY VARIANT.
	SideText	*MultilingualString*	0:1	Text to display on side of vehicle associated with DESTINATION DISPLAY VARIANT.
	FrontText	*MultilingualString*	0:1	Front of vehicle text associated with DESTINATION DISPLAY VARIANT.
«cntd»	***vias***	*Via*	1:1	Via display texts for variant.

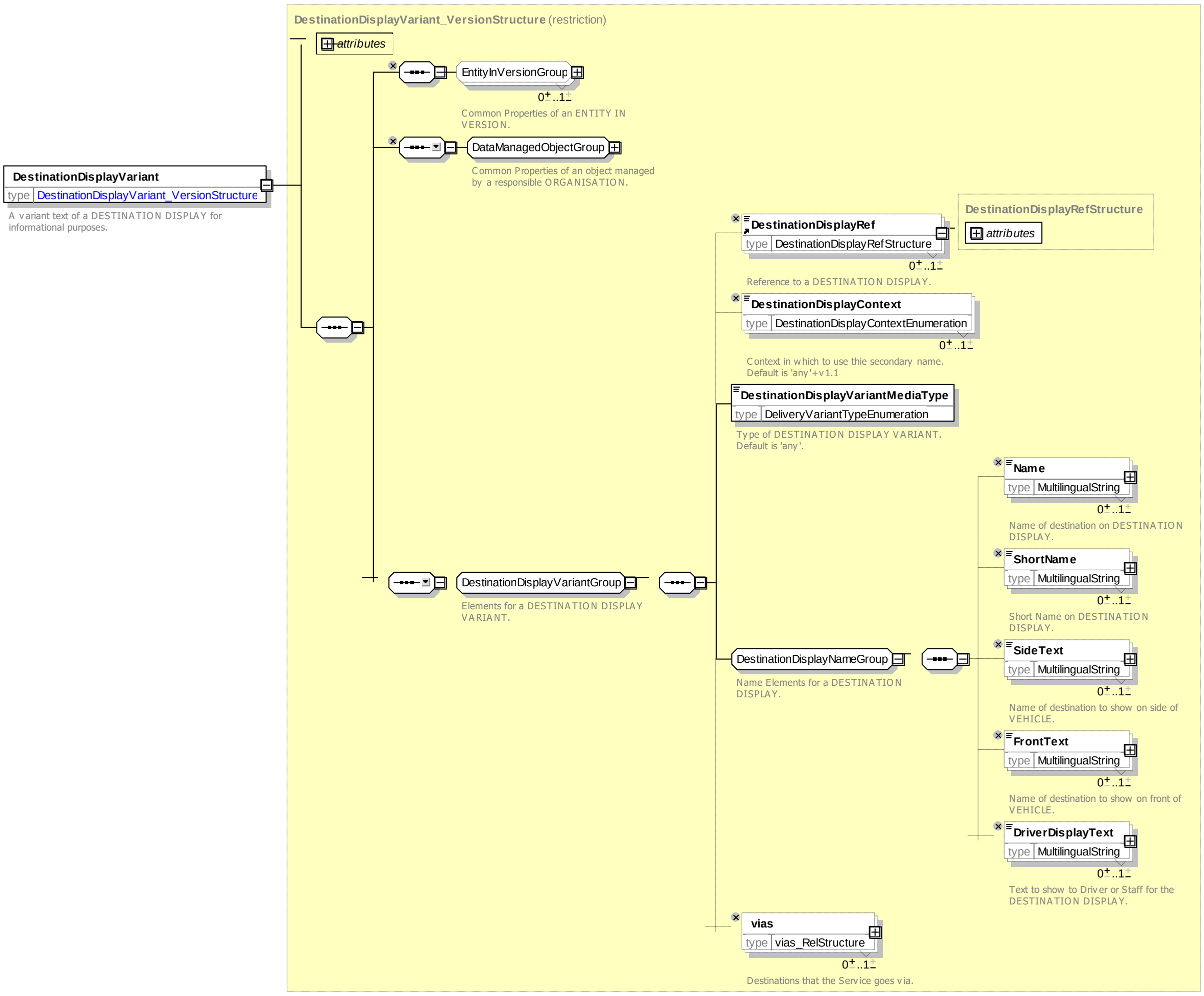


Figure 447 – *DestinationDisplayVariant* – XSD

8.3.4.8.3.1 DestinationDisplayContext – Allowed values

The following table shows the allowed values for DestinationDisplay***Context*** **(***SecondaryNameContext Enumeration***)*.***

Table 374 – DestinationDisplay*Context* – Allowed values

Value	Description
any	Any context.
continueTo	Use as “continue to” value.
endsAt	Use as “ends at” value.
transferAt	Use as “transfer at” value.
via	Use as “via” value.
inMessage	Useas “InMEsage” value.
unknown	Unknown context.

8.3.4.8.3.2 ***DestinationDisplayVariantMediaType*** **– Allowed values**

The following table shows the allowed values for ***DestinationDisplayVariantMediaType (****DeliveryVariantType-Enumeration****).***

Table 375 – *DestinationDisplayVariantMediaType* – Allowed Values

Value	Description
printed	Printed matter.
textToSpeech	Media is audio generated with a Text To Speech Engine.
recordedAnnouncement	Media is audio from recorded announcements.
web	Online web page.
mobile	Online mobile device content.
other	Other media.

8.3.4.8.4 Via – Model Element

A VIA is a POINT used as a ROUTE POINT dedicated to distinguish two possible paths (ROUTEs) between an origin and a destination.

Table 376 – *Via* – Element

Classification	Name		Type	Cardinality	Description
::>	*::>*		*VersionedChild*	*::>*	VIA inherits from VERSIONED CHILD.
«PK»	***id***		*ViaIdType*	1:1	Identifier of VIA.
«FK»	a	***Destination DisplayRef***	*DestinationDisplayRef*	0:1	DESTINATION DISPLAY corresponding to VIA description.
	b	***Name***	*MultiLingualString*	1:1	Name of VIA.
«FK»	***RoutePointRef***		*RoutePointRef*	0:1	Reference to ROUTE POINT corresponding to VIA name, if any.
«enum»	***ViaType***		*ViaTypeEnum*	0:1	Classification of meaning of VIA content. See allowed values below.

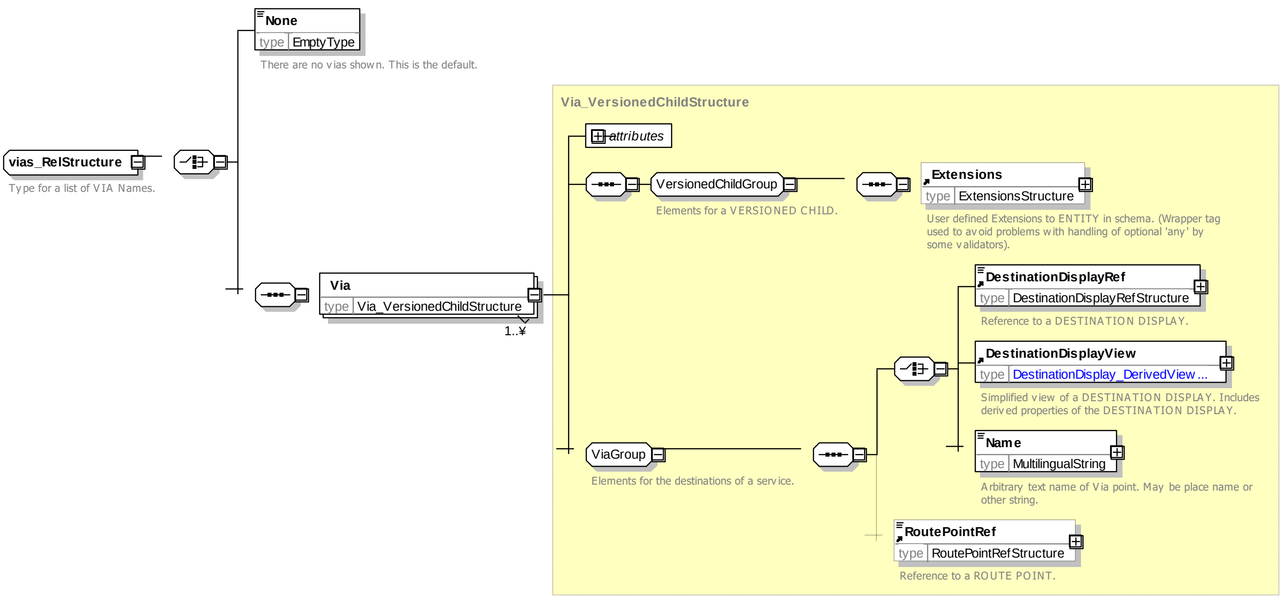


Figure 448 – *Via* – XSD

ViaType – Allowed values

The following table shows the allowed values for ***ViaType*** (*ViaTypeEnum*eration)**.**

Table 377 –*ViaType* – Allowed Values

Value	Description
stopPoint	Stop Point
name	Name (not a stop point)
other	Other

8.3.4.8.5 AllowedLineDirection – Model Element

An allowed DIRECTION that can be used on a given ROUTE.

This can be used to validate the selection of allowed values.

Table 378 – *AllowedLineDirection* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	ALLOWED LINE DIRECTION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*AllowedDirectionIdType*	1:1	Identifier of ALLOWED LINE DIRECTION.
«FK»	***LineRef***	*LineRef*	0:1	Reference to a LINE of ALLOWED LINE DIRECTION.
«FK»	***DirectionRef***	*DirectionRef*	1:1	Reference to a DIRECTION of ALLOWED LINE DIRECTION.
«cntd»	***Presentation***	*Presentation*	0:1	Pesentation to use for DIRECTION. +v1.1.

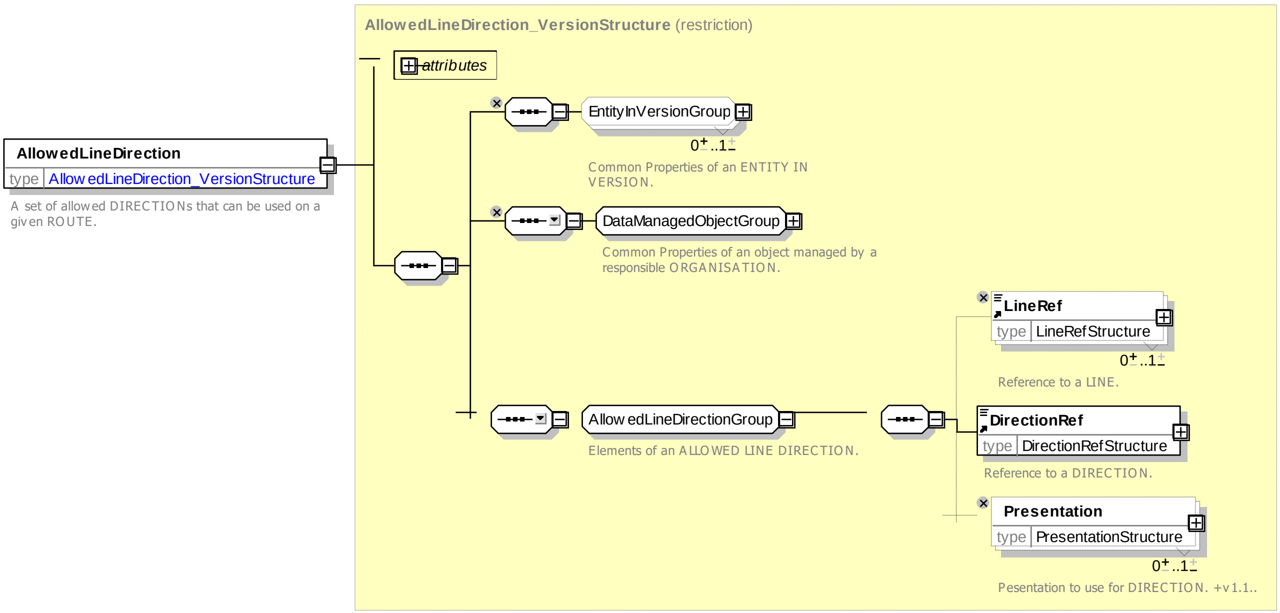


Figure 449 – *AllowedLineDirection – XSD*

8.3.4.8.6 GroupOfLines – Model Element

A grouping of LINEs which will be commonly referenced for a specific purpose.

Table 379 – *GroupOfLines* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF LINEs inherits from GROUP OF ENTITies.
«PK»	***id***	*GroupOfLinesIdType*	1:1	Identifier of a GROUP OF LINEs.
	UseToExclude	*xsd:boolean*	0:1	Whether contents of group represent excluded LINEs. +v1.1
«cntd»	***members***	*LineRef*	0:*	References to members of a GROUP of LINEs.
«FK»	***MainLineRef***	*LineRef*	0:1	Main LINE of a GROUP of LINEs.
«enum»	***TransportMode***	*AllVehicleModes-OfTransportEnum*	0:1	Primary Transport MODE of GROUP OF LINEs. See TRANSPORT MODE for allowed values.
«enum»	***Transport-SubMode***	*TransportSubmodeEnum*	0:1	A submode of a Public Transport MODE of GROUP OF LINEs. See TRANSPORT MODE for allowed values. +v1.1
«enum»	***GroupOfLines-Type***	*GroupOfLinesTypeEnum*	0:1	Classification of GROUP OF LINEs. See below for allowed values. +v1.1
«enum»	***PaymentMethods***	*PaymentMethodEnum*	0:*	PAYMENT METHODs accepted for this GROUP OF LINEs. See Service Restriction Model for allowed values. +v1.1
«cntd»	***typesOfPayment-Method***	*TypeOfPaymentMethod-Ref*	0:*	TYPEs OF PAYMENT METHOD accepted for this GROUP OF LINEs. +v1.1.
«enum»	***Purchase-Moments***	*PurchaseMomentEnum*	0:*	PURCHASE MOMEMNTs for this GROUP OF LINEs. See Service Restriction Model for allowed values. +v1.1

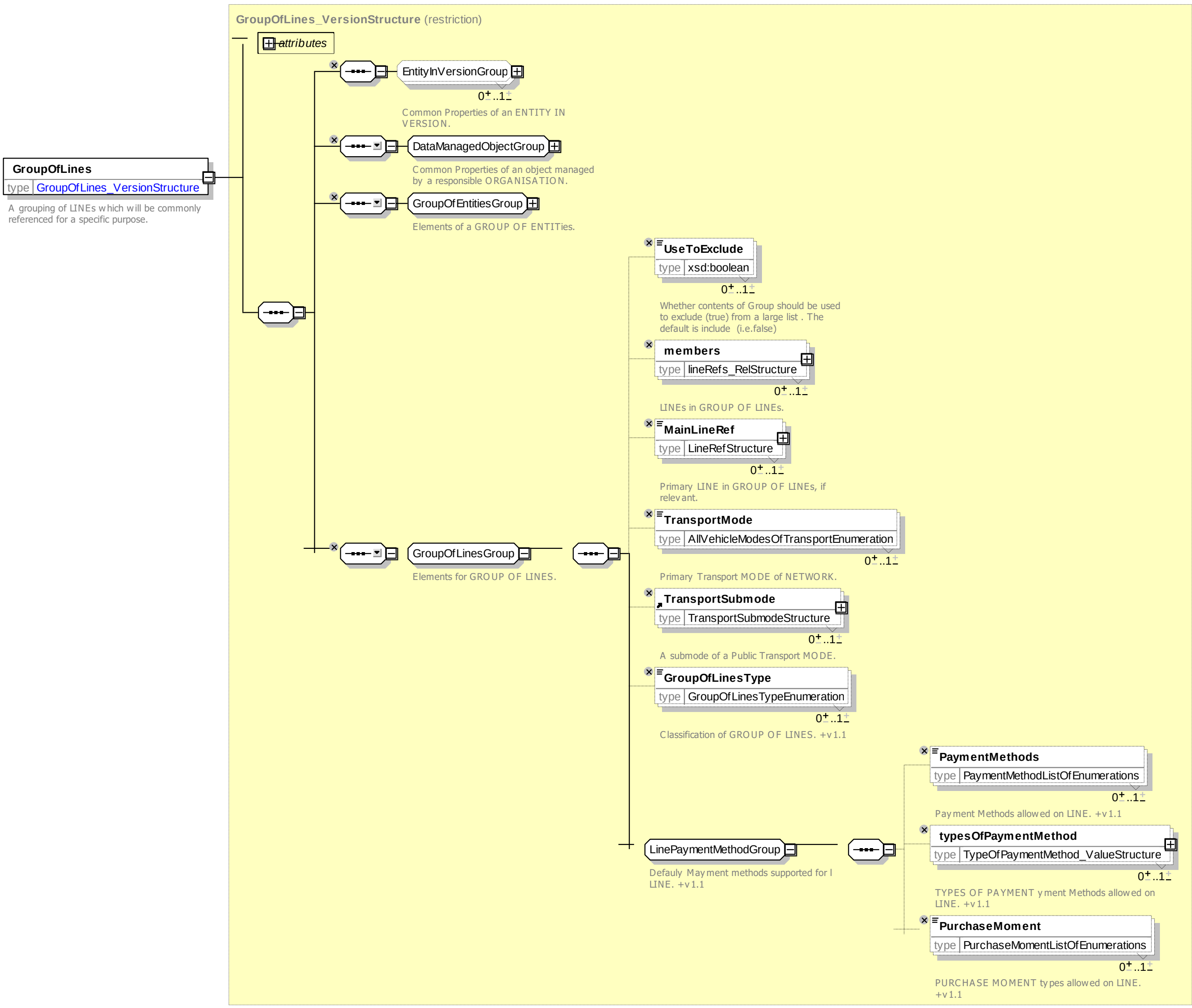


Figure 450 – GroupOfLines – XSD

8.3.4.8.6.1 GroupOfLinesType – Allowed values

The following table shows the allowed values for ***GroupOfLinesType (****GroupOfLinesTypeEnumeration****).***

Table 380 – *GroupOfLinesType* – Allowed Values

Value	Description
marketing	Purpose is marketing.
administrative	Purpose is administration.
scheduling	Purpose is creating scheduless
control	Purpose is operational control.
tariff	Purpose is for tariffs.
other	Other purpose.

8.3.4.8.7 Network – Model Element

A named grouping of LINEs under which a transport network is known.

Table 381 – *Network* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*GroupOfLines*	*::>*	NETWORK inherits from GROUP OF LINEs.
«PK»	***id***	*NetworkIdType*	1:1	Identifier of NETWORK.
	Name	*MultilingualString*	1:1	Name of NETWORK. Note that by convention the name is mandatory (enven if, as inherited from GroupOfEntities, it is not formally mandatory).
«FK»	***Transport-OrganisationRef***	*(TransportOrganisationRef) OperatorRef | AuthorityRef*	0:1	Reference to a transport organisation (an OPERATOR or AUTHORITY) in overall charge of the NETWORK.
«cntd»	***groupsOf-Operators***	*GroupOfOperators*	0:*	GROUPs OF OPERATORs associated with the Network. May include both inclusion and exclusions. +v1.1
«cntd»	***groupsOfLines***	*GroupOfLines*	0:*	Groups of LINEs in NETWORK.
«cntd»	***tariffZones***	*TariffZone*	0:*	TARIFF ZONEs in NETWORK.

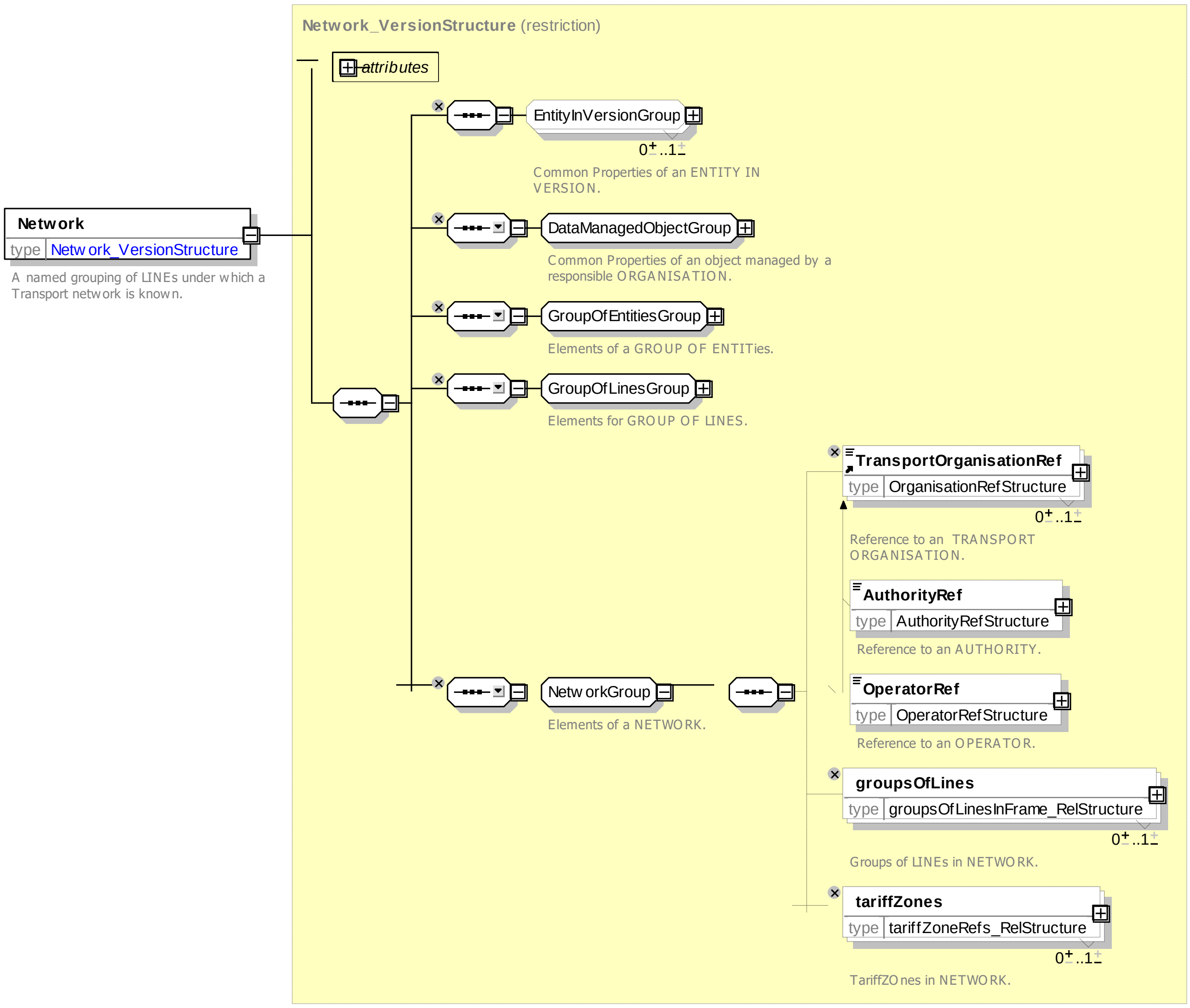


Figure 451 – Network – XSD

8.3.4.8.8 TypeOfLine – Model Element

A classification for the general orientation of ROUTEs.

Table 382 – *TypeOfLine* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*TypeOfEntity*</u>	::>	TYPE OF LINE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfLineIdType*	1:1	Identifier of TYPE OF LINE.

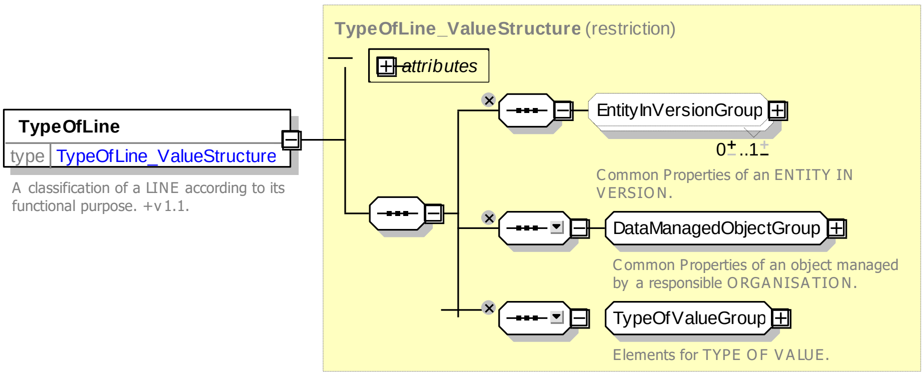


Figure 452 – *TypeOfLine – XSD*

8.3.4.9 Route Elements – XML Examples

The following XML fragments show the definition of a route that runs between three stops (A B and C) via a number of intermediate points.

8.3.4.9.1 XML Example – Route Points

EXAMPLE

```
<RoutePoint version="any" id="mybus:Rtpt_RT_24_SSP_001@">
    <Name>Point in Road near Stop A</Name>
    <Location>
        <Longitude>53.0000</Longitude>
        <Latitude>0.1000</Latitude>
    </Location>
</RoutePoint>
<RoutePoint version="any" id="mybus:Rtpt_RT_24_SSP_002@">
    <Name>Point in Road near Stop B</Name>
    <Location>
        <Longitude>53.0000</Longitude>
        <Latitude>0.2000</Latitude>
    </Location>
</RoutePoint>
<RoutePoint version="any" id="mybus:Rtpt_RT_24_SSP_003@">
    <Name>Point in Road near Stop C</Name>
    <Location>
        <Longitude>53.0000</Longitude>
        <Latitude>0.3000</Latitude>
    </Location>
</RoutePoint>
<RoutePoint version="any" id="mybus:Rtpt_RT_24_1">
    <Name>Point in Road between Stop A and B 1</Name>
    <Location>
        <Longitude>53.0000</Longitude>
        <Latitude>0.4000</Latitude>
    </Location>
</RoutePoint>
<RoutePoint version="any" id="mybus:Rtpt_RT_24_2">
    <Name>Point in Road between Stop A and B 2</Name>
    <Location>
        <Longitude>53.0000</Longitude>
        <Latitude>0.5000</Latitude>
    </Location>
</RoutePoint>
```

```xml
    <RoutePoint version="any" id="mybus:Rtpt_RT_24_3">
        <Name>Point in Road between Stop A and B 3</Name>
        <Location>
            <Longitude>53.0000</Longitude>
            <Latitude>0.6000</Latitude>
        </Location>
    </RoutePoint>
    <RoutePoint version="any" id="mybus:Rtpt_RT_24_4">
        <Name>Point in Road between Stop A and B 4</Name>
        <Location>
            <Longitude>53.0000</Longitude>
            <Latitude>0.7000</Latitude>
        </Location>
    </RoutePoint>
    <RoutePoint version="any" id="mybus:Rtpt_RT_24_5">
        <Name>Point in Road between Stop B and C 1</Name>
        <Location>
            <Longitude>53.0000</Longitude>
            <Latitude>0.7500</Latitude>
        </Location>
    </RoutePoint>
</routePoints>
```

8.3.4.9.2 XML Example – Route Links

EXAMPLE

```xml
        <routeLinks>
            <!--- ====== ROUTE LINKs ==OUTBOUND======== -->
            <RouteLink version="any" id="mybus:RL_24o_RT_SSP_001@_to_RT_24_1">
                <FromPointRef version="any" ref="mybus:Rtpt_RT_24_SSP_001@"/>
                <ToPointRef version="any" ref="mybus:Rtpt_RT_24_1"/>
            </RouteLink>
            <RouteLink version="any" id="mybus:RL_24o_RT_24_1_to_RT_24_2">
                <FromPointRef version="any" ref="mybus:Rtpt_RT_24_1"/>
                <ToPointRef version="any" ref="mybus:Rtpt_RT_24_2"/>
            </RouteLink>
            <RouteLink version="any" id="mybus:RL_24o_RT_24_2_to_RT_24_3">
                <FromPointRef version="any" ref="mybus:Rtpt_RT_24_2"/>
                <ToPointRef version="any" ref="mybus:Rtpt_RT_24_3"/>
            </RouteLink>
            <RouteLink version="any" id="mybus:RL_24o_RT_24_3_to_RT_24_4">
                <passingThrough>
                    <PointOnLink version="any" id="mybus:pol_SSP_002_to_SSP_077_01">
                        <Name>Goes through Bravo</Name>
                        <DistanceFromStart>300</DistanceFromStart>
                        <ScheduledStopPointRef version="any" ref="mybus:SSP_002"/>
                    </PointOnLink>
                </passingThrough>
                <FromPointRef version="any" ref="mybus:Rtpt_RT_24_3"/>
                <ToPointRef version="any" ref="mybus:Rtpt_RT_24_4"/>
            </RouteLink>
            <RouteLink version="any" id="mybus:RL_24o_RT_24_4_to_RT_24_5">
                <Name>Route point fopr stop 2 lies on this link</Name>
                <FromPointRef version="any" ref="mybus:Rtpt_RT_24_4"/>
                <ToPointRef version="any" ref="mybus:Rtpt_RT_24_5"/>
            </RouteLink>
            <RouteLink version="any" id="mybus:RL_24o_RT_24_5_to_RT_24_SSP_003@">
                <FromPointRef version="any" ref="mybus:Rtpt_RT_24_5"/>
```

```
                <ToPointRef version="any" ref="mybus:Rtpt_RT_24_SSP_003@"/>
            </RouteLink>
        </routeLinks>
```

8.3.4.9.3 XML Example – Route

EXAMPLE

```
<Route version="any" id="mybus:RT_24o">
    <Name>Line 24 Alpha to Charley Green</Name>
    <DirectionType>outbound</DirectionType>
    <DirectionRef version="any" ref="mybus:DR_Westbound"/>
    <pointsInSequence>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_001" order="1">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_SSP_001@"/>
        </PointOnRoute>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_002" order="2">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_1"/>
        </PointOnRoute>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_003" order="3">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_2"/>
        </PointOnRoute>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_004" order="4">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_3"/>
        </PointOnRoute>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_005" order="5">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_4"/>
        </PointOnRoute>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_006" order="6">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_5"/>
        </PointOnRoute>
        <PointOnRoute version="any" id="mybus:POR_Rt24o_007" order="7">
            <RoutePointRef version="any" ref="mybus:Rtpt_RT_24_SSP_003@"/>
        </PointOnRoute>
    </pointsInSequence>
</Route>
```

8.3.4.9.4 XML Example – Line

The following XML fragments show the definition of a LINE. The Line is associated with a route via a VEHICLE JOURNEY and a JOURNEY PATTERN that follows the ROUTE (see later).

EXAMPLE

```
<Line version="any" id="mybus:LN_24">
    <Name>Line 24 Alpha to Charley Green</Name>
    <ShortName>Line 24</ShortName>
    <TransportMode>bus</TransportMode>
    <PublicCode>24</PublicCode>
</Line>
```

8.3.4.9.5 XML Example – Destination Display

The following XML fragments show the definition of the DESTINATION DISPLAY elements associated with either end of the simple route shown above.

EXAMPLE

```
<DestinationDisplay version="any" id="mybus:DST_Alpha">
```

```
    <Name>Alpha Green</Name>
    <ShortName>Charley G</ShortName>
    <PublicCode>ALPH</PublicCode>
</DestinationDisplay>
<DestinationDisplay version="any" id="mybus:DST_Charley">
    <Name>Charley Green</Name>
    <ShortName>Charley G</ShortName>
    <PublicCode>CHAS</PublicCode>
</DestinationDisplay>
```

8.3.5 Line Network

The Line Network model provides a means of describing the overall topology of a route - or rather all the ROUTEs of a LINE - including branches and alternatives etc. This is in contrast to ROUTEs, SERVICE PATTERNs, JOURNEY PATTERNs, etc., which show a single path through the network for a single journey.

The grouping of patterns into a LINE NETWORK has uses for visualisations – for example a schematic map of a network, and for relating SITUATIONs to affected parts of the network.

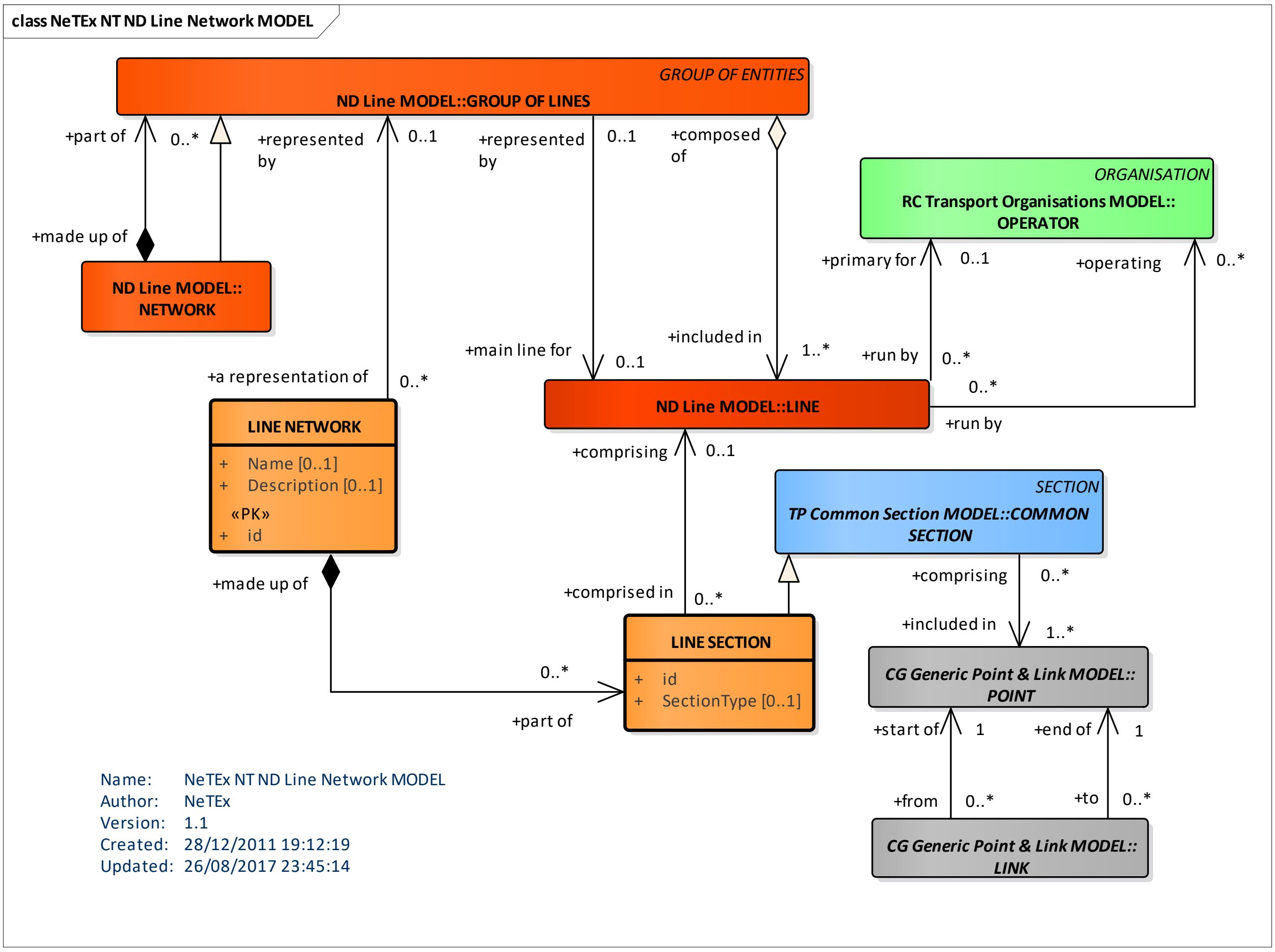


Figure 453 – LINE NETWORK – Conceptual MODEL (UML)

8.3.5.1 LINE NETWORK – Conceptual MODEL

A LINE NETWORK represents the network topology as a names set of LINE SECTIONs. Each LINK SECTION is a sequence of SCHEDULED STOP POINTs and/or SERVICE LINKs.

8.3.5.2 Line Network – Example

The following diagram shows an example of a LINE NETWORK for the Northern Line of the London Underground. It includes a number of branches and covers both direction of the line.

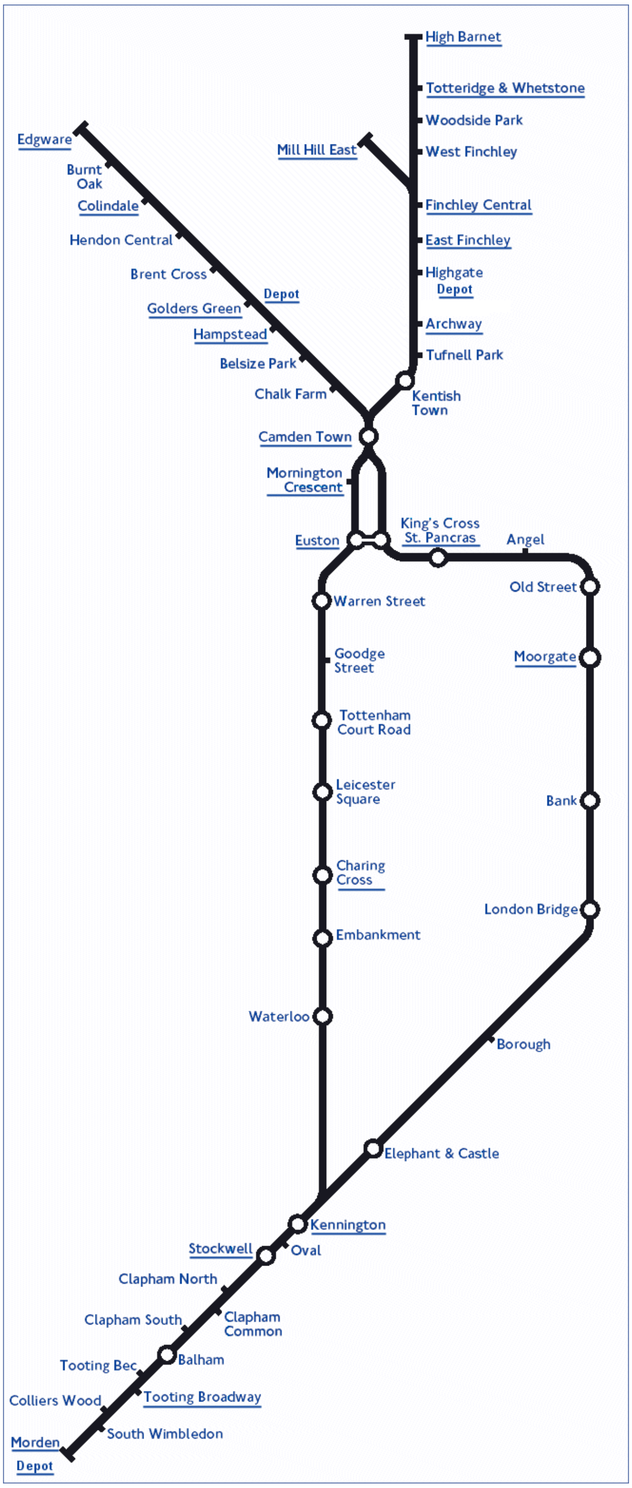


Figure 454 – Example of a Line Network LUL Northern Line

8.3.5.3 Line Network – Physical Model

Figure 455 – Line Network – Physical Model (UML) shows the physical representation of the LINE NETWORK model.

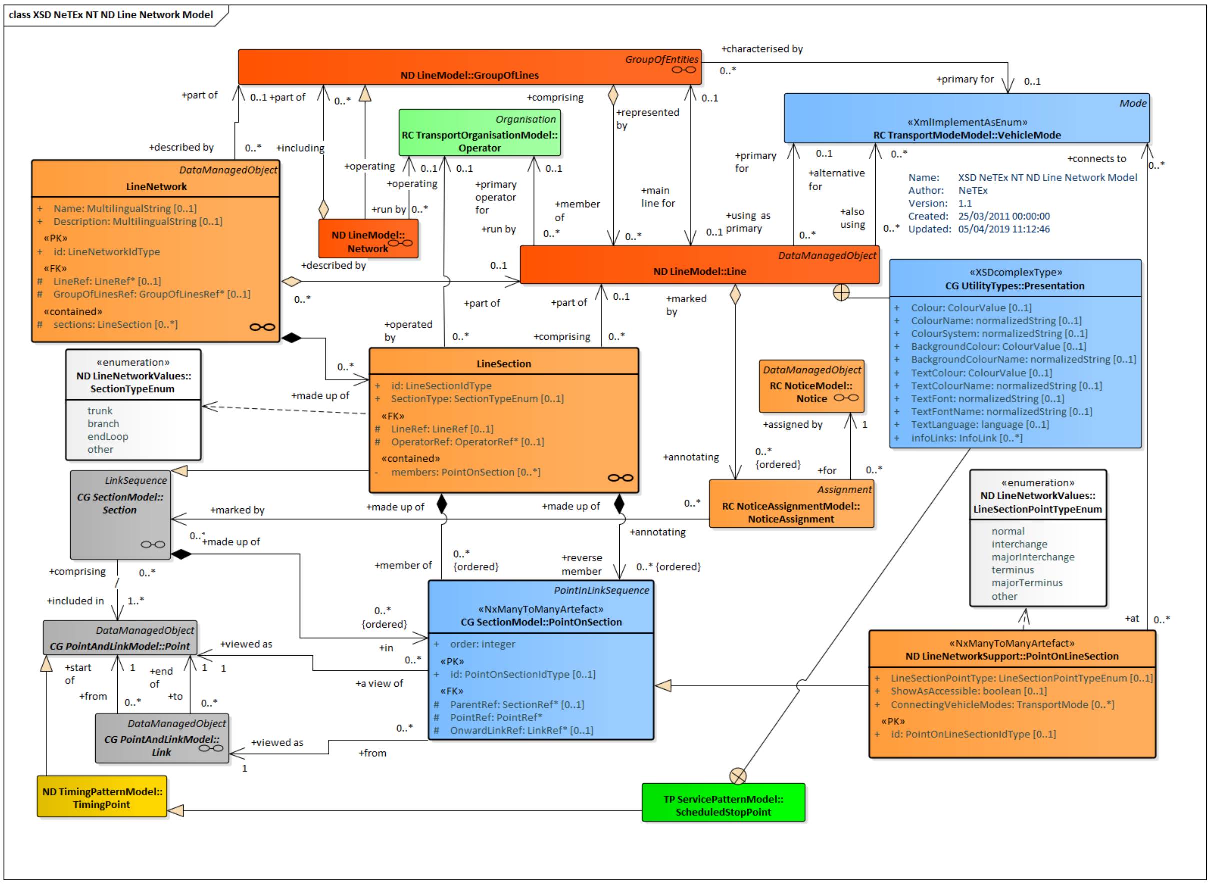


Figure 455 – Line Network – Physical Model (UML)

8.3.5.4 Line Network – Attributes and XSD

8.3.5.4.1 LineNetwork – Model Element

The Topological structure of a NETWORK as a graph of LINE SECTIONs. This allows the branches and loops of a LINE to be described as a whole.

Table 383 – *LineNetwork* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	LINE NETWORK inherits from DATA MANAGED OBJECT.
«PK»	***id***	*LineNetworkIdType*	1:1	Identifier of LINE NETWORK.
	Name	*MultilingualString*	1:1	Name of LINE NETWORK.
	Description	*MultilingualString*	0:1	Description of LINE NETWORK.
«FK»	***GroupOfLinesRef***	*GroupOfLinesRef*	0:1	Reference to a GROUP OF LINEs.
«FK»	***LineRef***	*LineRef*	0:1	LINE that this LINE NETWORK describes.
«cntd»	***sections***	*LineSection*	0:*	LINE SECTIONs in this LINE NETWORK.

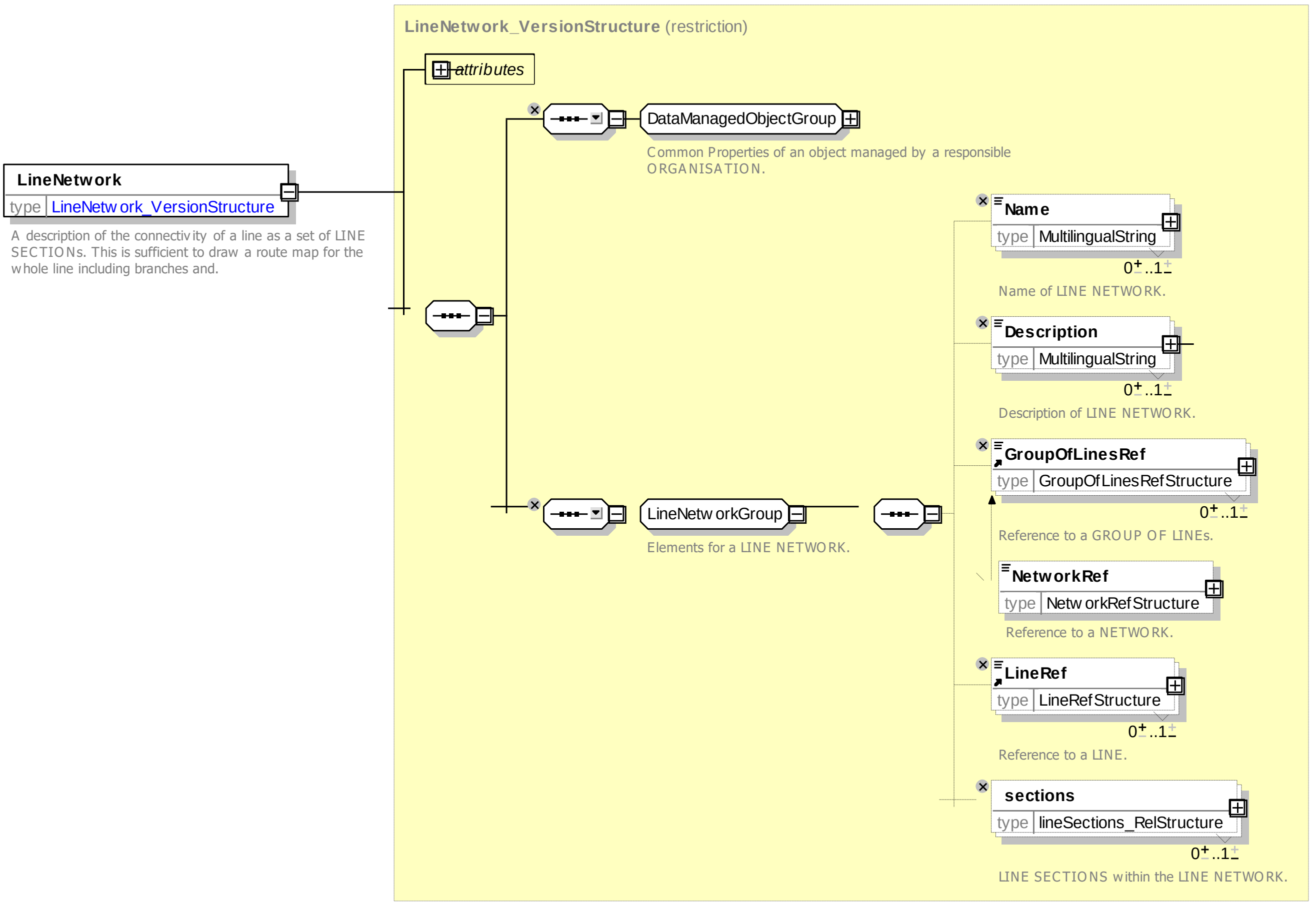


Figure 456 – *LineNetwork – XSD*

8.3.5.4.2 LineSection – Model Element

A section of a LINE NETWORK comprising an edge between two nodes. It is not directional.

Table 384 – *LineSection* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Section*</u>	::>	LINE SECTION inherits from SECTION.
	id	*CommonSectionIdType*	1:1	Identifier of LINE SECTION.
«cntd»	***pointsOn-Section***	<u>*PointOnLineSection*</u>	0:*	Members in forwards direction. POINTs on LINE SECTION that are in LINE SECTION. +v1.1
«cntd»	***members***	<u>*CommonSectionPointMember*</u>	0:*	Members in forwards direction. COMMON SECTIONs that are in LINE SECTION.
«cntd»	***reversePoints-OnSection***	<u>*PointOnLineSection*</u>	0:*	Members in reverse direction. POINTs ON LINE SECTION that are in LINE SECTION. +v1.1
«cntd»	***Reverse-Members***	<u>*CommonSectionPointMember*</u>	0:*	DEPRECATED Members in reverse direction. COMMON SECTIONs that are in LINE SECTION.

«enum»	***SectionType***	*SectionTypeEnumeration*	0:1	Type of LINE SECTION.
«FK»	***LineRef***	*LineRef*	0:1	LINE for which this is a LINE SECTION.

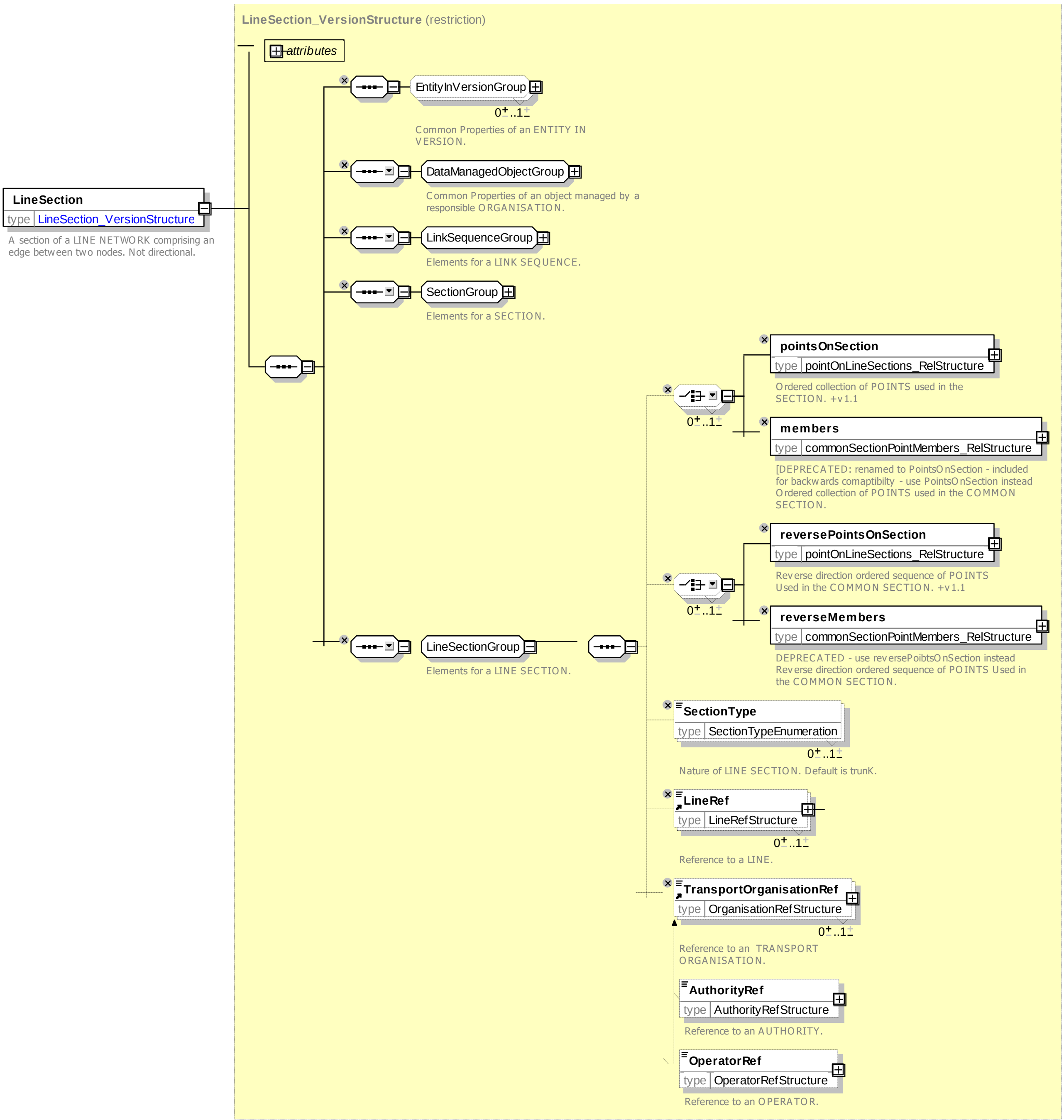


Figure 457 – *LineSection – XSD*

SectionType – Allowed values

The following table shows the allowed values for ***SectionType*** (*SectionTypeEnum*eration***).***

Table 385 –*fLineType* – Allowed Values

Value	Description
trunk	Section is main trunk
branch	Section is a branch (with an end free)
eyeBranch	Section is a n internal branch (with both ends lnked)ng)
endLoop	Section is a loop at the end of a line
other	Other

8.3.5.4.2.1 XML Example of Line Section

The following XML code fragment describes the topography of a LINE NETWORK of seven stops that is branched at both ends.

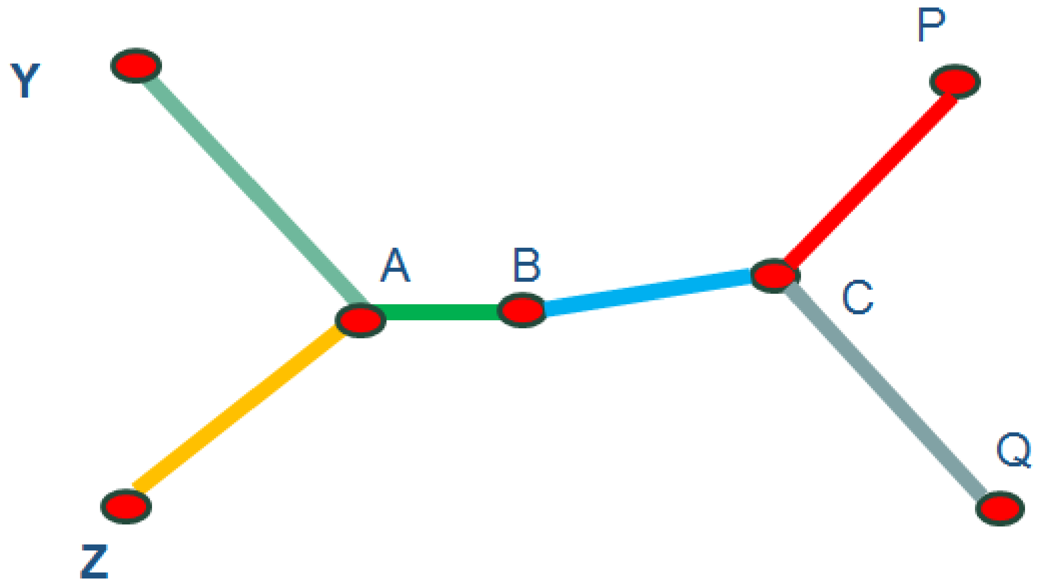


Figure 458 – Line sections – Example

The sections are thus Y-A, Z-A, A-B-C, C-Q and C-P. There are SERVICE LINKS in both directions, so for each section there are a set of points and LINE SECTIONs.

EXAMPLE XML Example of StopPlace

```
<LineNetwork version="any" id="ntp_ln_24">
    <Name>Line 24 with all branches</Name>
    <LineRef version="any" ref="ln_24"/>
    <sections>
        <LineSection version="any" id="LS_Yankee_Alpha">
            <members>
                <CommonSectionPointMember version="any" id="LSM_Yankee_Alpha_01" order="1">
                    <Description>Yankee to Alpha: West-East</Description>
                    <ScheduledStopPointRef version="any" ref="Yankee"/>
                    <ServiceLinkRef version="any" ref="Yankee_to_Alpha"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember version="any" id="LSM_Yankee_Alpha_02" order="2">
                    <Description>Yankee to Alpha End: West-East</Description>
                    <ScheduledStopPointRef version="any" ref="Alpha"/>
                </CommonSectionPointMember>
            </members>
            <reverseMembers>
                <CommonSectionPointMember version="any" id="LSM_Yankee_Alpha_01r" order="1">
                    <Description>Alpha to Yankee: East-West</Description>
                    <ScheduledStopPointRef version="any" ref="Alpha"/>
                    <ServiceLinkRef version="any" ref="Alpha_to_Yankee"/>
```

```xml
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Yankee_Alpha_02r" order="2">
            <Description>Alpha to Yankee: East-West</Description>
            <ScheduledStopPointRef version="any" ref="Yankee"/>
        </CommonSectionPointMember>
    </reverseMembers>
    <SectionType>branch</SectionType>
</LineSection>
<LineSection version="any" id="LS_Zulu_Alpha">
    <members>
        <CommonSectionPointMember version="any" id="LSM_Zulu_Alpha_01" order="1">
            <Description>Zulu to Alpha: West-East</Description>
            <ScheduledStopPointRef version="any" ref="Zulu"/>
            <ServiceLinkRef version="any" ref="Zulu_to_Alpha"/>
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Zulu_Alpha_02" order="2">
            <Description>Zulu to Alpha: West-East</Description>
            <ScheduledStopPointRef version="any" ref="Alpha"/>
        </CommonSectionPointMember>
    </members>
    <reverseMembers>
        <CommonSectionPointMember version="any" id="LSM_Zulu_Alpha_01r" order="1">
            <Description>Zulu to Alpha: West-East</Description>
            <ScheduledStopPointRef version="any" ref="Alpha"/>
            <ServiceLinkRef version="any" ref="Alpha_to_Zulu"/>
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Zulu_Alpha_02r" order="2">
            <Description>Zulu to Alpha: West-East</Description>
            <ScheduledStopPointRef version="any" ref="Zulu"/>
        </CommonSectionPointMember>
    </reverseMembers>
    <SectionType>branch</SectionType>
</LineSection>
<LineSection version="any" id="LS_Alpha_Charley">
    <members>
        <CommonSectionPointMember version="any" id="LSM_Alpha_Charley_01" order="1">
            <ScheduledStopPointRef version="any" ref="Alpha"/>
            <ServiceLinkRef version="any" ref="Alpha_to_Bravo"/>
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Alpha_Charley_02" order="2">
            <ScheduledStopPointRef version="any" ref="Bravo"/>
            <ServiceLinkRef version="any" ref="Bravo_to_Charley"/>
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Alpha_Charley_03" order="3">
            <ScheduledStopPointRef version="any" ref="Charley"/>
        </CommonSectionPointMember>
    </members>
    <reverseMembers>
        <CommonSectionPointMember version="any" id="LSM_Alpha_Charley_01r" order="1">
            <ScheduledStopPointRef version="any" ref="Charley"/>
            <ServiceLinkRef version="any" ref="Charley_to_Bravo"/>
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Alpha_Charley_02r" order="2">
            <ScheduledStopPointRef version="any" ref="Bravo"/>
            <ServiceLinkRef version="any" ref="Bravo_to_Alpha"/>
        </CommonSectionPointMember>
        <CommonSectionPointMember version="any" id="LSM_Alpha_Charley_03r" order="3">
            <ScheduledStopPointRef version="any" ref="Alpha"/>
        </CommonSectionPointMember>
```

```xml
            </reverseMembers>
            <SectionType>trunk</SectionType>
        </LineSection>
        <LineSection version="any" id="LS_Charley_Papa">
            <members>
                <CommonSectionPointMember version="any" id="LSM_Charley_Papa_01" order="1">
                    <ScheduledStopPointRef version="any" ref="Charley"/>
                    <ServiceLinkRef version="any" ref="Charley_to_Papa"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember version="any" id="LSM_Charley_Papa_02" order="2">
                    <ScheduledStopPointRef version="any" ref="Papa"/>
                </CommonSectionPointMember>
            </members>
            <reverseMembers>
                <CommonSectionPointMember version="any" id="LSM_Charley_Papa_01r" order="1">
                    <ScheduledStopPointRef version="any" ref="Papa"/>
                    <ServiceLinkRef version="any" ref="Papa_to_Charley"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember version="any" id="LSM_Charley_Papa_02r" order="2">
                        <ScheduledStopPointRef version="any" ref="Charley"/>
                </CommonSectionPointMember>
            </reverseMembers>
            <SectionType>branch</SectionType>
        </LineSection>
        <LineSection version="any" id="LS_Charley_Quebec">
            <members>
                <CommonSectionPointMember version="any" id="LSM_Charley_Quebec_01" order="1">
                    <ScheduledStopPointRef version="any" ref="Charley"/>
                    <ServiceLinkRef version="any" ref="Charley_to_Quebec"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember version="any" id="LSM_Charley_Quebec_02" order="2">
                    <ScheduledStopPointRef version="any" ref="Quebec"/>
                </CommonSectionPointMember>
            </members>
                <reverseMembers>
                <CommonSectionPointMember version="any" id="LSM_Charley_Quebec_01r" order="1">
                    <ScheduledStopPointRef version="any" ref="Quebec"/>
                    <ServiceLinkRef version="any" ref="Quebec_to_Charley"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember version="any" id="LSM_Charley_Quebec_02r" order="2">
                    <ScheduledStopPointRef version="any" ref="Charley"/>
                </CommonSectionPointMember>
            </reverseMembers>
            <SectionType>branch</SectionType>
        </LineSection>
    </sections>
</LineNetwork>
```

8.3.6 Timing Pattern

8.3.6.1 TIMING PATTERN – Conceptual MODEL

The TIMING PATTERN model describes the point and links in the transport network that should be used for timing journeys. It is made up of TIMING POINTs and TIMING LINKs. Many TIMING POINTs are also SCHEDULED STOP POINTs but others may control points that are not stops. The TIMING PATTERN does not actually specify the actual time values – these are provided by the JOURNEY TIME and other models in the context of specific JOURNEY PATTERNs and VEHICLE JOURNEYs (see later).

The same TIMING POINTs and TIMING LINKs may be used in many different TIMING PATTERNs.

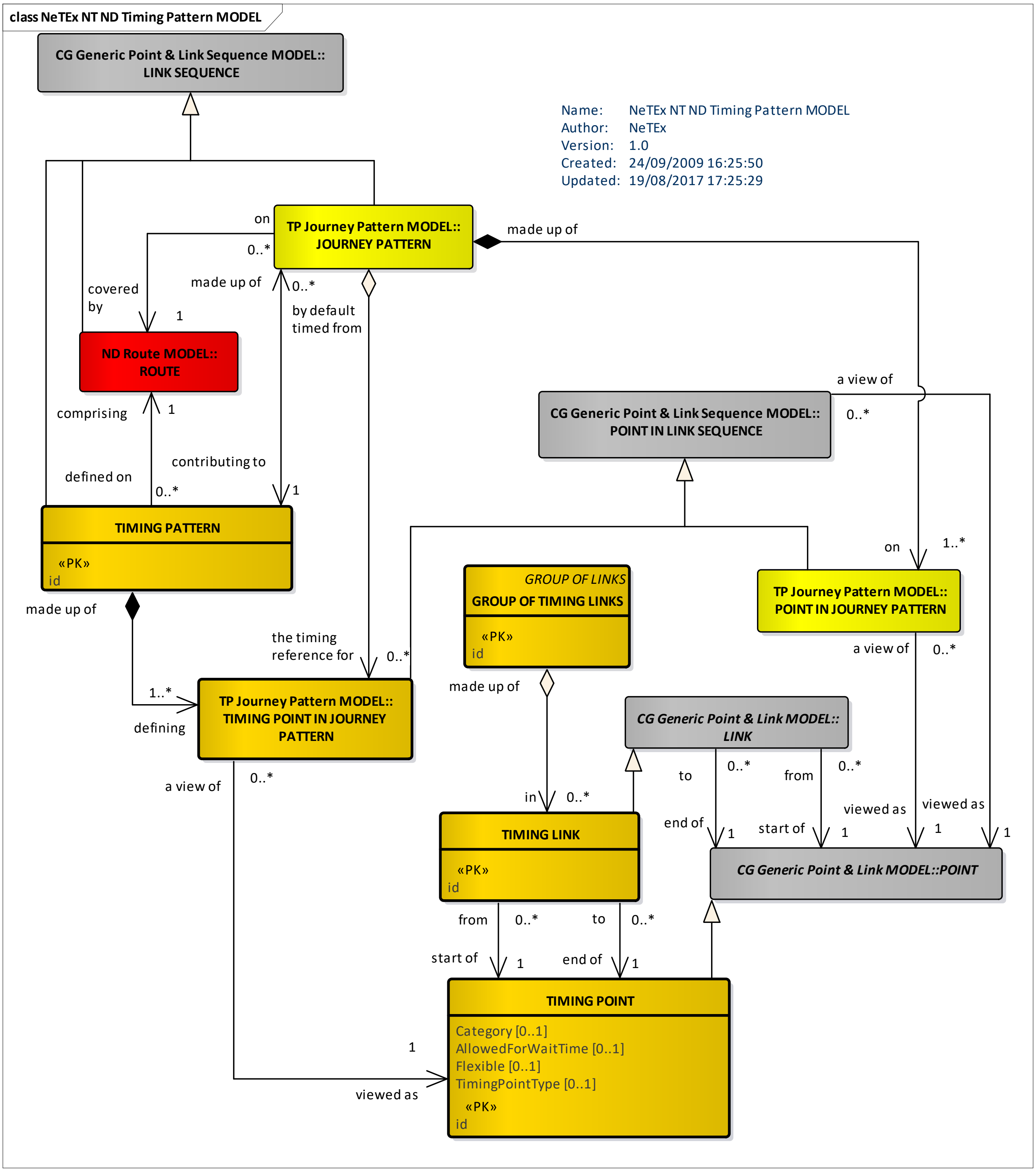


Figure 459 – TIMING PATTERN – Conceptual MODEL (UML)

8.3.6.2 Timing Pattern – Example

8.3.6.2.1 Timing Pattern in bus operation – Example

This example shows how TIMING PATTERNs for bus operation; TIMING POINTs determine the TIMING PATTERN (green) independently from the SERVICE PATTERN (red) defined by the SCHEDULED STOP POINTs.

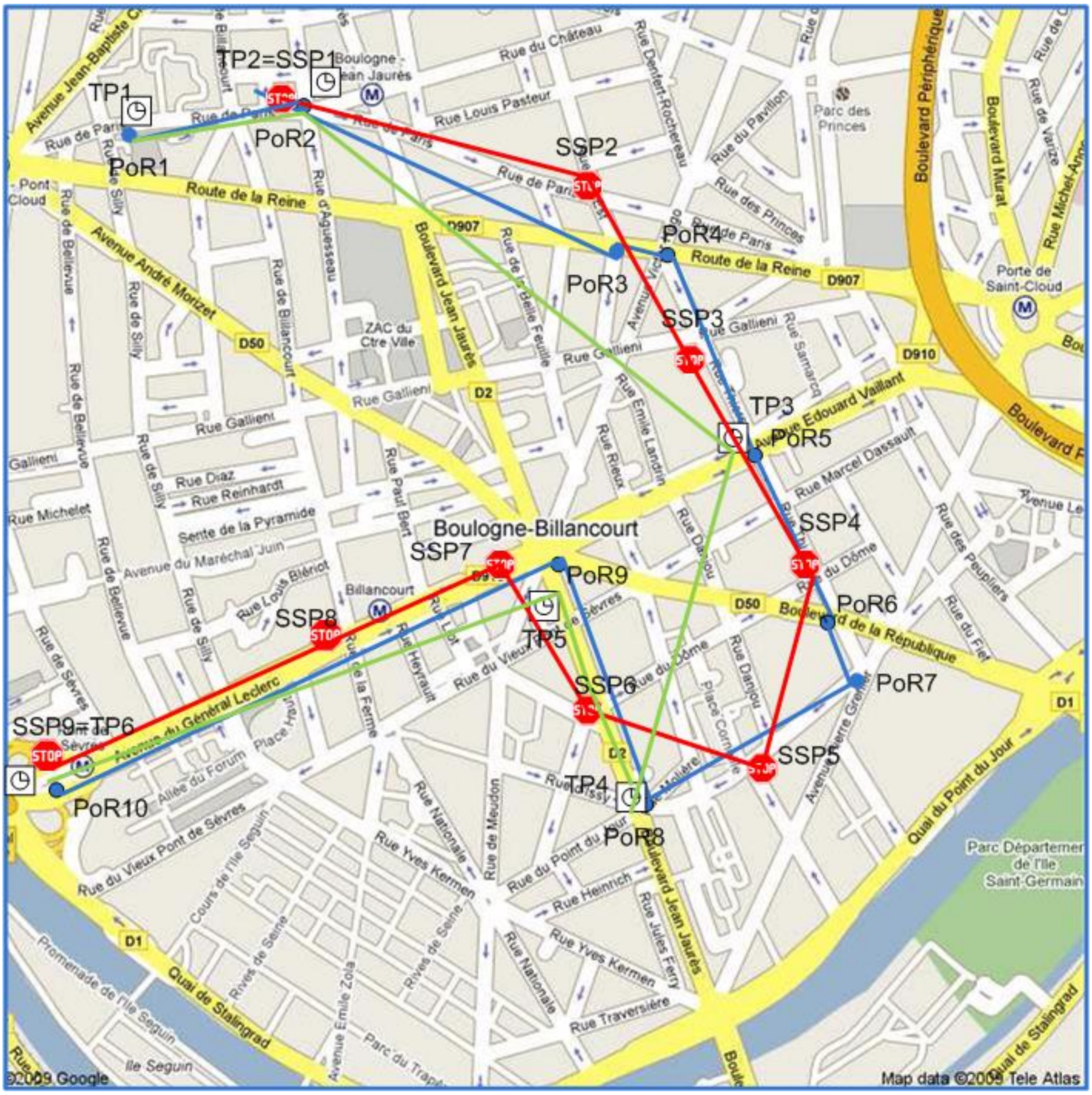


Figure 460 – Timing Link and Timing Points example

8.3.6.2.2 Simple Rail Timing Pattern–Example

The following fictional example illustrates the use of timing patterns for a simple rail service. On the top are shown a number of different SERVICE PATTERNs followed by different trains on the outbound London to Paris Eurostar route. Below are shown a hypothetical set of TIMING LINKs and TIMING POINTs that can be reused to describe the different TIMING PATTERNs found. In reality there would probably be many more intermediate

TIMING POINTs; this example includes just two extra TIMING POINTs (entry and exit from the tunnel) that are not SCHEDULED STOP POINTs.

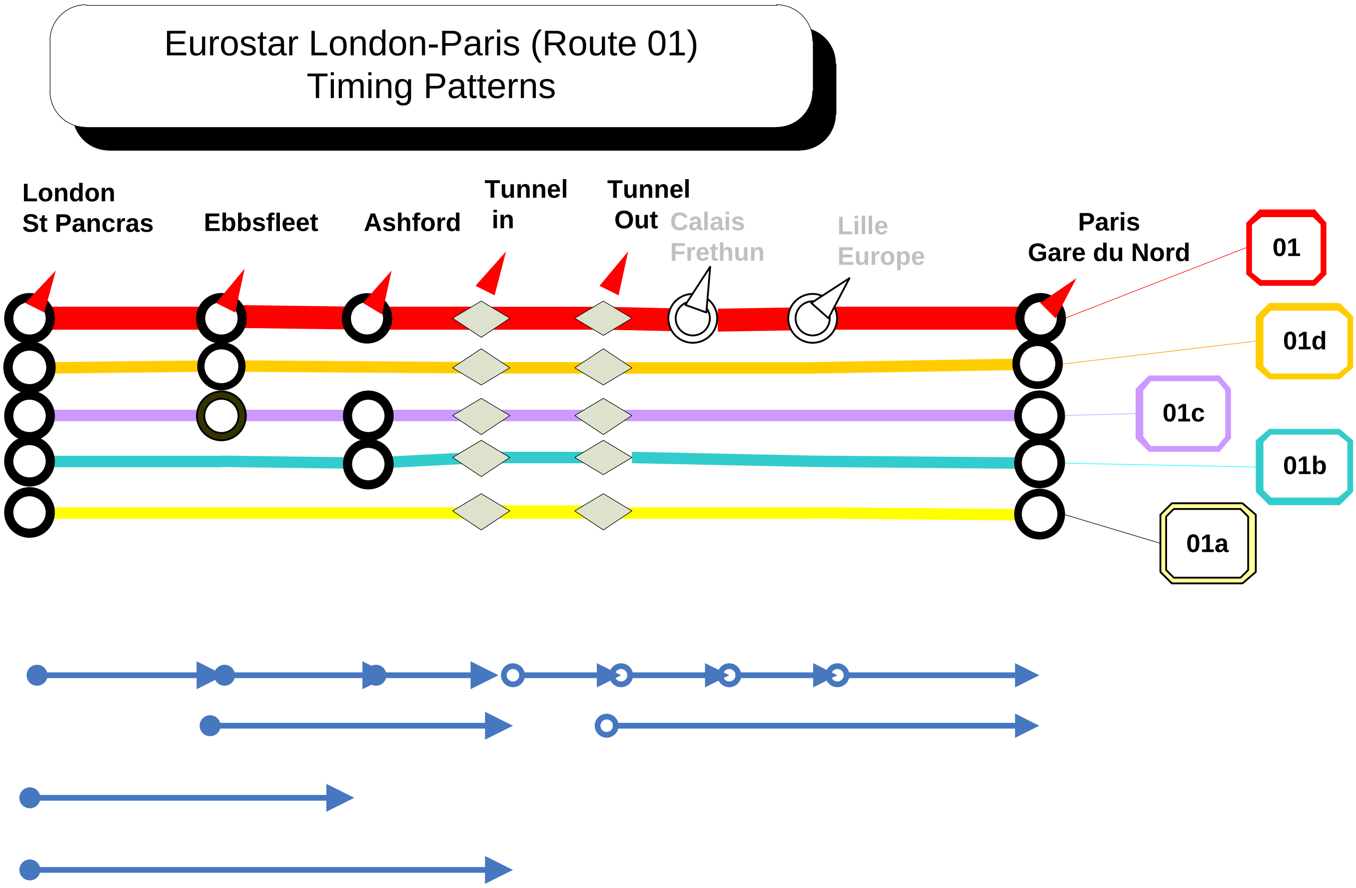


Figure 461 – Rail Timing Link and Timing Points example

The following figure shows some hypothetical timing patterns composed from the timing points and links. The actual timings (run times on TIMING LINKs and wait times on TIMING POINTs) are distinct from the timing patterns. Some SCHEDULED STOP POINTs are used as TIMING POINTs even if the train does not stop at them. Different sets of timings may be associated with the same TIMING PATTERN for different SERVICE JOURNEYs.

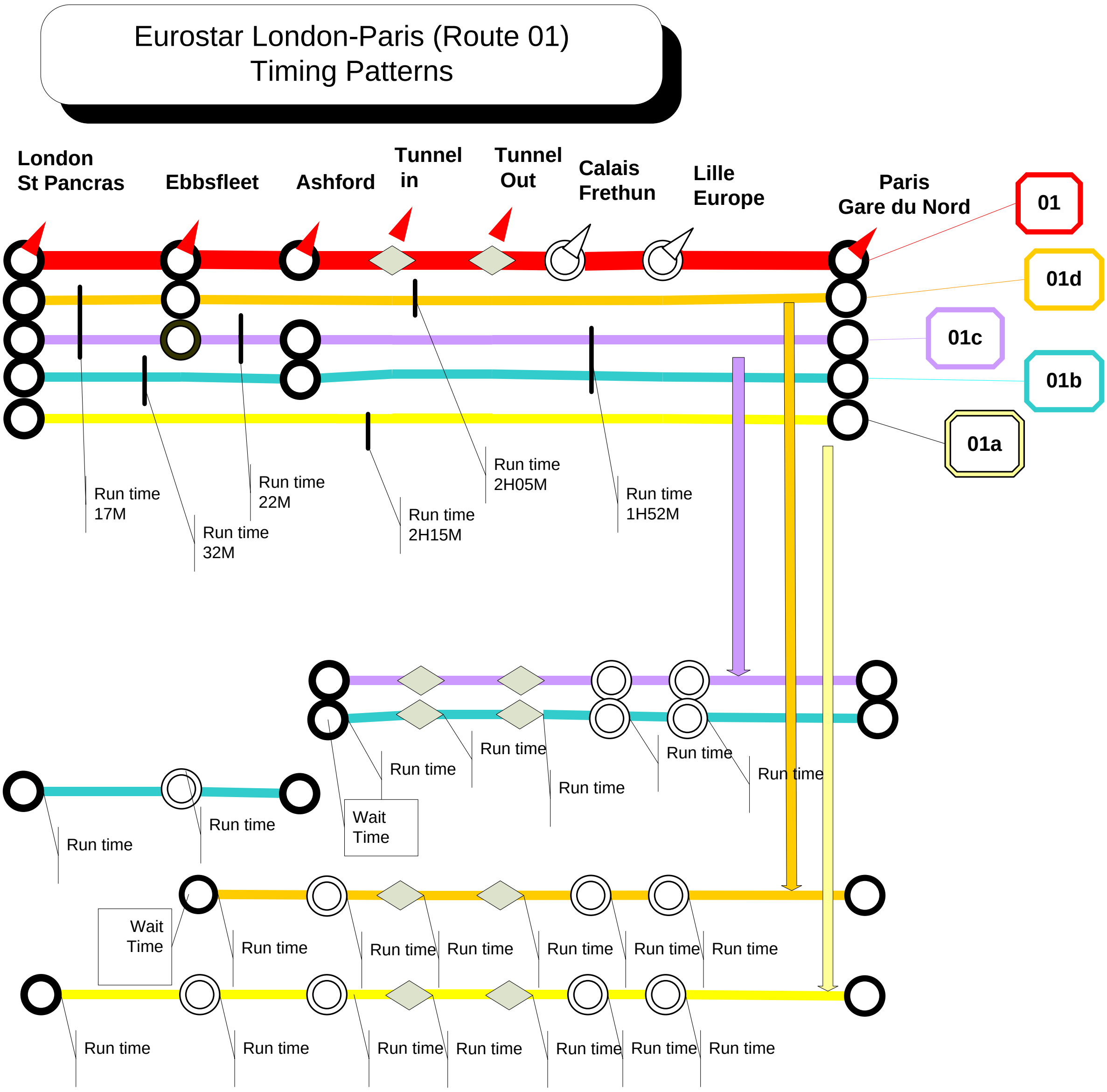


Figure 462 – Rail Timing Pattern Example

8.3.6.3 Timing Pattern – Physical Model

The following figure shows the physical representation of the TIMING PATTERN model.

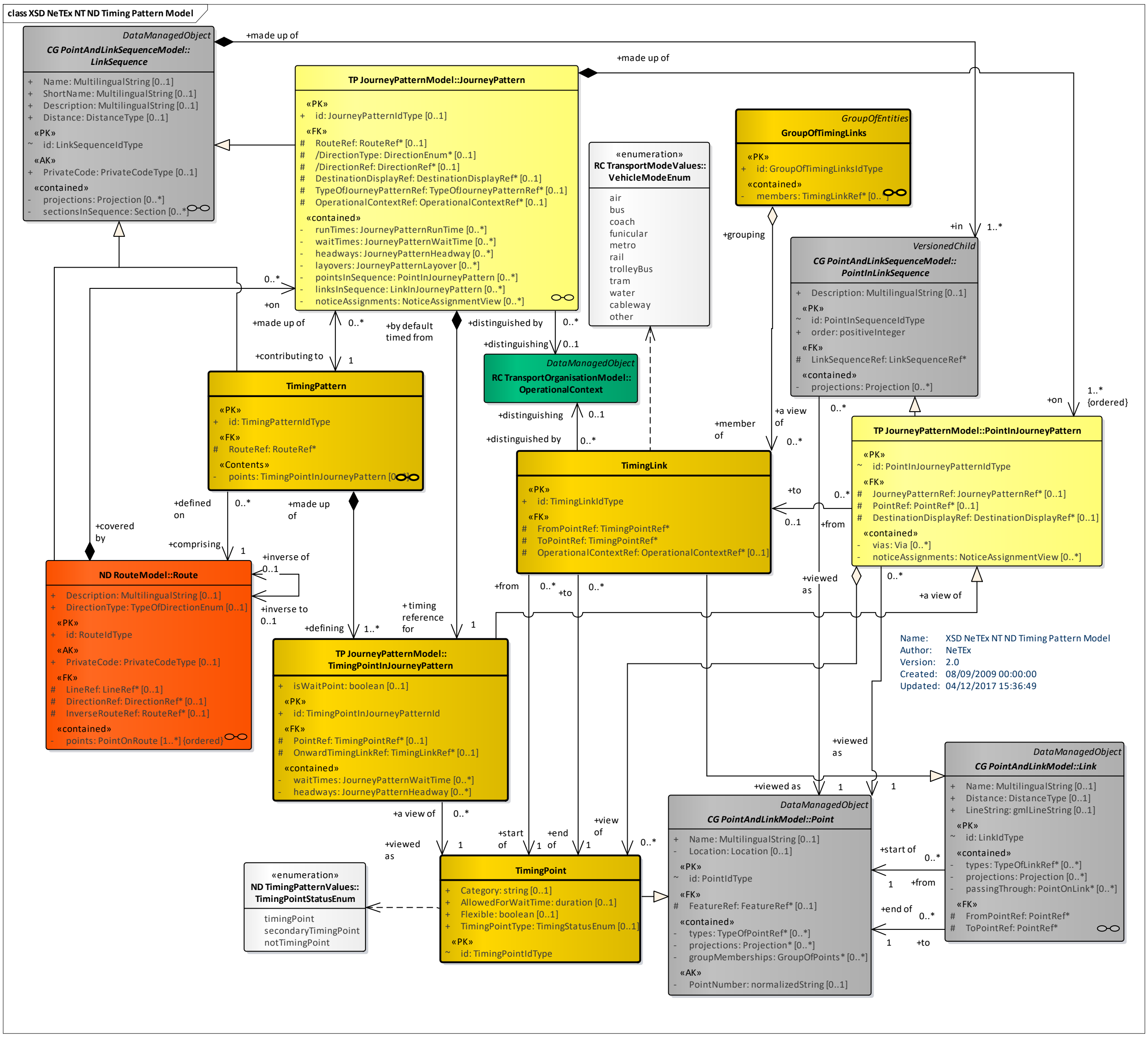


Figure 463 – Timing Pattern – Physical Model (UML)

8.3.6.4 Timing Pattern – Attributes and XSD

8.3.6.4.1 TimingPoint – Model Element

A POINT against which the timing information necessary to build schedules may be recorded.

Table 386 – *TimingPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TimingPoint*	::>	TIMING POINT inherits from POINT.
«PK»	***id***	*TimingPointIdType*	1:1	Identifier of a TIMING POINT.

«enum»	***TimingPointStatus***	*TimingPointStatusEnum*	0:1	Timing compliance status of POINT: See allowed values below.
	AllowedForWait-Time	*xsd:duration*	0:1	Default Wait time to use for TIMING PATTERNs that use TIMING POINT. TIME DEMAND TYPE specific values may be specified using a TIME DEMAND TYPE TIMING.

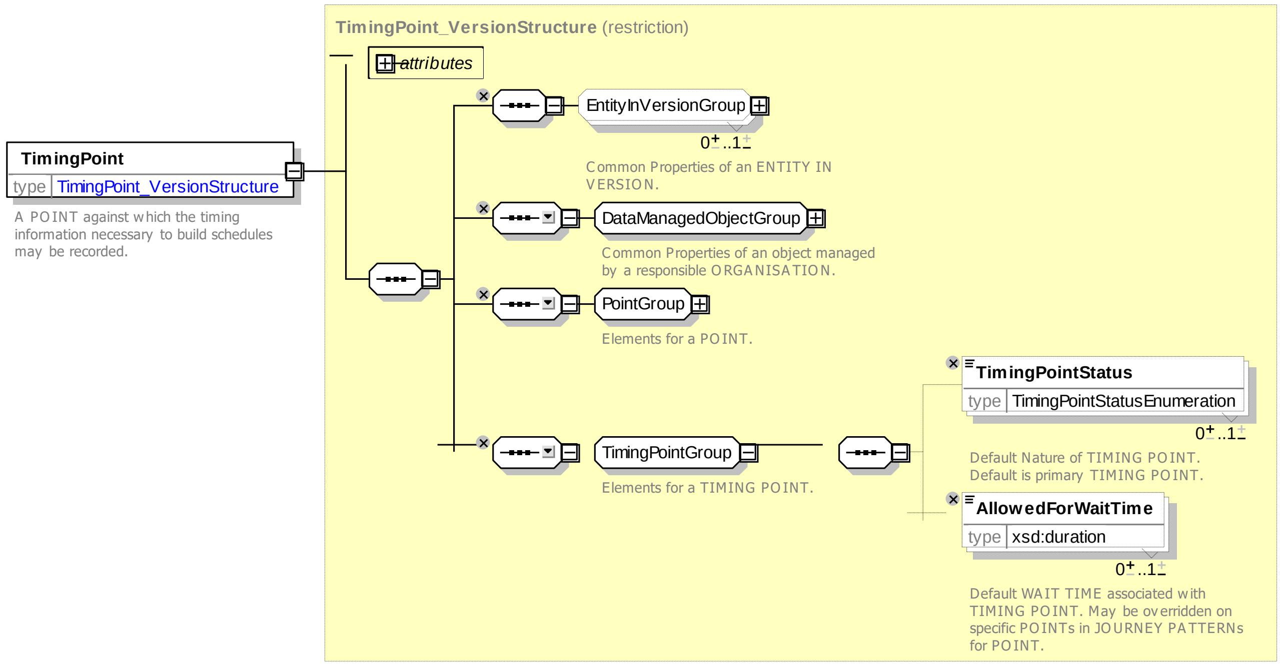


Figure 464 – *TimingPoint – XSD*

8.3.6.4.1.1 **TimingPointStatus – Allowed values**

The following table shows the allowed values for ***TimingPointStatus (****TimingStatusEnum*eration***).***

Table 387 – *TimingPointType* – Allowed values

Value	Description
timingPoint	Primary timing point
secondaryTimingPoint	Secondary
notTimingPoint	Not a t

8.3.6.4.2 TimingLink – Model Element

An ordered pair of TIMING POINTs for which run times may be recorded.

Table 388 – *TimingLink* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Link*	*::>*	TIMING LINK inherits from LINK.

«PK»	***id***	*TimingLinkIdType*	1:1	Identifier of a TIMING LINK.
«enum»	***VehicleMode***	*VehicleModeEnum*	0:1	Transport MODE of a TIMING LINK. See allowed values for TRANSPORT MODE.
«FK»	***FromPointRef***	*TimingPointRef+*	1:1	Reference to TIMING POINT at which TIMING LINK starts.
«FK»	***ToPointRef***	*TimingPointRef+*	1:1	Reference to TIMING POINT at which TIMING LINK ends.
«FK»	***Operational-ContextRef***	*OperationalContextRef*	0:1	Reference to OPERATIONAL CONTEXT used to distinguish LINK from similar links.

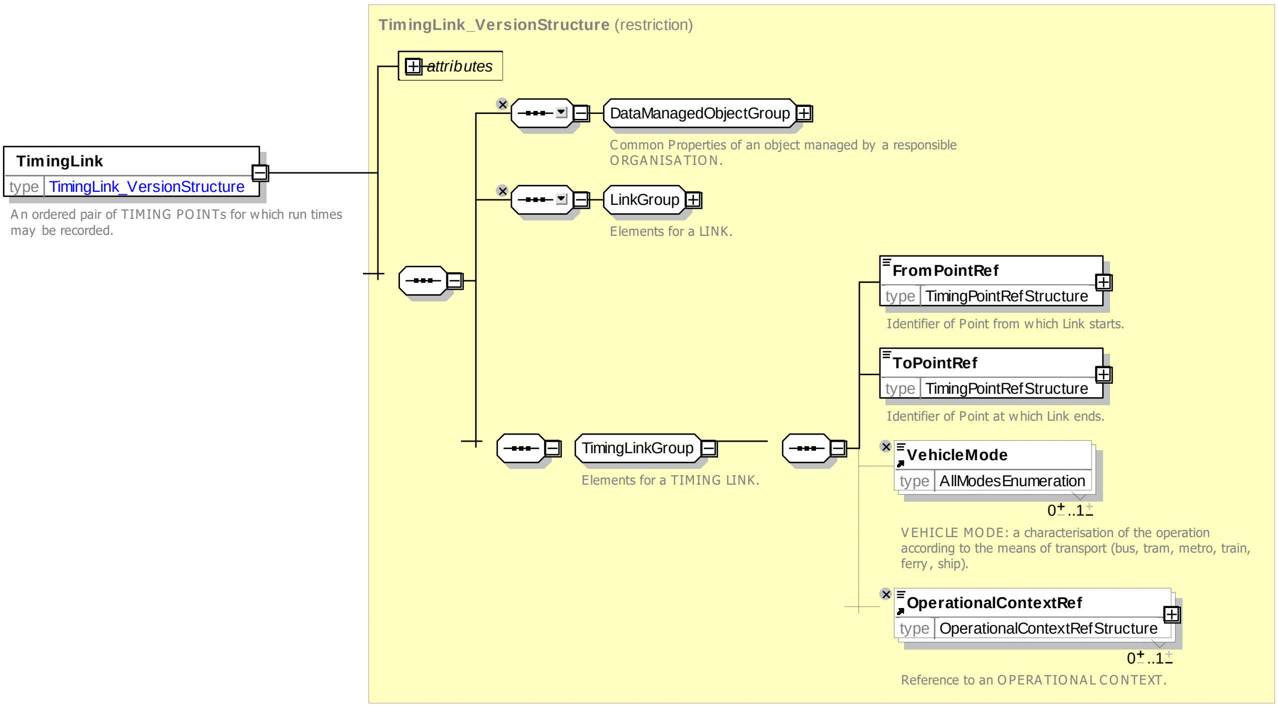


Figure 465 – *TimingLink – XSD*

8.3.6.4.3 TimingPattern – Model Element

The subset of a JOURNEY PATTERN made up only of TIMING POINTs IN JOURNEY PATTERN.

Table 389 – *TimingPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*LinkSequence*</u>	::>	TIMING PATTERN inherits from LINK SEQUENCE.
«PK»	***id***	*TimingPatternIdType*	1:1	Identifier of TIMING PATTERN.
«FK»	***RouteRef***	*RouteRef*	1:1	ROUTE that TIMING PATTERN follows.

«FK»	***DirectionRef***	*DirectionRef*	0:1	A Direction of a ROUTE. One of a restricted set of values. The default is '*outbound*'.
		choice	0:1	Time demand at start of pattern.
«FK»	*a* ***TimeDemandTypeRef***	*TimeDemandTypeRef*	0:1	Reference to a TIME DEMAND TYPE. If given by context need not to be stated.
«FK»	*b* ***TimeBandRef***	*TimeBandRef*	0:1	Reference to a TIME BAND.
«cntd»	***pointsInSequence***	*TimingPointInJourney-Pattern*	0:*	TIMING POINTs in TIMING PATTERN.
«cntd»	***points***	*TimingPoint*	0:*	List of points used in TIMING PATTERN. May also be defined elsewhere. Can be used to encapsulate TIMING PATTERN with its component POINTS.
«cntd»	***Links***	*TimingLink*	0:*	List of links used in TIMING PATTERN. May also be defined elsewhere. Can be used to encapsulate TIMING PATTERN with its component Links.

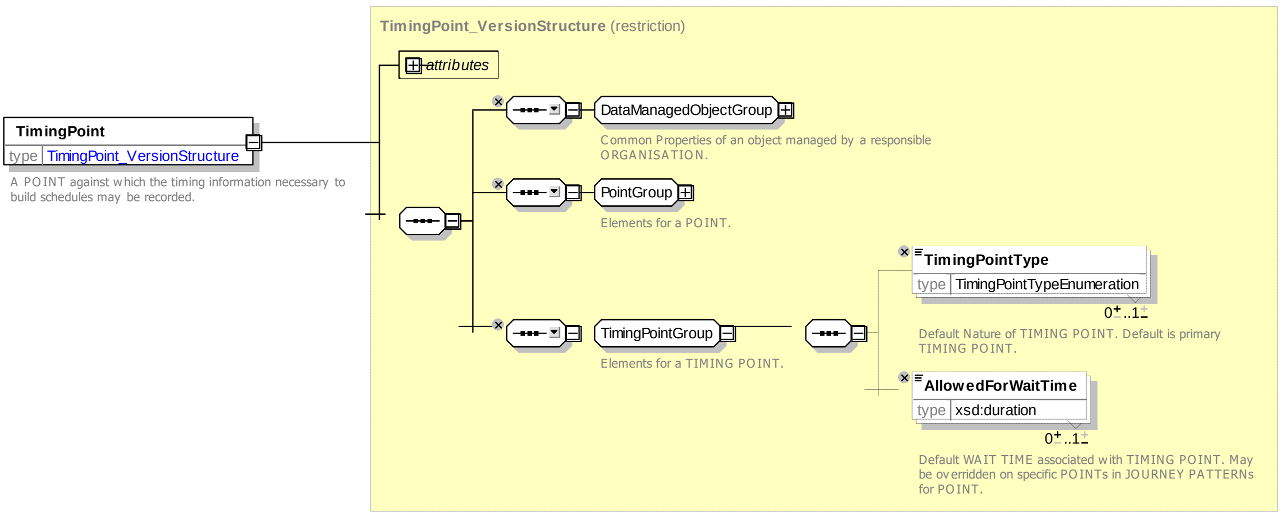


Figure 466 – *TimingPattern – XSD*

8.3.6.4.4 GroupOfTimingLinks – Model Element

A set of TIMING LINKs grouped together according to the similarity of TIME BANDs which are relevant to them. There may be a GROUP OF TIMING LINKs which covers all TIMING LINKs, for use when different GROUPs OF TIMING LINKs are not needed.

Table 390 – *GroupOfTimingLinks* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	GROUP OF TIMING LINKs inherits from GROUP OF ENTITIES.
«PK»	***id***	*GroupOfTimingLinksIdType*	1:1	Identifier of GROUP OF TIMING LINKs.

«cntd»	***members***	*TimingLinkRef++*	0:*	Reference to TIMING LINKS that are members of GROUP OF TIMING LINKs.

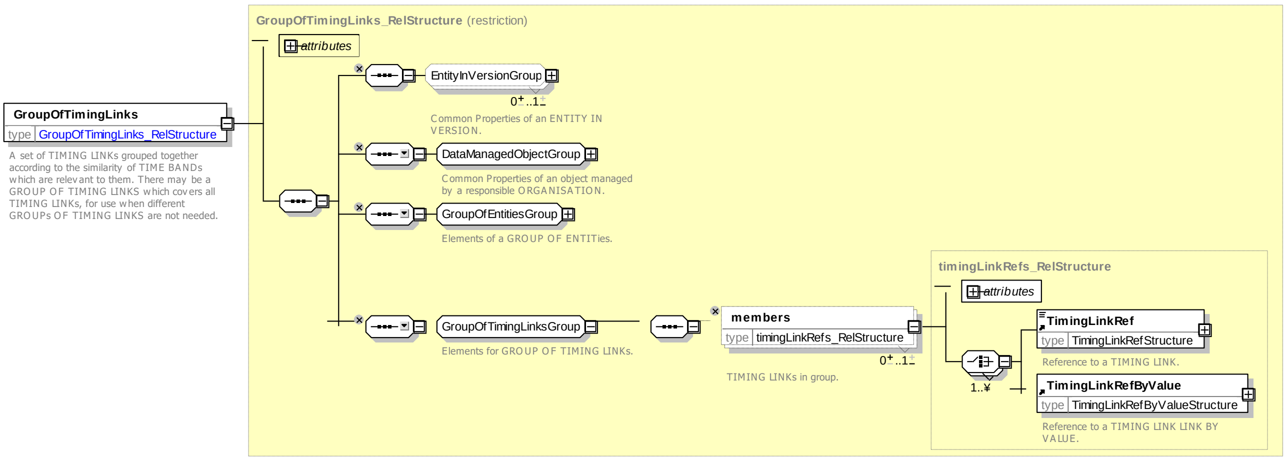


Figure 467 – *GroupOfTimingLinks – XSD*

8.3.6.5 XML Examples of Timing Pattern Elements

The following XML fragments show the definition of a TIMING PATTERN between three SCHEDULED STOP POINTs (A B and C) that also has three additional intermediate TIMING POINTs which are not SCHEDULED STOP POINTs. Thus the route is [A], [a_t1], [a_t2], [B], [b-t1], [C].

EXAMPLE

```
<!--- =========STOP POINTs============ -->
    <scheduledStopPoints>
        <ScheduledStopPoint version="any" id="SSP_001">
            <Name>Alpha &amp; Castle</Name>
            <TimingPointType>timingPoint</TimingPointType>
            <ShortName>Alpha</ShortName>
            <StopType>onstreetBus</StopType>
        </ScheduledStopPoint>
        <ScheduledStopPoint version="any" id="SSP_002">
            <Name>Bravo Street</Name>
            <TimingPointType>timingPoint</TimingPointType>
            <ShortName>Bravo</ShortName>
            <StopType>onstreetBus</StopType>
        </ScheduledStopPoint>
        <ScheduledStopPoint version="any" id="SSP_077">
            <Name>Charley Crescent</Name>
            <TimingPointType>timingPoint</TimingPointType>
            <ShortName>Charley</ShortName>
            <StopType>onstreetBus</StopType>
        </ScheduledStopPoint>
    </scheduledStopPoints>
    <!--- =========TIMING POINTs============ -->
    <timingPoints>
        <TimingPoint version="any" id="TimingPoint:SSP_001_t1">
            <Name>Between Alpha and Bravo Point 1</Name>
        </TimingPoint>
        <TimingPoint version="any" id="TimingPoint:SSP_001_t2">
            <Name>Between Alpha and Bravo Point 2</Name>
```

```xml
        </TimingPoint>
        <TimingPoint version="any" id="TimingPoint:SSP_002_t3">
            <Name>Between Bravo and Charley Point 1</Name>
        </TimingPoint>
    </timingPoints>
    <!--- =========TIMING LINKS=========== -->
    <timingLinks>
        <TimingLink version="any" id="TimingLink:SSP_001_to_SSP_077">
            <Name>Overall timing Alpha to Charley green</Name>
            <FromPointRef version="any" ref="SSP_001"/>
            <ToPointRef version="any" ref="SSP_077"/>
        </TimingLink>
        <TimingLink version="any" id="TimingLink:SSP_001_to_SSP_001_t1">
            <Name> After Alpha t1</Name>
            <FromPointRef version="any" ref="SSP_001"/>
            <ToPointRef version="any" ref="TimingPoint:SSP_001_t1"/>
        </TimingLink>
        <TimingLink version="any" id="TimingLink:SSP_001_t1_to_SSP_001_t2">
            <Name>After Alpha t1 to After Alpha t2</Name>
            <FromPointRef version="any" ref="TimingPoint:SSP_001_t1"/>
            <ToPointRef version="any" ref="TimingPoint:SSP_002_t3"/>
        </TimingLink>
        <TimingLink version="any" id="TimingLink:SSP_001_t2_to_SSP_002">
            <Name>After Alpha t2 to Bravo</Name>
            <FromPointRef version="any" ref="TimingPoint:SSP_001_t2"/>
            <ToPointRef version="any" ref="SSP_002"/>
        </TimingLink>
        <TimingLink version="any" id="TimingLink:SSP_002_to_SSP_002_t3">
            <Name>Bravo to After Bravo t1</Name>
            <FromPointRef version="any" ref="SSP_002"/>
            <ToPointRef version="any" ref="TimingPoint:SSP_002_t3"/>
        </TimingLink>
        <TimingLink version="any" id="TimingLink:SSP_002_t3_to_SSP_077">
            <Name>After Bravo t1 to Charley</Name>
            <FromPointRef version="any" ref="TimingPoint:SSP_002_t3"/>
            <ToPointRef version="any" ref="SSP_077"/>
        </TimingLink>
    </timingLinks>
    <!--- =========TIMING PATTERNs============ -->
    <timingPatterns>
        <TimingPattern version="any" id="hde:TimingPattern:tp_24o">
            <Name>Route 24</Name>
            <RouteRef version="any" ref="Route:RT_24o"/>
            <DirectionType>outbound</DirectionType>
            <pointsInSequence>
                <TimingPointInJourneyPattern                 version="any"                 order="1"
id="hde:TimingPointInJourneyPattern:tpijp_24o_01">
                    <TimingPointRef version="any" ref="SSP_001"/>
                    <OnwardTimingLinkRef version="any" ref="TimingLink:SSP_001_to_SSP_001_t1"/>
                </TimingPointInJourneyPattern>
                <TimingPointInJourneyPattern                 version="any"                 order="2"
id="hde:TimingPointInJourneyPattern:tpijp_24o_02">
                    <TimingPointRef version="any" ref="TimingPoint:SSP_001_t1"/>
                    <OnwardTimingLinkRef version="any" ref="TimingLink:SSP_001_t1_to_SSP_001_t2"/>
                </TimingPointInJourneyPattern>
                <TimingPointInJourneyPattern                 version="any"                 order="3"
id="hde:TimingPointInJourneyPattern:tpijp_24o_03">
                    <TimingPointRef version="any" ref="TimingPoint:SSP_001_t2"/>
                    <OnwardTimingLinkRef version="any" ref="TimingLink:SSP_001_t2_to_SSP_002"/>
```

```
                </TimingPointInJourneyPattern>
                <TimingPointInJourneyPattern version="any" order="4"
id="hde:TimingPointInJourneyPattern:tpijp_24o_04">
                    <TimingPointRef version="any" ref="SSP_002"/>
                    <OnwardTimingLinkRef version="any" ref="TimingLink:SSP_002_to_SSP_002_t3"/>
                </TimingPointInJourneyPattern>
                <TimingPointInJourneyPattern version="any" order="5"
id="hde:TimingPointInJourneyPattern:tpijp_24o_05">
                    <TimingPointRef version="any" ref="TimingPoint:SSP_002_t3"/>
                    <OnwardTimingLinkRef version="any" ref="TimingLink:SSP_002_t3_to_SSP_077"/>
                </TimingPointInJourneyPattern>
                <TimingPointInJourneyPattern version="any" order="6"
id="hde:TimingPointInJourneyPattern:tpijp_24o_06">
                    <TimingPointRef version="any" ref="SSP_077"/>
                </TimingPointInJourneyPattern>
            </pointsInSequence>
        </TimingPattern>
    </timingPatterns>
```

8.3.7 Flexible Network

NeTEx is designed to support FTS (Flexible Transport Service) and DRT (Demand Responsive Transport). DRT and FTS often cover similar services; FTS being more generic since flexibility may not be directly linked to the demand, but may be related to some operating needs or cost optimisations. The term ‘FTS’ will be used in the following text to cover both concepts.

8.3.7.1 Flexible Network Introduction

NeTEx does not have a separate FTS specific model, but has extra properties that can be used to describe FTS systems.

For Network Topology, the main FTS aspect considered is the FTS line structure. Other aspects of FTS such as reservation rules and means will be managed in NeTEx Part 3 "Fare Information Exchange".

Different types of FTS are considered in the present document. The FTS type considered are defined on JOURNEY PATTERN level (or POINT IN JOURNEY PATTERN, in the case when only a part of the JOURNEY PATTERN is flexible) or on ROUTE level. This allows for:

— Virtual line service

— Flexible service with main route

— Corridor service (flexible service without main route)

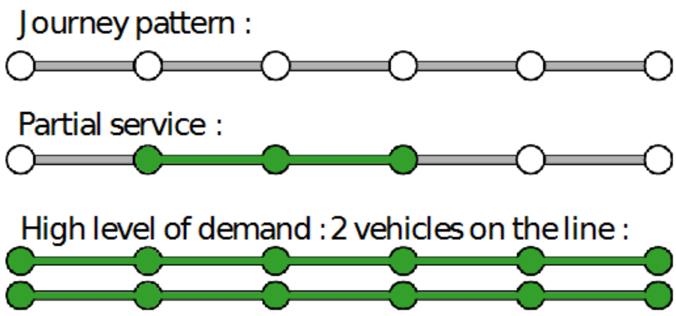
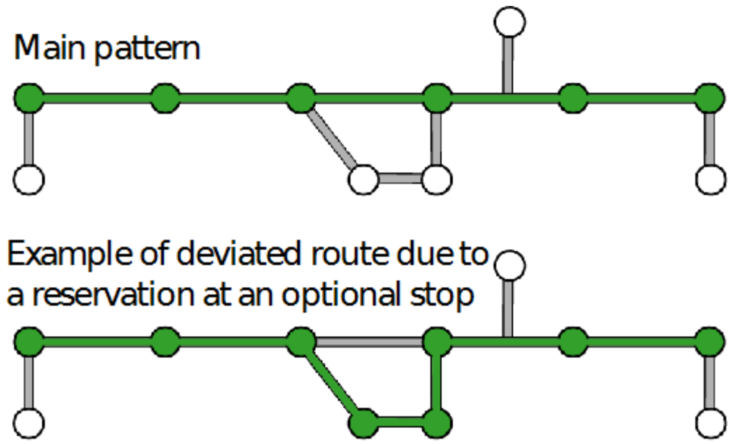
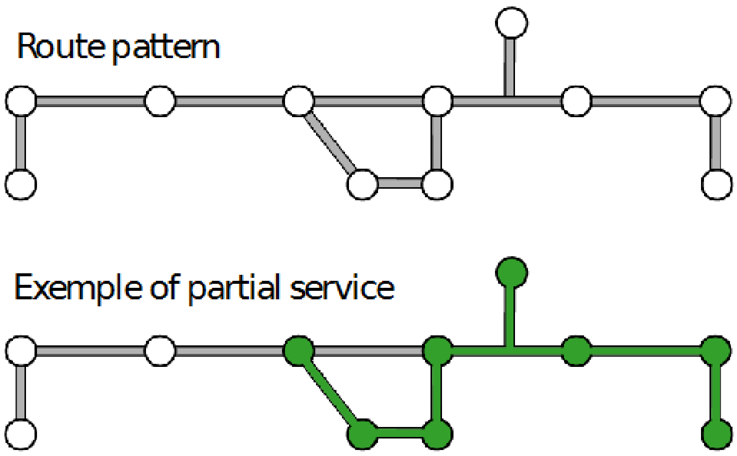
— Fixed stop area-wide flexible service

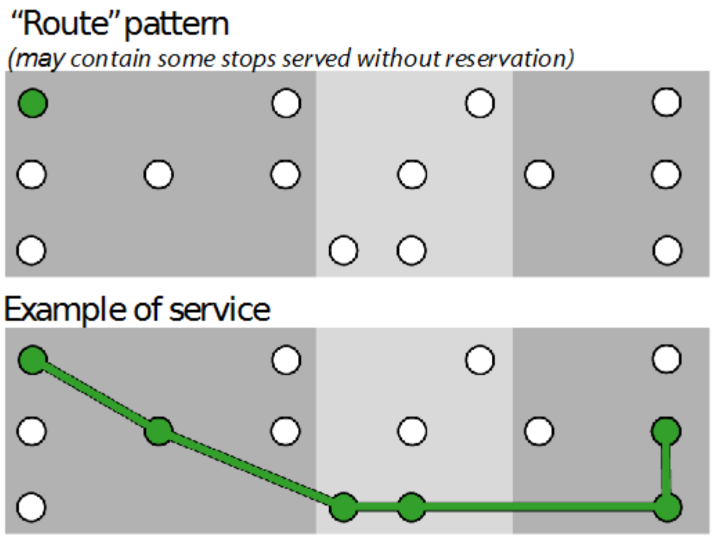
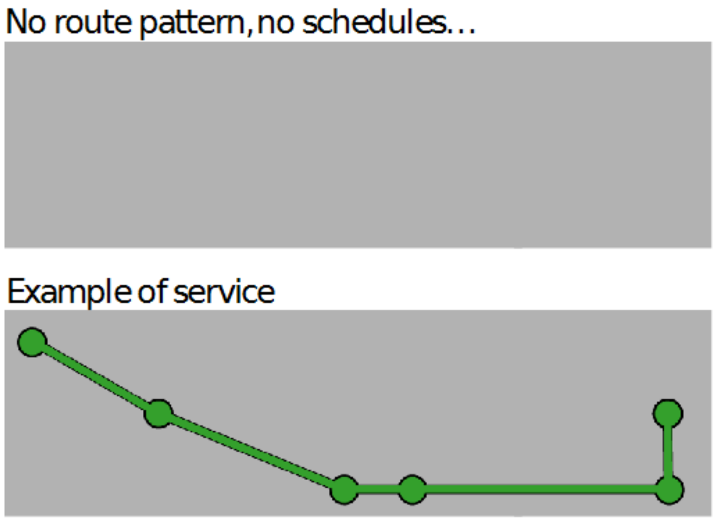
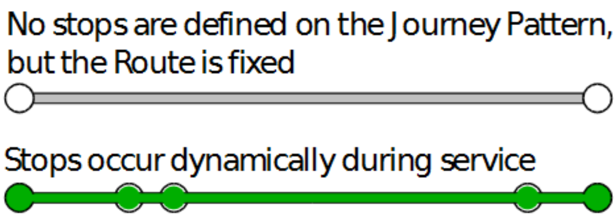
— Free area-wide flexible service

— Mixed types of flexible service (not at POINT level)

The following table summarize the FTS LINE topologies taken into account in NeTEx.

Table 391 – FTS typology

Name	Description	Figure
Virtual Line	This case is very similar to fixed line operation: journey patterns are defined as usual, but stops are served only if there is a passenger booking for it. Several vehicles may be allocated to the same journey when high level of demand occurs. Virtual line can be operated with fixed or dynamic passing times.	 *Figure 468*
Flexible line with main route	A minimal list and order of stops are defined determining a “main and minimal” journey pattern. Possible additional stops are defined but will be served only in case of passenger reservation. A range of journey patterns is determined through a stop list and order defined dynamically according to the passenger reservations and "around" the “main and minimal” journey pattern.	 *Figure 469*
Flexible line without main route	The possible stops of the journey pattern are known, and the possible stop sequences are also defined and the real stop list and order are defined dynamically according to the passenger reservations without any reference to a main pattern.	 *Figure 470*

Flexible zone with fixed stops	The service is defined by one or several zones (in sequence). Each zone is defined by a set of possible stops. Stops served, and stop order are defined for each vehicle journey according to the reservations. Passing times (entry and exit time) are usually defined for each zone. They may also be defined for each stop.	 *Figure 471*
Flexible zone without fixed stops	The service is defined by one or several zones (in sequence). A stop can occur anywhere in each Zone. Stops served, and stop order will be defined four each vehicle journey according to the reservations. Passing times may be defined for each zone (entry and exit time), or for each stop.	 *Figure 472*
Hail & Ride	The route is defined, but the Journey Pattern only has a start and an end. Boarding or alighting is obtained by signalling the driver that one wishes to board/alight, and can occur anywhere along the Route. It also sometimes happens that boarding occurs on fixed stop, and only alighting can occur on demand anywhere along the Route.	 *Figure 473*
Combination of any of the previous FTS structure	A lot of FTS services are defined as a sequence of the previously described FTS types.	

The following table summarizes the FTS reservation issues taken into account in NeTEx.

Table 392 – FTS Reservation

Reservation type and constraint
The passenger must/can/cannot make a reservation.
The reservation must be done x minutes/days/… before his/vehicle departure time.
The reservation may be done x minutes/days/… before departure.
The passenger mays open at most x reservations at the same time.
At most x seats may be booked in one reservation.
A reservation may be refused/updated/cancelled/… by the passenger/operator until x minutes before departure.

It should be noted that most of the reservation aspects above are not FTS specific and may also be useful for regular transport.

8.3.7.2 FLEXIBLE NETWORK – Conceptual MODEL

The following UML diagram shows that there is no major difference between FTS and fixed route points, links and zones. The main difference resides in the way the model and the typology are used.

— FTS often needs to refer to a ZONE of operation instead of specific SCHEDULED STOP POINT. As shown by the diagram, a Point may be represented by the centroid of a ZONE (i.e. not be the actual centre nor any kind of barycentre, but only a point used to refer to the ZONE). Therefore NeTEx uses the convention that if the POINT (a SCHEDULED STOP POINT, a TIMING POINT or a ROUTE POINT) is the centroid of a ZONE, this means that this POINT (SCHEDULED STOP, TIMING or ROUTE POINT) is representing a ZONE, to be used for FTS as a flexible ZONE. In order to specify the nature of the flexibility, the additional FLEXIBLE POINT PROPERTIES element is available.

— The ZONE itself may contain a set of POINTs through the 'including' relation (it is then possible to define all the SCHEDULED STOP POINTs of a flexible zone with fixed stops, for example). A ZONE may include other ZONEs. According to the typology of FTS chosen, a constraint is formulated requiring that a ZONE may contain POINTs or ZONEs, but not both.

— The FLEXIBLE LINK PROPERTIES are available for LINKs, making it possible to describe hail & ride LINKs or a LINE structure combining different kind of FTS. The LINK with FLEXIBLE LINK PROPERTIES can have some generic VALIDITY CONDITIONs in order to be able to describe situations likes LINKs being hail & ride only between 9 pm and 7 am.

Note that in order to make coverage by flexible services visible to journey planners, FLEXIBLE STOP PLACEs, FLEXIBLE AREAs and HAIL AND RIDE AREAs can be defined. See Section 8.4.5 on FLEXIBLE STOP PLACE model.

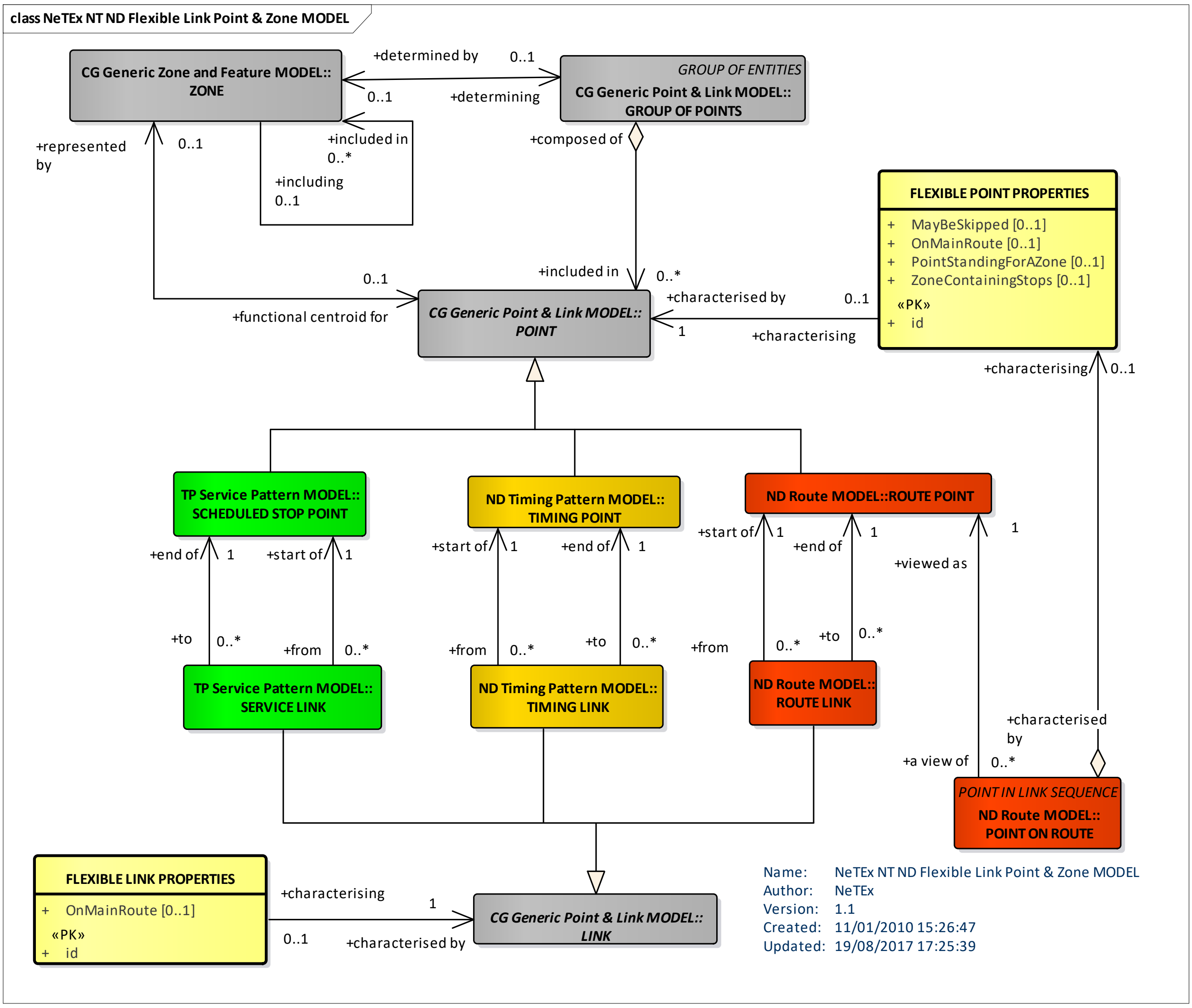


Figure 474 – FLEXIBLE NETWORK (FTS ZONE, POINT & LINK) – Conceptual MODEL (UML)

8.3.7.2.1 Flexible Routes

A FLEXIBLE ROUTE is a specialisation of ROUTE allowing flexible behaviour, the type of flexibility being described by its attribute.

As shown above, ROUTE POINTs and ROUTE LINKs may have flexible properties, as may POINTs on ROUTE modelling in order to describe FTS (FLEXIBLE POINT and LINK PROPERTIES have already been described with the previous schema).

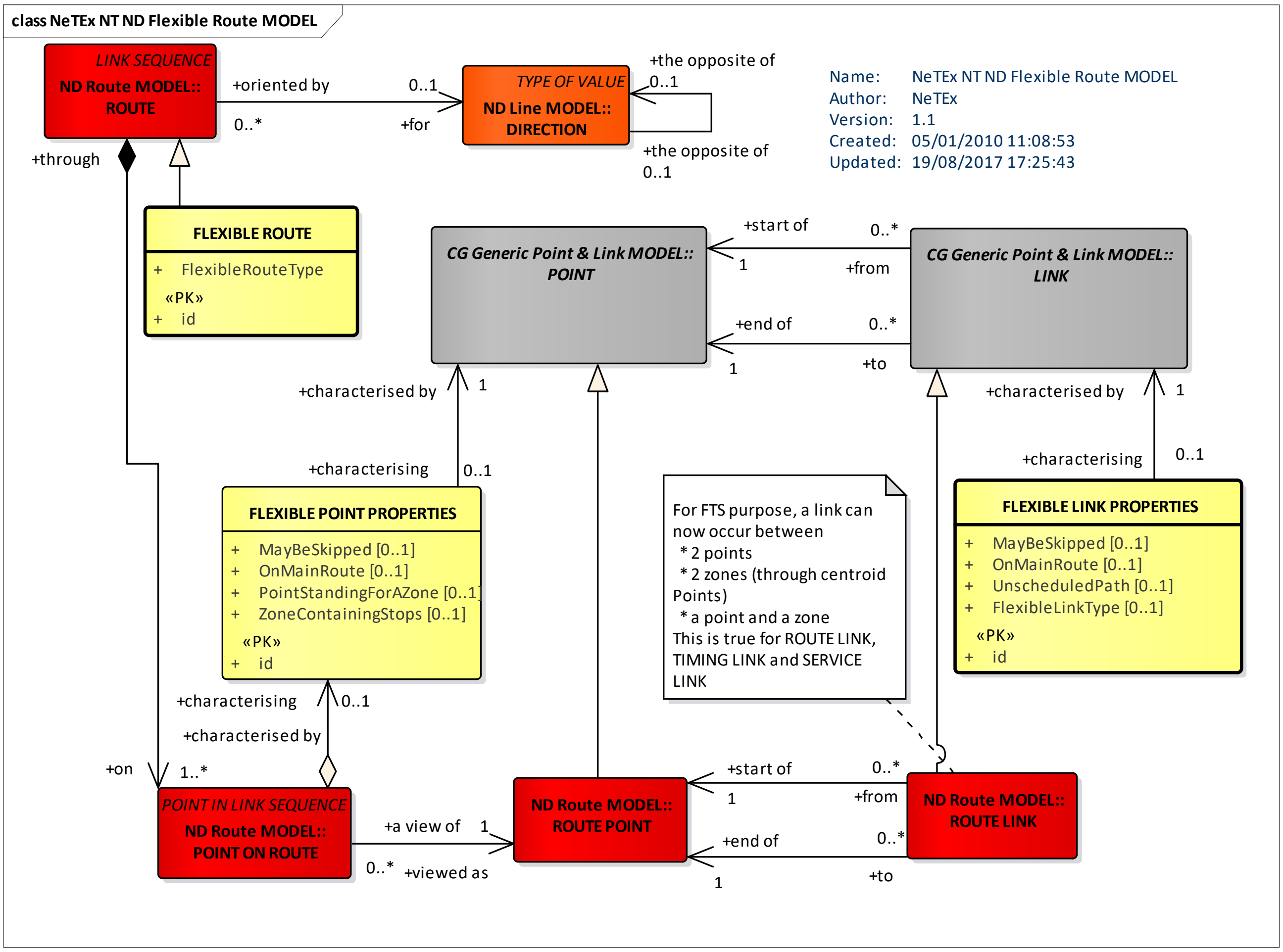


Figure 475 – Flexible ROUTE – Conceptual MODEL (UML)

8.3.7.2.2 Flexible Lines

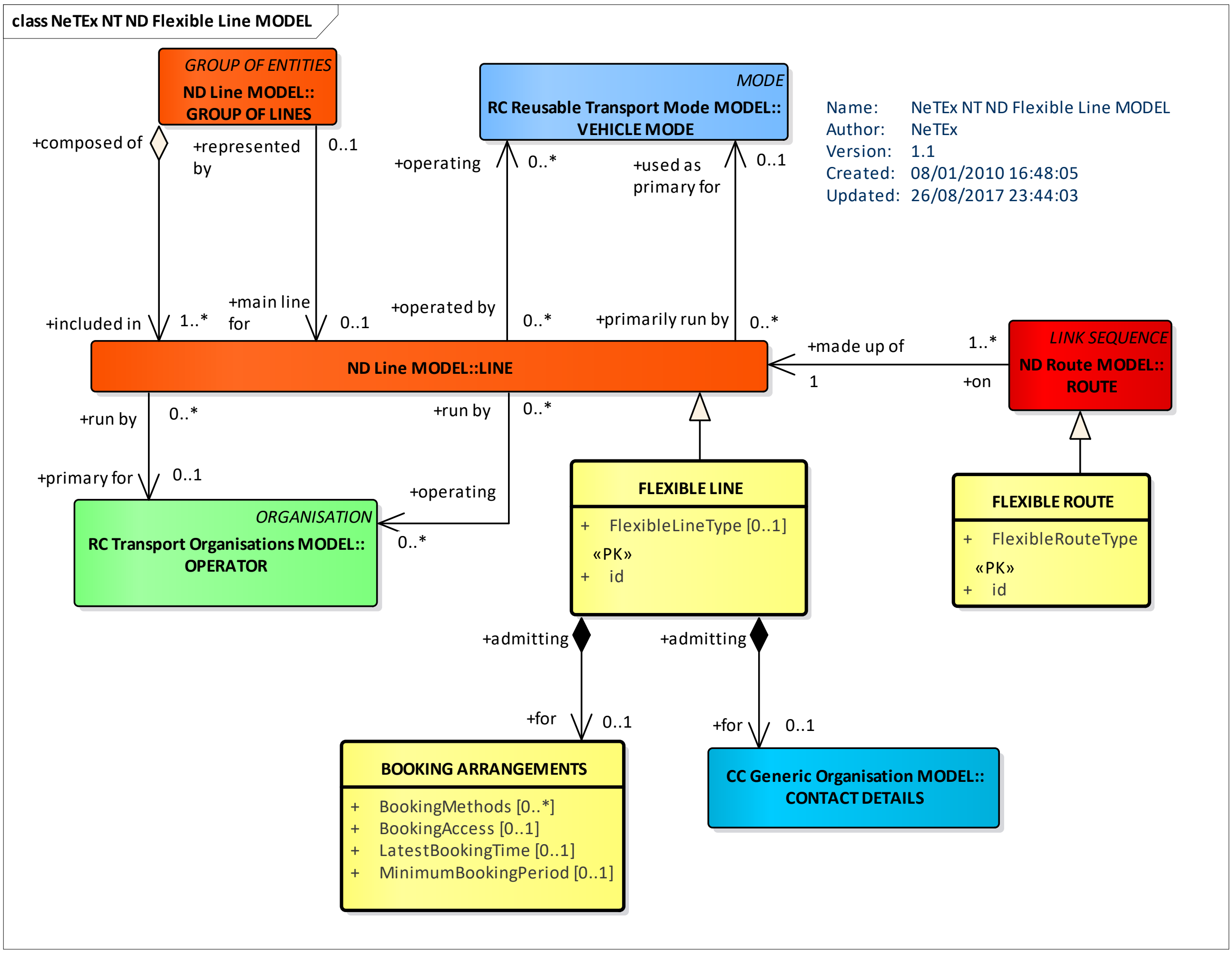


Figure 476 – Flexible LINE – Conceptual MODEL (UML)

8.3.7.3 Flexible Network – Physical Model

8.3.7.3.1 Flexible Network – Physical Model – Points, Links & Zones

The following figure shows the physical representation of the POINTs and LINKs of a Flexible Network model. Additional Properties can be associated with the POINTs and LINKs of a FLEXIBLE ROUTE

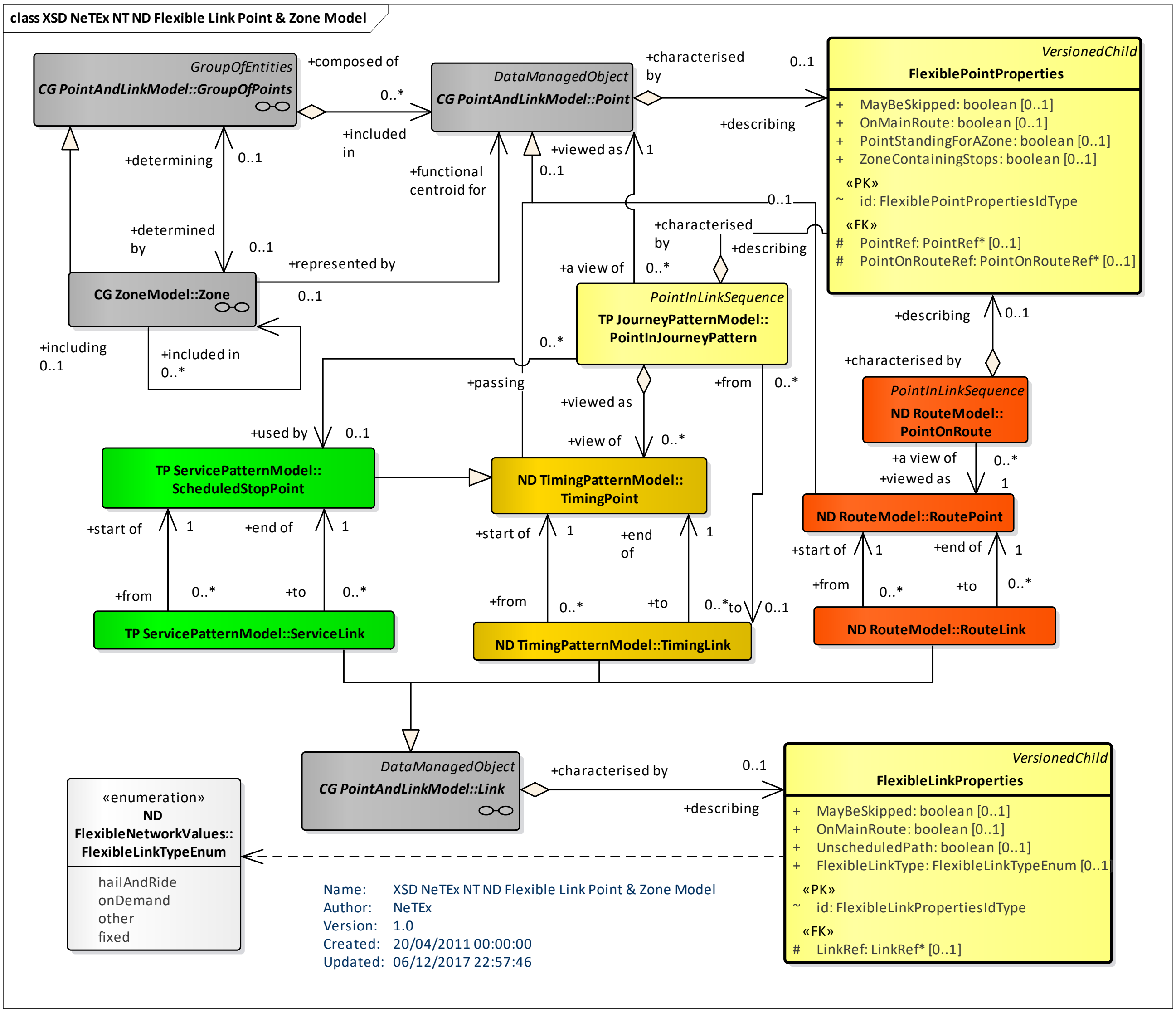


Figure 477 – Flexible Zone, Point & Link – Physical model (UML)

8.3.7.3.2 Flexible Network – Physical Model – Lines & Routes

The following figure shows the physical representation of the FLEXIBLE LINES & FLEXIBLE ROUTEs. Booking arrangements can be associated with the FLEXIBLE LINEs. In order for the flexible services to appear in a journey planner, the flexible stops can be declared as FLEXIBLE STOP PLACEs.

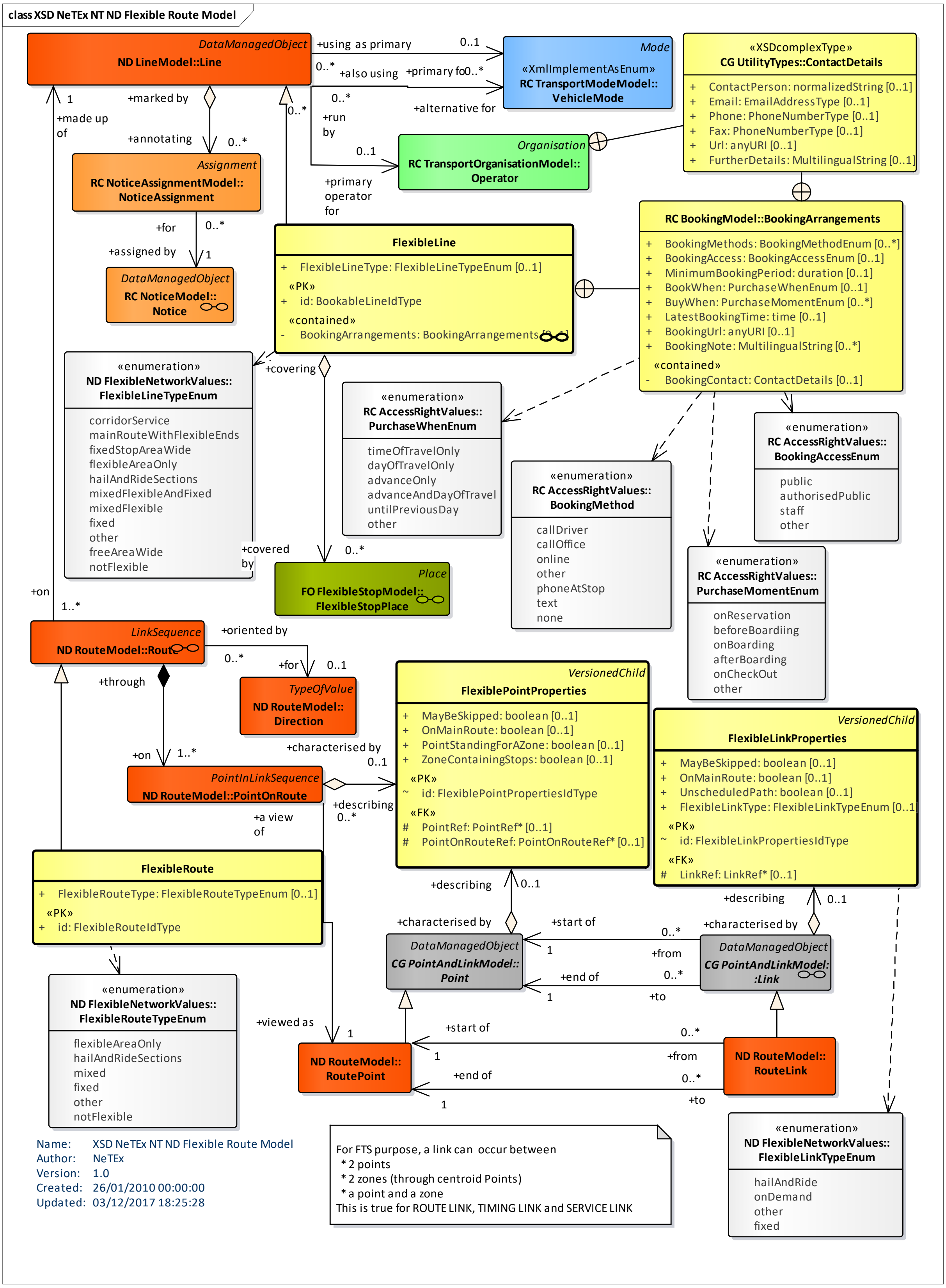


Figure 478 – Flexible Line & Route – Physical model (UML)

8.3.7.4 Flexible Network – Attributes and XSD

8.3.7.4.1 FlexibleRoute – Model Element

Specialisation of a ROUTE for a flexible service. May include both point ant zonal areas and ordered and unordered sections.

Table 393 – *FlexibleRoute* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*DataManagedObject*</u>	::>	FLEXIBLE ROUTE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*FlexibleRouteIdType*	1:1	Identifier of FLEXIBLE ROUTE.
«enum»	***FlexibleRoute-Type***	*FlexibleRouteTypeEnum*	1:1	Type of FLEXIBLE ROUTE. See allowed values below.

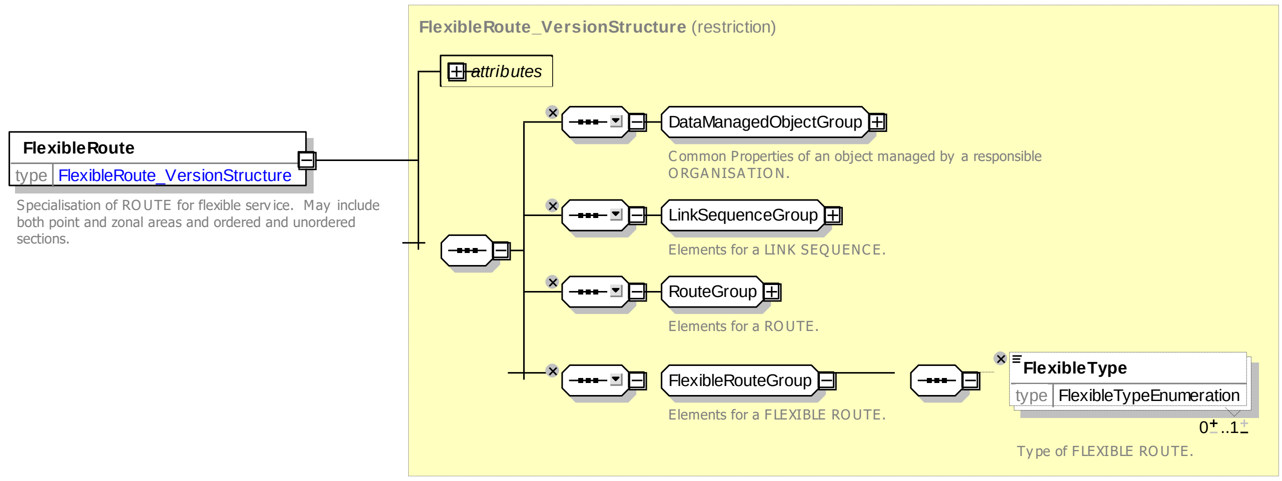


Figure 479 – *FlexibleRoute – XSD*

8.3.7.4.1.1 FlexibleType – Allowed values

Allowed values for ***FlexibleRouteType*** (*FlexibleTypeEnumeration*).

Table 394 – *FlexibleType* – Allowed Values

Value	Description
flexibleAreaOnly	Flexible Route has only flexible areas.
hailAndRideSections	Flexible Route has hail and ride sections.
mixed	Flexible Route has both fixed and flexible areas.
fixed	Route has fixed sections only.
other	Flexible Route has other characteristics.

8.3.7.4.2 FlexibleLine – Model Element

A group of FLEXIBLE ROUTEs which is generally known to the public by a similar name or number and which have common booking arrangements.

Table 395 – *FlexibleLine* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Line*</u>	::>	FLEXIBLE LINE inherits from LINE
«PK»	***id***	*FlexibleLineIdType*	1:1	Identifier of FLEXIBLE LINE.
«enum»	***FlexibleLineType***	*FlexibleLineTypeEnum*	0:1	Type of FLEXIBLE LINE. See allowed values below.
«cntd»	***Booking-Arrangements*Group**	***xmlGroup***	0:1	Booking arrangements for FLEXIBLE LINE. See Part 1 RC ervice Retsrictions Model.

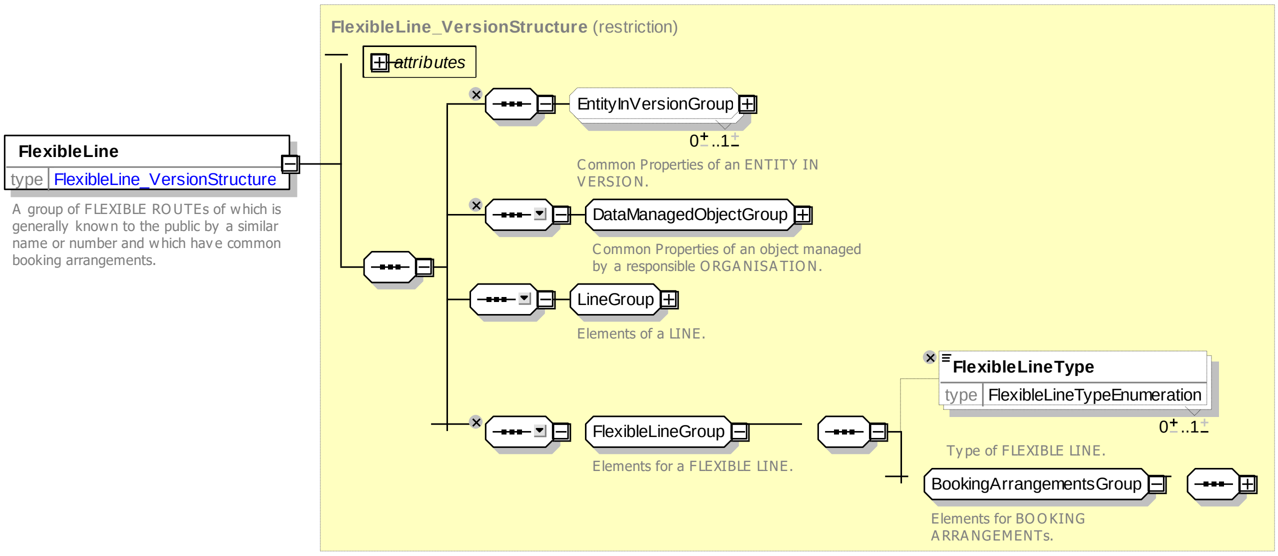


Figure 480 – *FlexibleLine – XSD*

8.3.7.4.3 FlexiblePointProperties – Model Element

The set of characteristics describing the possible flexibility of a POINT.

A composition is used with POINT in order to avoid multiple inheritance and a type explosion.

Table 396 – *FlexiblePointProperties* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	FLEXIBLE POINT PROPERTIES inherits from VERSIONED CHILD.
«PK»	***id***	*FlexiblePointProperties-IdType*	1:1	Identifier of a FLEXIBLE POINT PROPERTIES.
		CHOICE	1:1	Element for which these are the flexible properties.

«FK»	*a*	***PointOnRoute-Ref***	*PointOnRouteRef*	0:1	POINT ON ROUTE for which these are the properties.
«FK»	*b*	***PointRef***	*(PointRef)*	0:1	POINT for which these are the properties.
		MayBeSkipped	*xsd:boolean*	0:1	Whether the POINT may be skipped.
		OnMainRoute	*xsd:boolean*	0:1	Whether the POINT is on the main ROUTE.
		PointStandingForAZone	*xsd:boolean*	0:1	Whether the POINT represents a FLEXIBLE ZONE.
		ZoneContaining-Stops	*xsd:boolean*	0:1	Whether the ZONE is defined by a GROUP of STOP POINTs (value ='*true*') or is a geographical zone defined by its boundary.

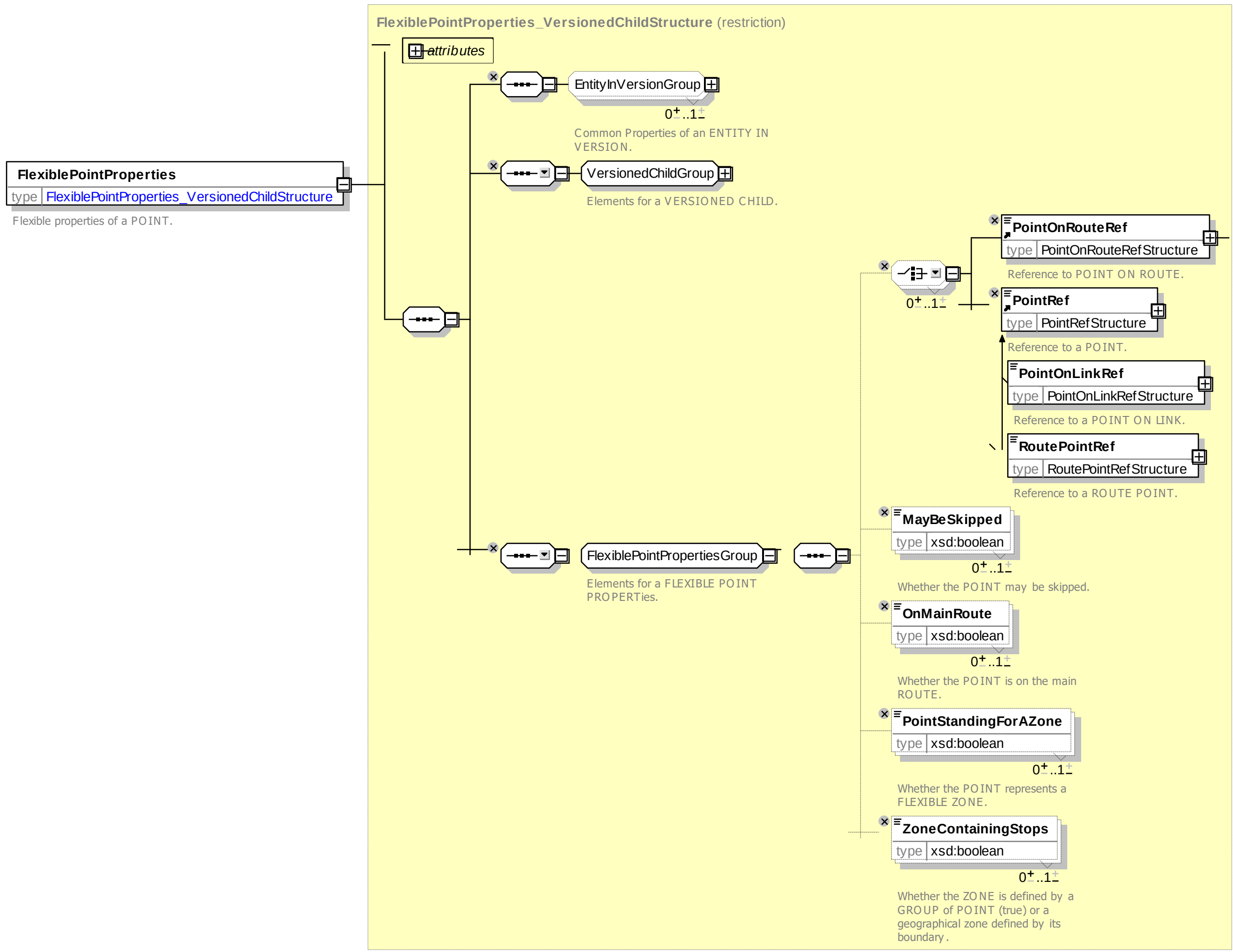


Figure 481 – *FlexiblePointProperties* – *XSD*

8.3.7.4.4 *FlexibleLinkProperties* – Model Element

Set of properties describing the flexible characteristics of a LINK.

A composition is used with LINK in order to avoid multiple inheritance and a type explosion of link subtypes.

Table 397 – *FlexibleLinkProperties* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	FLEXIBLE LINK PROPERTIES inherits from VERSIONED CHILD.
«PK»	***id***	*FlexibleLinkPropertiesIdType*	1:1	Identifier of a FLEXIBLE LINK PROPERTies.
«FK»	***LineRef***	*LineRef*	0:1	LINE for which these are the properties.
	MayBeSkipped	*xsd:boolean*	0:1	Whether the LINK may be skipped.
	OnMainRoute	*xsd:boolean*	0:1	Whether the LINK is on the main ROUTE of the LINE.
	UnscheduledPath	*xsd:boolean*	0:1	Whether this link is on an unscheduled path route.
«enum»	***FlexibleLinkType***	*FlexibleLinkTypeEnum*	0:1	Type of FLEXIBLE ROUTE LINK. See allowed values below.

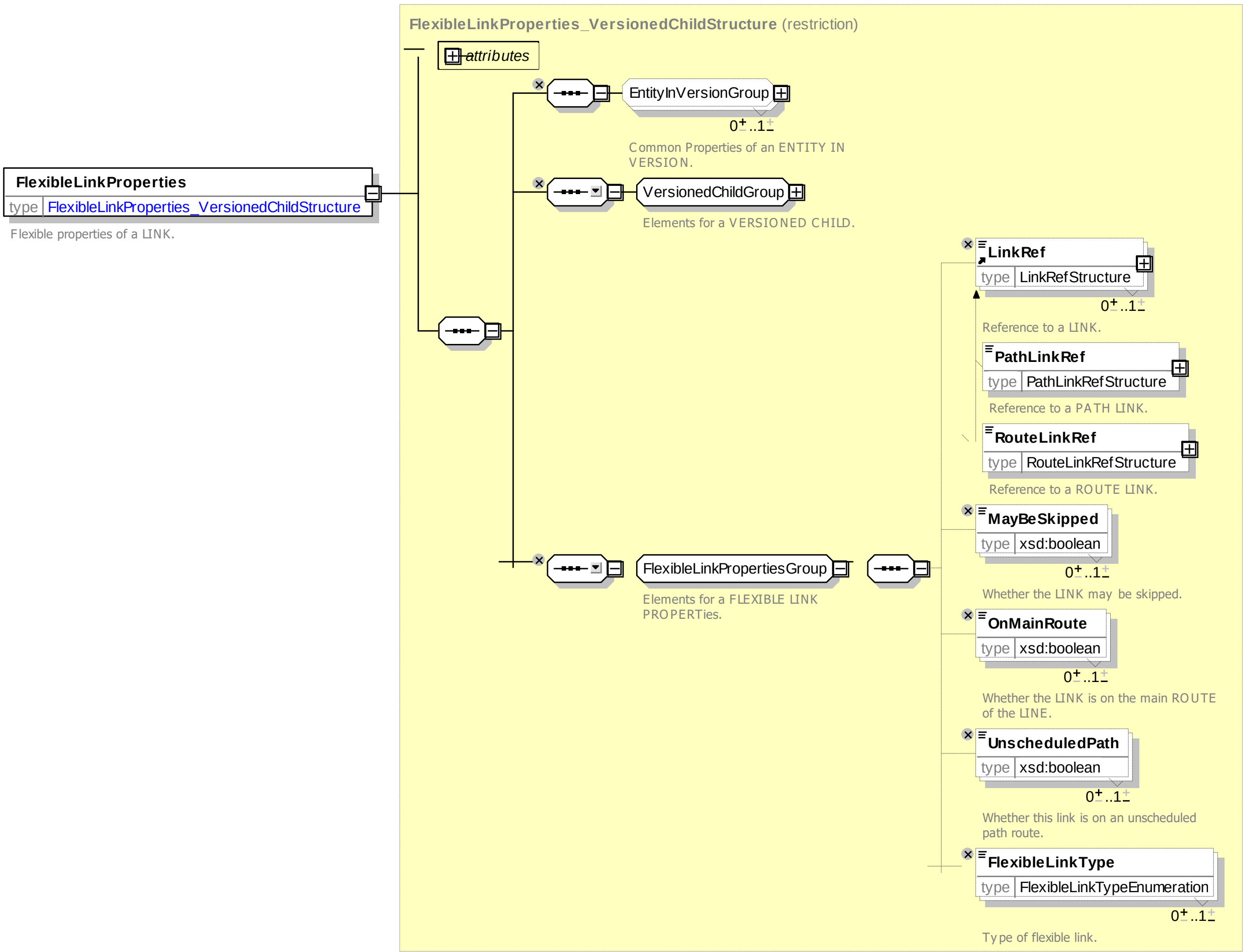


Figure 482 – *FlexibleLinkProperties* – XSD

8.3.7.4.4.1 FlexibleLinkType – Allowed values

Allowed values for FLEXIBLE LINK Type (*FlexibleLinkTypeEnumeration*).

Table 398 – *FlexibleLinkType* – Allowed Values

Value	Description
hailAndRide	Hail and Ride section.
onDemand	On demand section.
fixed	Fixed section.
other	Other type of flexible link.

8.4 Fixed Objects – Subsystem

The FIXED OBJECT model provides a representation of all the fixed context of travel by public transport, including:

- The stops and stations at which transport is accessed.
- The points of interest to which passengers are travelling.
- The detailed pathways between the various locations.
- The equipment found in them.
- The parking locations relative to both stops and points of interest.
- Accessibility data about all of the above.
- The delays that may occur at different times of day due to processes at a site.
- Vehicle stopping positions at an interchange.

8.4.1 Fixed Objects – Model Dependencies

The FIXED OBJECT Model describes fixed stops and out of vehicle passenger aspects of a network and is itself divided into a number of separate submodels.

The submodels themselves depend on a number of general NeTEx framework models (for example, GENERIC POINT AND LINK, NOTICE, etc.) described elsewhere – See REUSABLE COMPONENTs in 7.7

Figure 386 shows the dependencies between the FIXED OBJECT physical submodels. The terminal model is the SITE FRAME. This VERSION FRAME organises the other elements into a coherent set of elements suitable for exchange as a serialised file. The payload elements are contained in the following packages:

SITE FRAME:

- SITE: models a location that passengers travel from or to.
- STOP PLACE: models a Station or Stop.
- FLEXIBLE STOP PLACE: models an area covered by a FTS.
- POINT OF INTEREST: models a site other than a Station or stop to which a Passenger may want travel directions.
- PARKING: models a Parking facility associated with a SITE.
- NAVIGATION PATH MODEL: models the paths through a SITE.
- CHECK CONSTRAINT: models processes that may slow a passenger down when using a SITE.
- VEHICLE STOPPING MODEL: models where vehicles stop within a SITE.

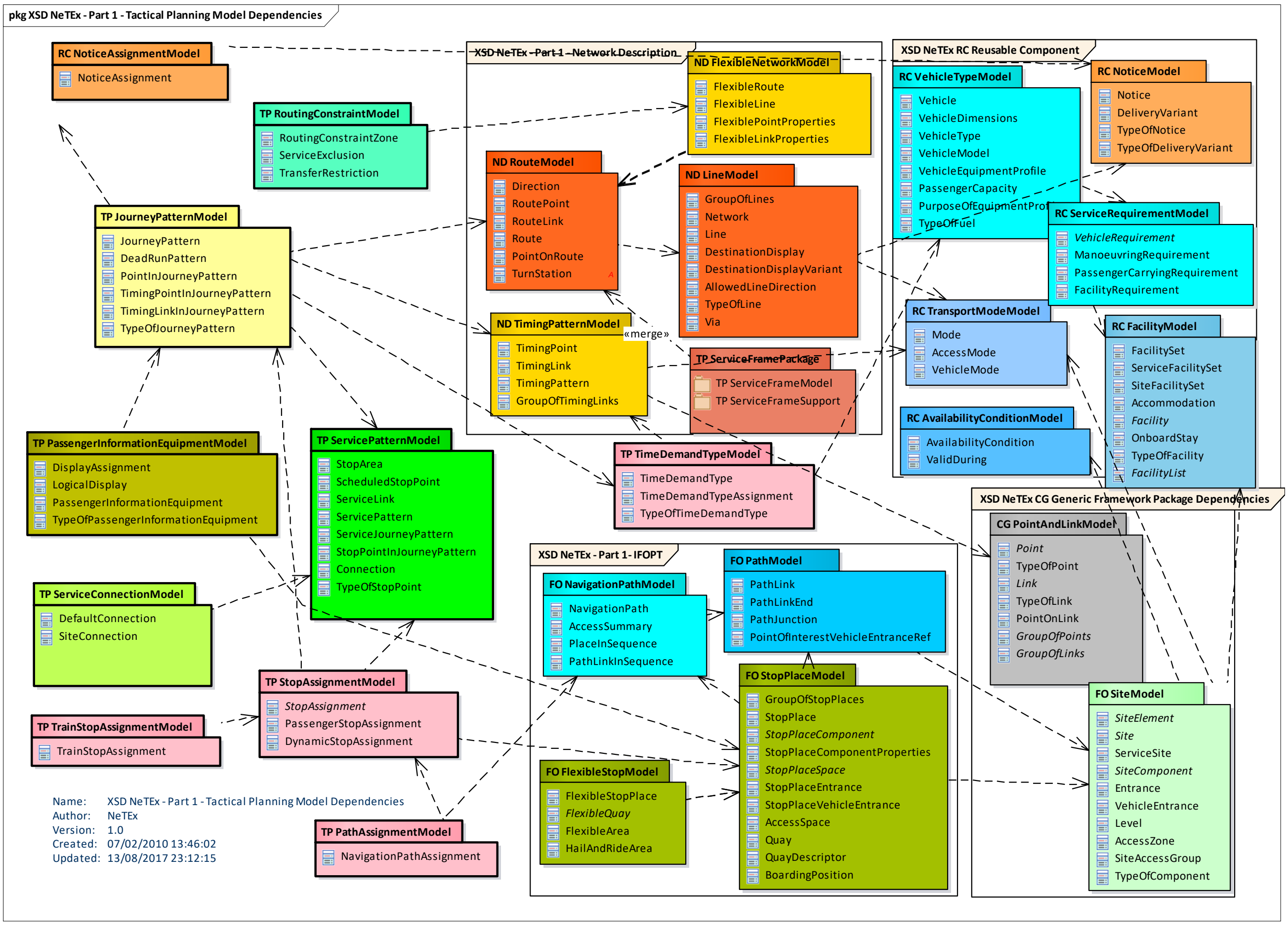


Figure 483 – Fixed objects – Model dependencies (UML)

8.4.2 Site Frame

8.4.2.1 Site Frame – Physical Model

The SITE FRAME (Figure 484) holds a coherent set of SITE and ancillary elements for data exchange. These elements are explained in subsequent sections.

8.4.2.1.1 Site Frame Overview – Physical Model

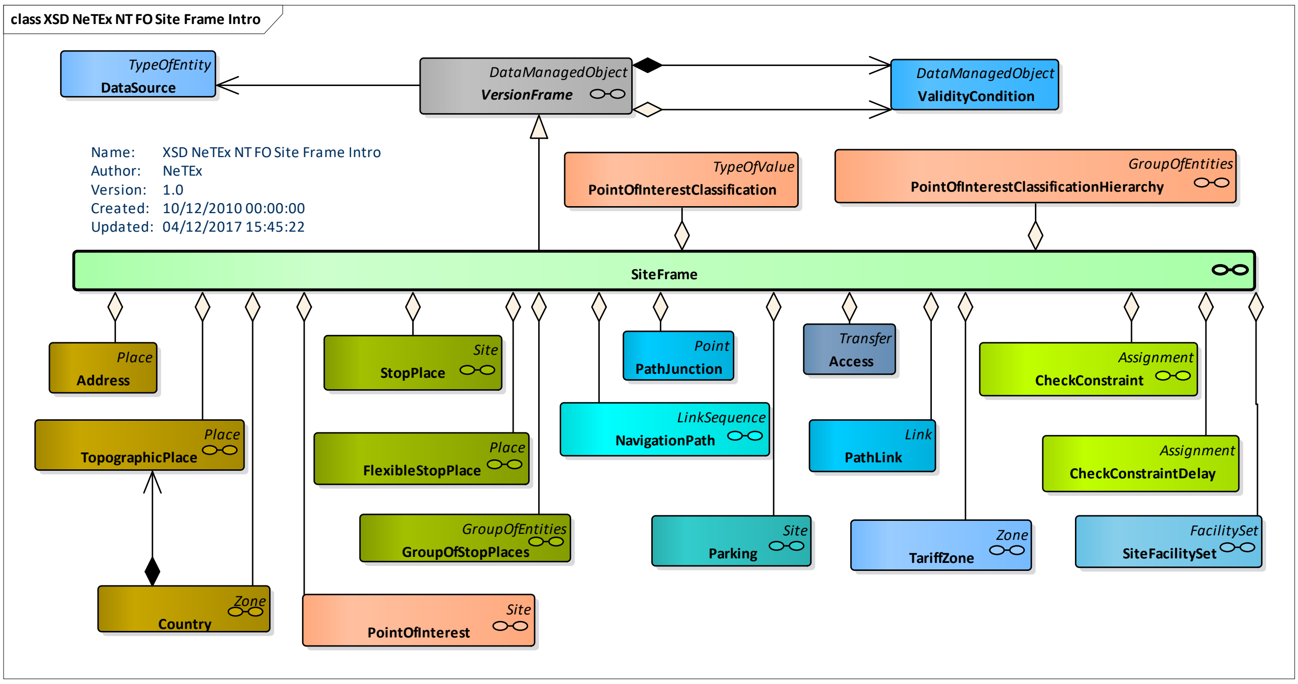


Figure 484 – Site Frame – Physical Model (UML) – Overview

8.4.2.1.2 Site Frame Details – Physical Model

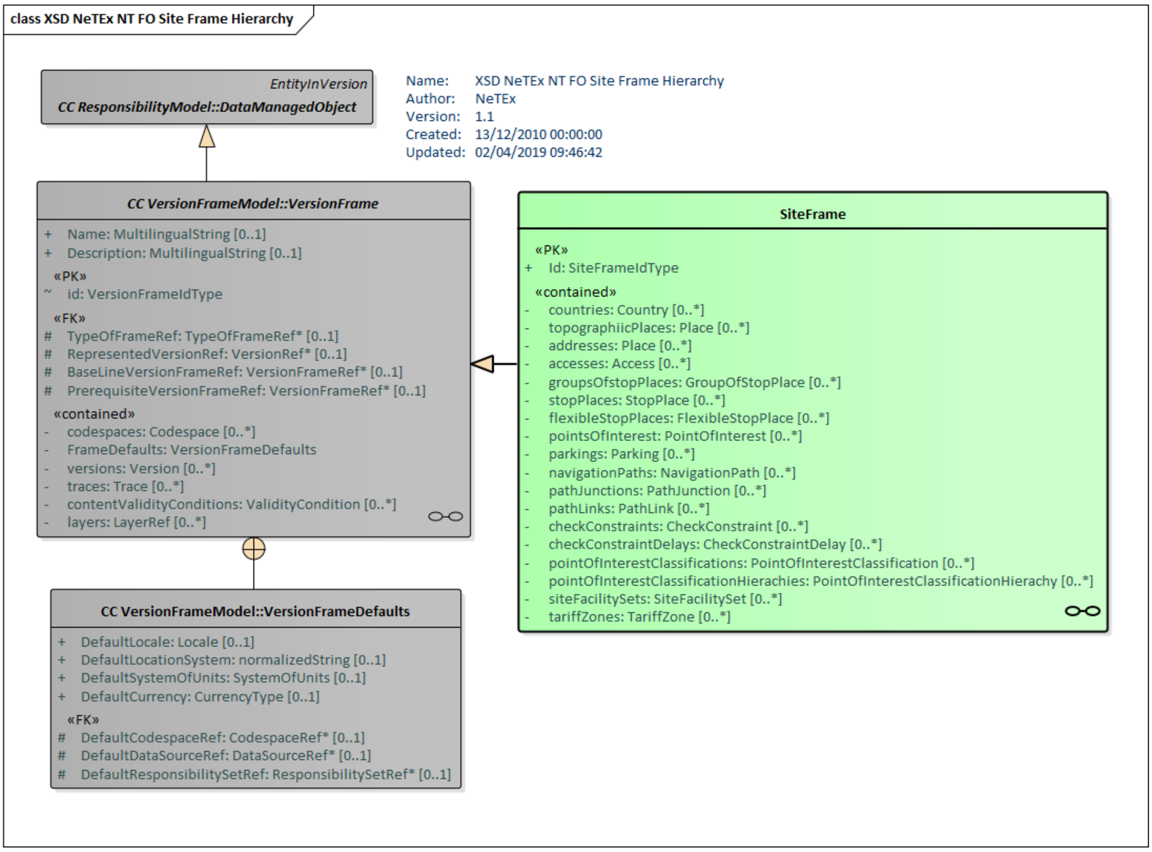


Figure 485 – Site Frame – Physical Model (UML) – Details

8.4.2.2 Site Frame – Attributes and XSD

8.4.2.2.1 Site Frame – Model Element

A set of SITE data (and other data logically related to these) to which the same VALIDITY CONDITIONs have been assigned.

The SITE FRAME (Figure 484) holds a coherent set of Site elements for data exchange. These elements are explained in subsequent sections.

Table 399 – *SiteFrame* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*VersionFrame*	*::>*	SITE FRAME inherits from VERSION FRAME.
«PK»	***id***	*SiteFrameIdType*	1:1	Identifier of Site Frame.
Place Frame Group	***topographicPlaces***	*Place*	0:*	TOPOGRAPHIC PLACEs in frame.
	addresses	*Address*	0:*	ADDRESSes in frame.
	accesses	*Access*	0:*	ACCESSes in frame.
Site Frame Group	***stopPlaces***	*StopPlace*	0:*	STOP PLACEs in frame.
	flexibleStopPlaces	*FlexibleStopPlace*	0:*	FLEXIBLE STOP PLACEs in frame.
	pointsOfInterest	*PointOfInterest*	0:*	POINTs OF INTEREST in frame.
	parking	*Parking*	0:*	PARKINGs in frame.
Path Frame Group	***navigation Paths***	*NavigationPath*	0:*	NAVIGATION PATHs in frame.
	pathJunctions	*PathJunction*	0:*	PATH JUNCTIONs in frame.
	pathLinks	*PathLink*	0:*	PATH LINKs in frame.
«cntd»	***checkConstraints***	*CheckConstraint*	0:*	CHECK CONSTRAINTs in frame.
«cntd»	***checkConstraint-Delays***	*CheckConstraintDelay*	0:*	CHECK CONSTRAINT DELAYs in frame.
Point Of Interest Group	***pointOfInterest-Classifications***	*PointOfInterest-Classification*	0:*	POINT OF INTEREST CLASSIFICATIONs in frame.
	pointOfInterest-Classification-Hierachies	*PointOfInterest-ClassificationHierachy*	0:*	POINT OF INTEREST CLASSIFICATION HIERARCHies in frame.
«cntd»	***siteFacilitySets***	*SiteFacilitySet*	0:*	SITE FACILITY SETs in frame.
«cntd»	***tariffZones***	*TariffZone*	0:*	TARIFF ZONEs in frame.

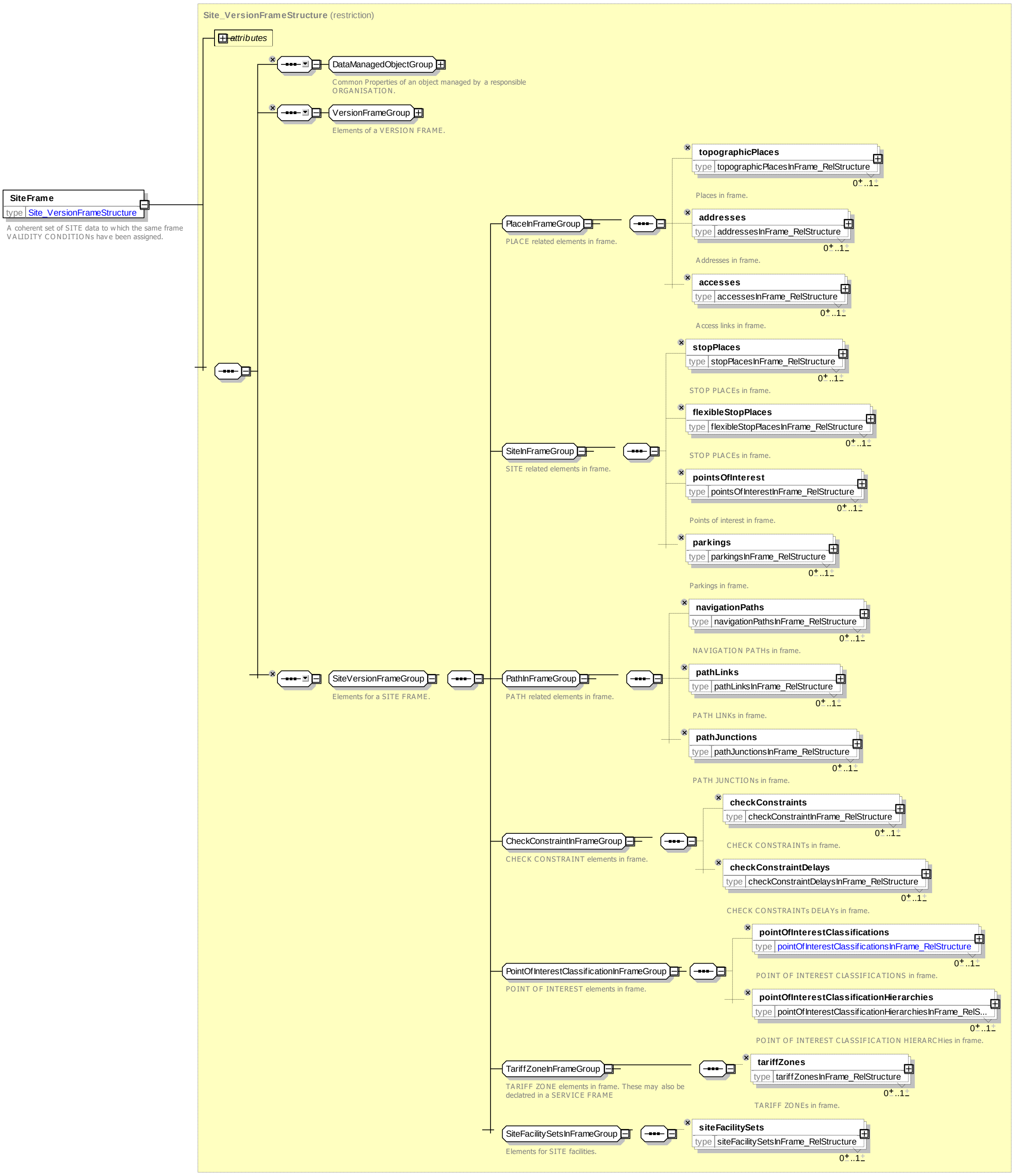


Figure 486 – *SiteFrame* – XSD

8.4.3 Site

The SITE MODEL provides a general description of common properties of a physically situated location, such as a station or point of interest, including its entrances, levels, equipment, paths, accessibility properties, etc. The SITE MODEL is refined by specific submodels such as STOP PLACE, POINT OF INTEREST, PARKING, etc. to define specific types of ADDRESSABLE PLACE.

8.4.3.1 SITE – Conceptual MODEL

8.4.3.1.1 SITE – Conceptual MODEL – Basic

The following shows the basic elements making up a SITE:

— SITE COMPONENT: an element of a SITE describing part of its structure, such as a platform, concourse, ticket hall, entrance hall forecourt, room, retail area, etc. These may be specialised in submodels.

— LEVEL: an identified storey (ground, first, basement, mezzanine, etc.) within an interchange building or SITE on which SITE COMPONENTs reside.

— ENTRANCE: a physical entrance or exit to/from a SITE.

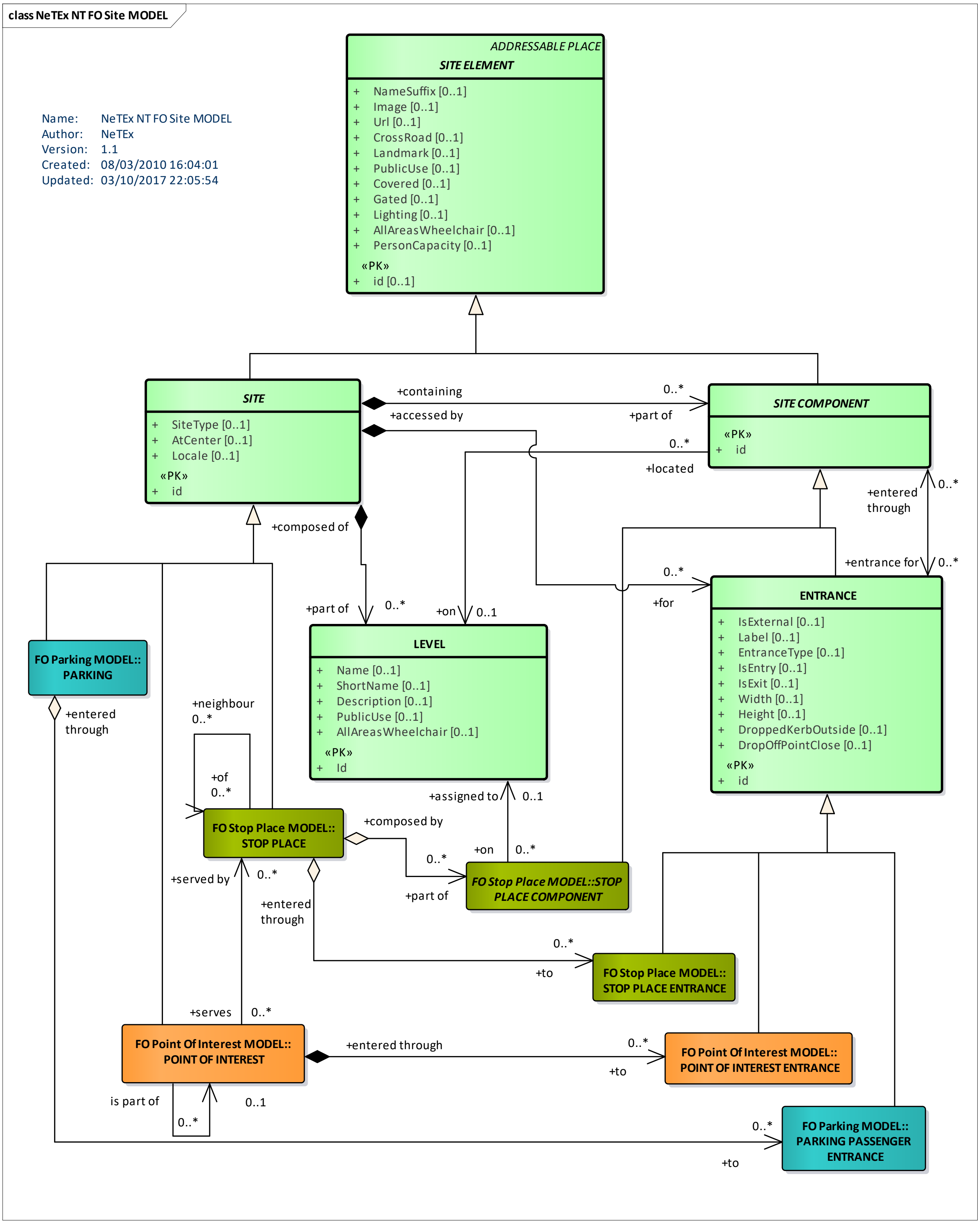


Figure 487 – SITE – Conceptual MODEL – Basic (UML)

8.4.3.1.2 SITE – Conceptual MODEL – Detailed

The SITE MODEL allows a number of additional characteristics to be specified that are relevant to passenger use of the SITE (see Physical model for details).

- The accessibility of the SITE and its components can be described using ACCESSIBILITY ASSESSMENTs. Accessibility is described further in the next section.
- The availability of parts of the SITE may be specified using a VALIDITY CONDITION.
- The nature and likely time impact of processes taking place at points on the site, such as check in, ticket purchase, security checks etc., can be specified with a CHECK CONSTRAINT and a CHECK CONSTRAINT DELAY.
- The ROAD and POSTAL ADDRESS of components can be specified.
- The EQUIPMENT found on the site such as barriers, gates, stairs, lifts, seats, lavatories etc., can be specified.
- The NAVIGATION PATHs to go between the different points of the site can be specified. These are made up of PATH LINKs and PATH JUNCTIONs – see separate model.

8.4.3.1.3 SITE – Conceptual MODEL – Accessibility

The accessibility of the SITE and its components can be specified using ACCESSIBILITY ASSESSMENTs. (See REUSABLE COMPONENT MODEL). A number of different types of EQUIPMENT may also be relevant for disabled users.

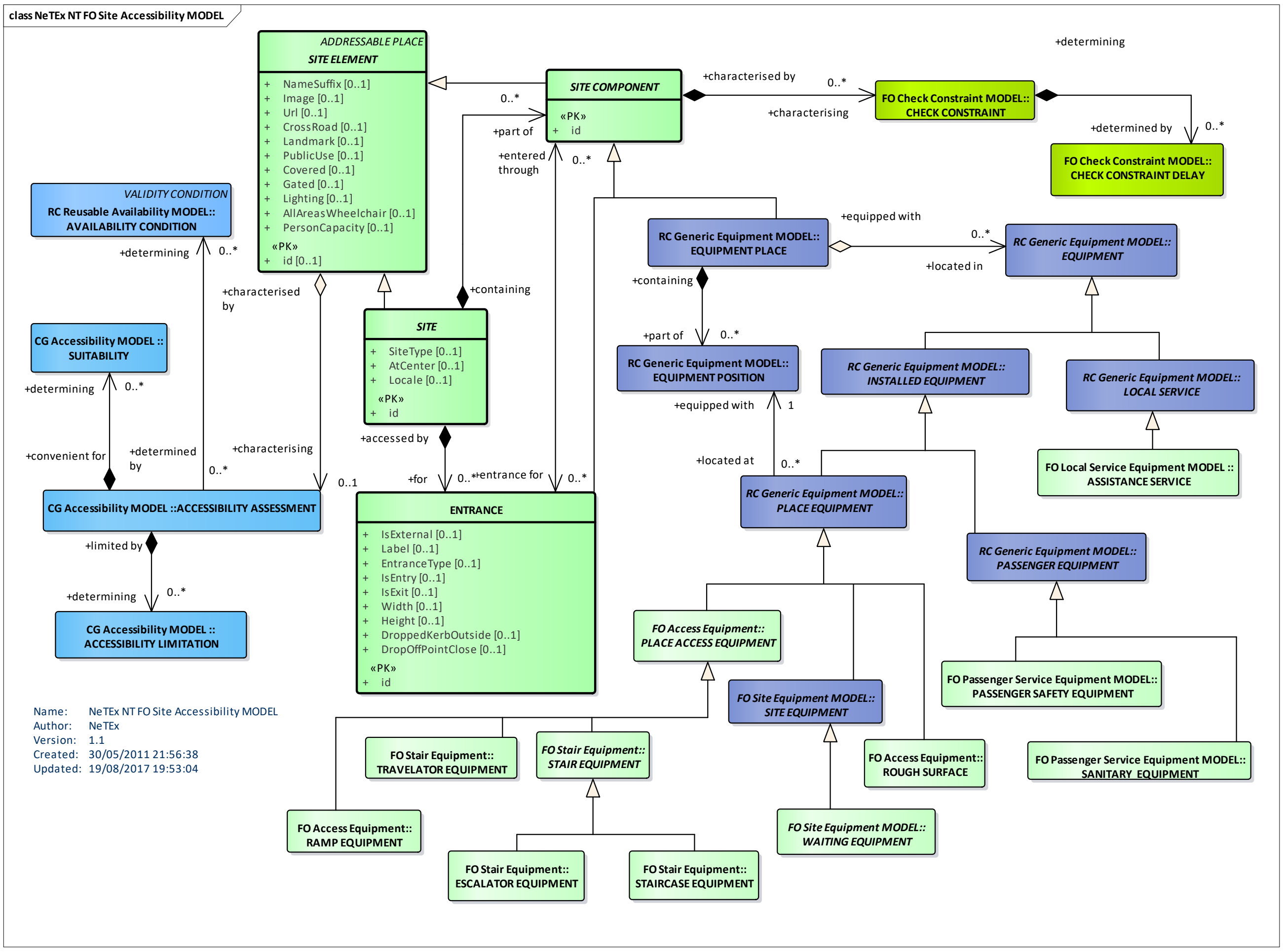


Figure 488 – SITE ACCESSIBILITY – Conceptual MODEL (UML)

8.4.3.2 Site – Physical model

The following figure shows physical attributes of the SITE model.

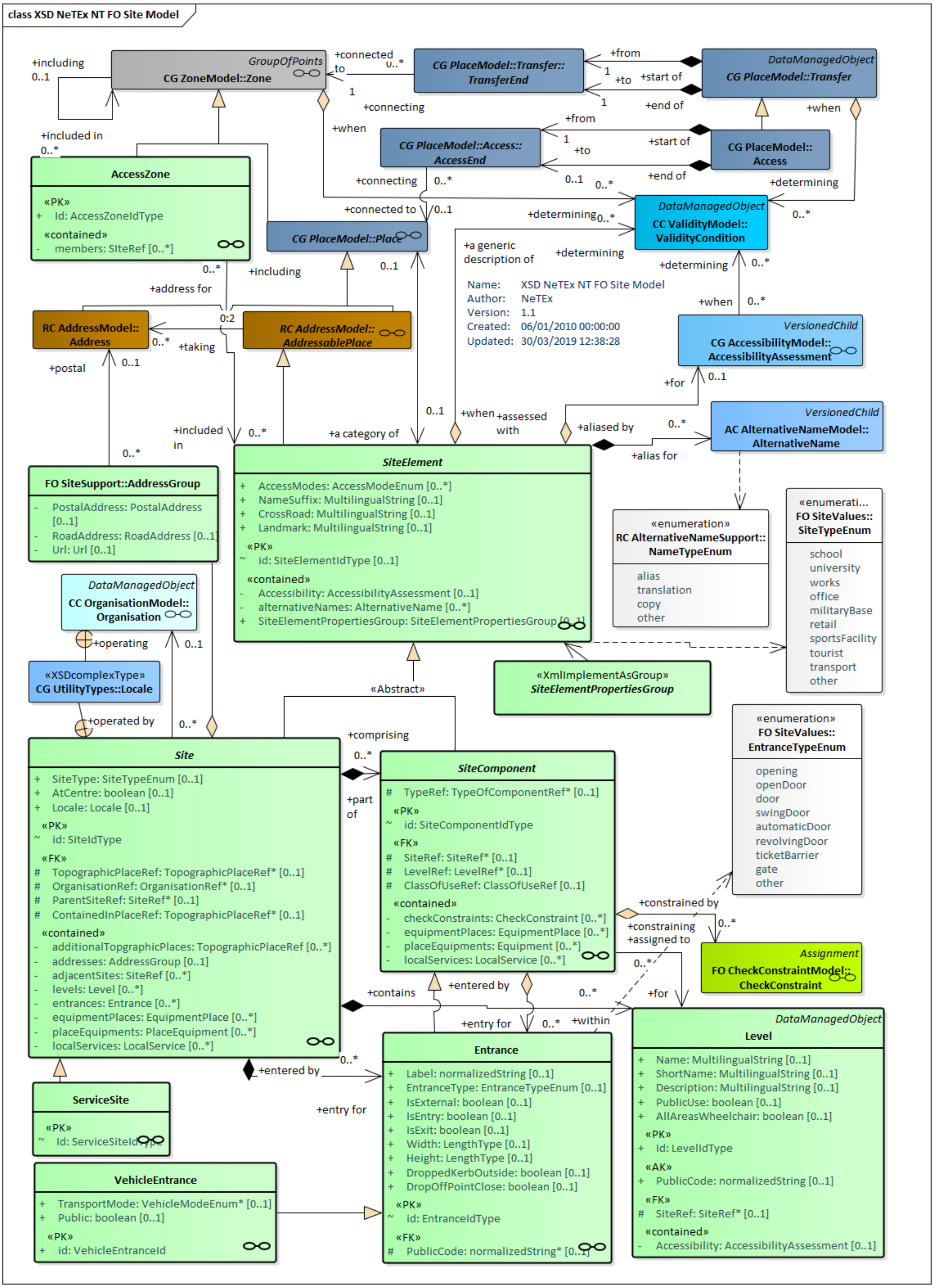


Figure 489 – Site– Physical Model (UML) – Detail

8.4.3.2.1.1 Site Element – Physical model

The following figure shows further details of the SITE ELEMENT model

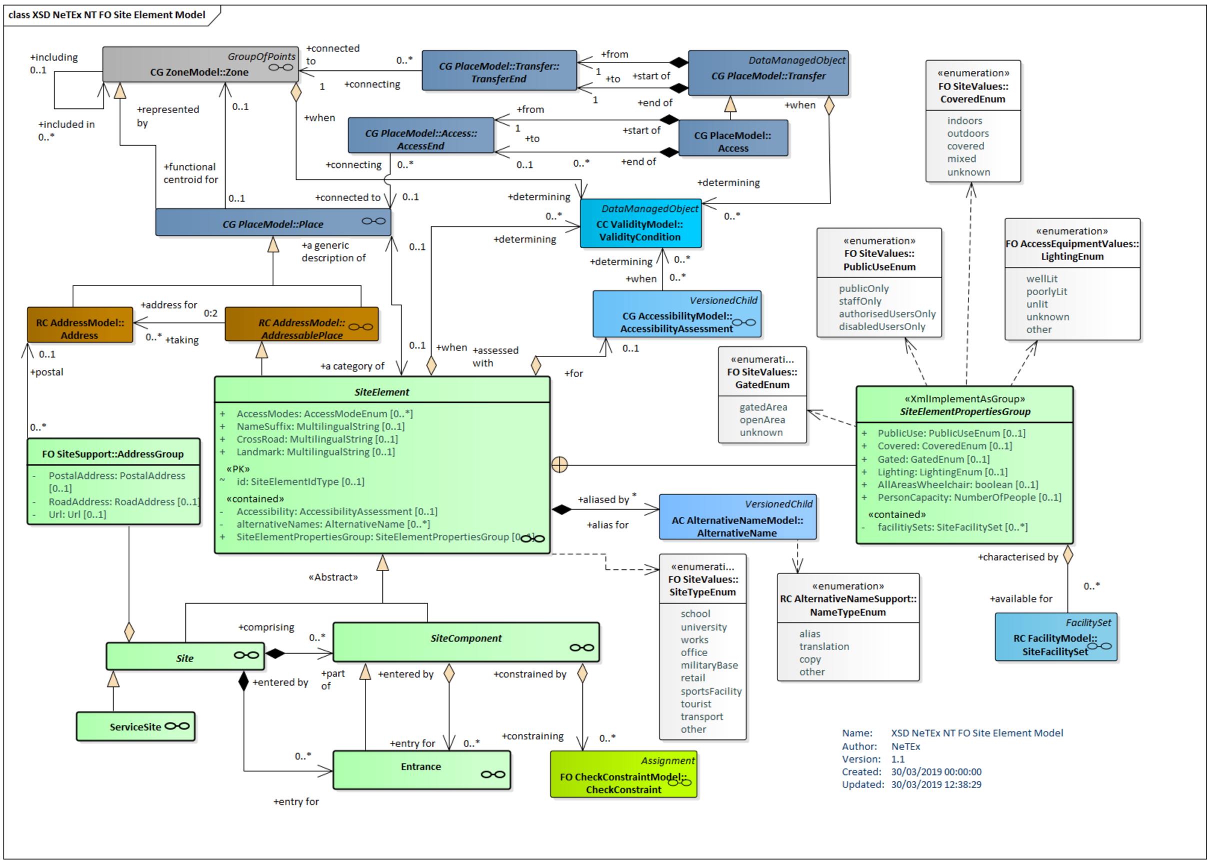


Figure 490 – Site Element – Physical Model – Accessibility (UML)

8.4.3.2.1.2 Site Accessibility – Physical model

The following figure shows attributes of the SITE model relating to accessibility.

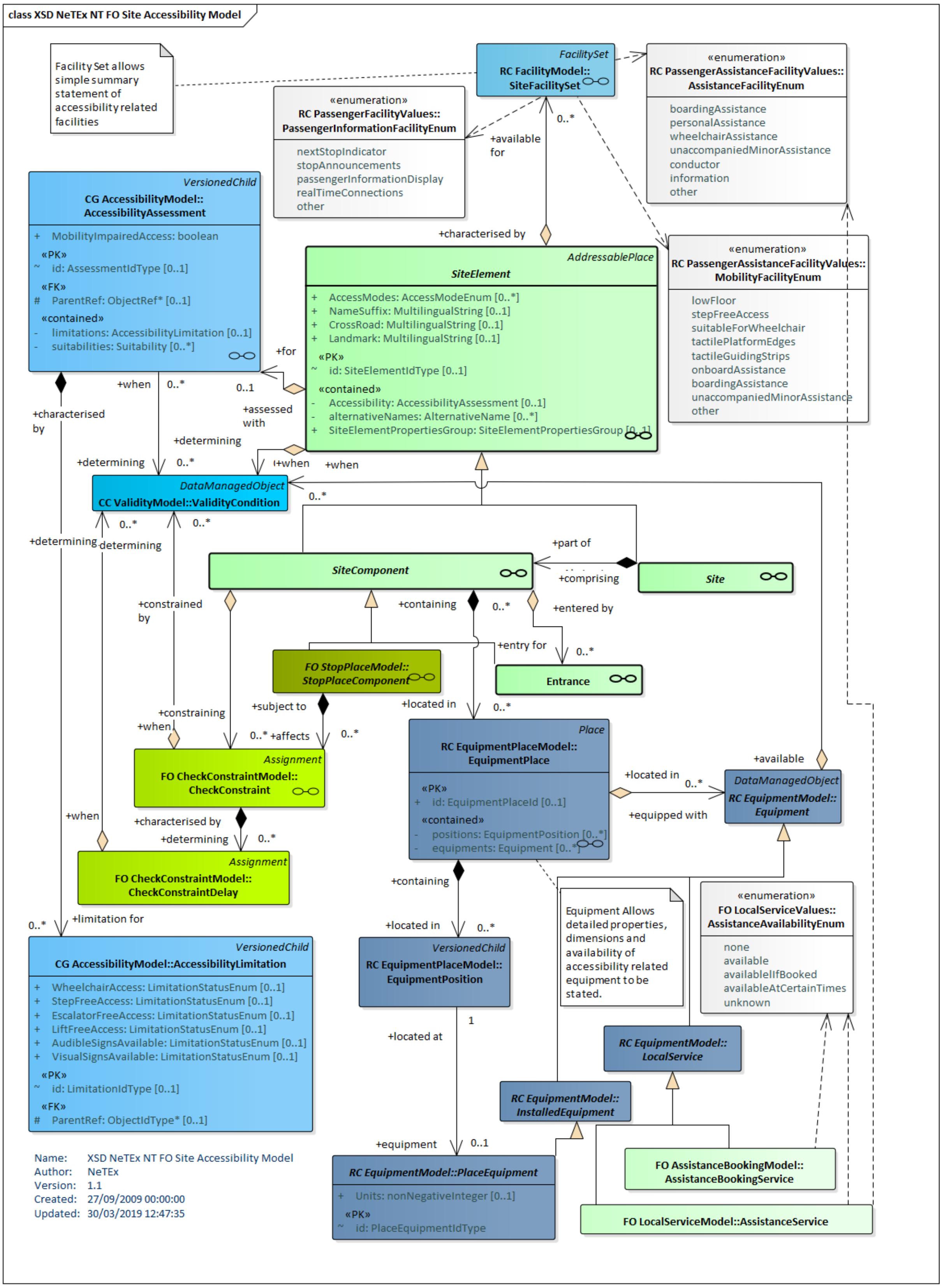


Figure 491 – Site – Physical Model – Accessibility (UML)

8.4.3.3 Site – Attributes and XSD

8.4.3.3.1 SiteElement – Model Element

A type of PLACE specifying common properties of a SITE or a SITE COMPONENT. May have an ACCESSIBILITY ASSESSMENT and other properties to describe it.

Table 400 – *SiteElement* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*AddressablePlace*	*::>*	SITE ELEMENT inherits from *ADDRESSABLE* PLACE.
«PK»	***id***	*SiteElementIdType*	0:1	Identifier of SITE ELEMENT.
«cntd»	***Accessibility-Assessment***	*AccessibilityAssessment*	0:1	ACCESSIBILITY of SITE ELEMENT.
«enum»	***AccessModes***	*AccessModeEnum*	0:*	Access modes by which SITE ELEMENT may be accessed. See ACCESS MODE for allowed values.
	NameSuffix	*MultilingualString*	1:1	Suffix to use on Name.
«cntd»	***alternativeNames***	*AlternativeName*	0:*	Alternative names for SITE ELEMENT.
	Image	*xsd:anyUri*	0:1	Image associated with SITE ELEMENT.
	CrossRoad	*MultilingualString*	0:1	Name of nearest cross road or crossing street on which site is on which can be used to locate SITE.
	Landmark	*MultilingualString*	0:1	Name of a nearby Landmark.
XGRP	***SiteElement-PropertiesGroup***	***xmlGroup***	0:1	Further properties of the SITE ELEMENT: see below.

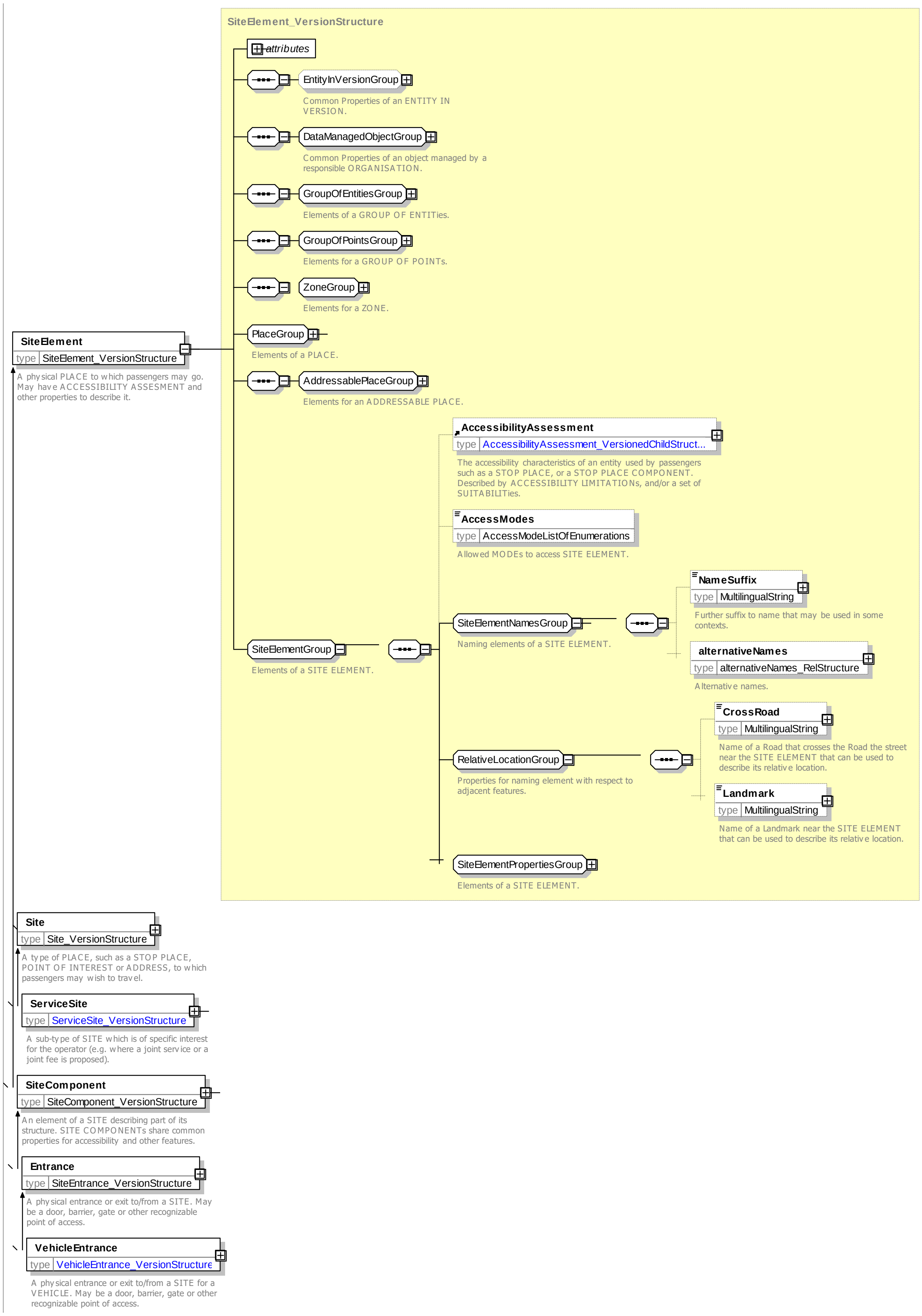


Figure 492 – *SiteElement* – XSD

8.4.3.3.1.1 SiteElementPropertiesGroup – Model Group

SiteElementPropertiesGroup holds common properties of a SITE ELEMENT***.***

Table 401 – *SiteElementPropertiesGroup* – Group

Classification	Name	Type	Cardinality	Description
«enum»	***PublicUse***	*PublicUseEnum*	0:1	Whether SITE ELEMENT can be used by the general public. See allowed values below.
enum»	***Covered***	*CoveredEnum*	0:1	Nature of covering. See allowed values below.
enum»	***Gated***	*GatedEnum*	0:1	Whether element is within a gated area. See allowed values below.
enum»	***Lighting***	*LightingEnum*	0:1	How element is lit. See allowed values below.
	AllAreas-Wheelchair-Accessible	*xsd:boolean*	0:1	Whether all areas of component are accessible in a Wheelchair.
	PersonCapacity	*NumberOfPeople*	0:1	Number of people that can be in component at a time.
«cntd»	***facilities***	*SiteFacilitySet*	0:*	SITE FACILITY SET associated with SITE ELEMENT.

Figure 493 – *SiteElementPropertiesGroup* XSD

PublicUse – Allowed values

The following table shows the allowed values for ***PublicUse (****PublicUseEnum*eration***).***

Table 402 – *PublicUse* – Allowed values

Value	Description
publicOnly	Component can be used by public.
staffOnly	Component can be used only by staff.
authorisedPublic-Only	Component can be used only by authorised users.
disabledPublicOnly	Component can be used only by disabled users.
all	Component can be used by all

Covered – Allowed values

The following table shows the allowed values for **Covered** (*CoveredEnumeration*).

Table 403 – *Covered* – Allowed Values

Value	Description
indoors	Component is indoors.
outdoors	Component is outdoors.
covered	Component is covered.
mixed	Component is mixed.
unknown	Component cover is unknown.

Gated – Allowed values

The following table shows the allowed values for ***Gated (****GatedEnumeration****).***

Table 404 – *Gated* – Allowed values

Value	Description
gatedArea	Component is within a gated section.
openArea	Component is outside of any gated section.
unknown	Component gated status is unknown.

Lighting – Allowed values

The following table shows the allowed values for ***Lighting (****LightingEnumeration****).***

Table 405 – *Lighting* – Allowed values

Value	Description
wellLit	Component is well lit.
poorlyLit	Component is poorly lit.
unlit	Component is not lit at all.
other	Component has other lighting.
unknown	Component lighting is unknown.

8.4.3.3.2 Site – Model Element

A type of PLACE, such as a STOP PLACE, POINT OF INTEREST or ADDRESS, to which passengers may wish to travel.

Table 406 – *Site* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SiteElement*	*::>*	SITE inherits from SITE ELEMENT.
«PK»	***id***	*SiteIdType*	1:1	Identifier of SITE.
XGRP	***SiteGroup***	***xmlGroup***	0:1	Elements describing SITE.

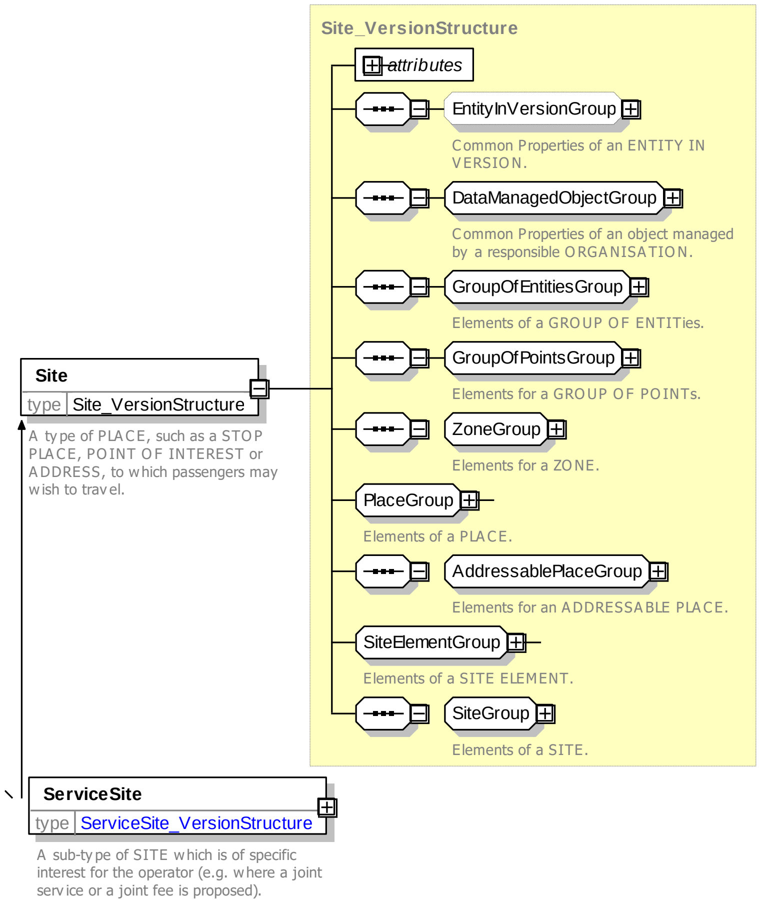


Figure 494 – *Site* – XSD

8.4.3.3.2.1 SiteGroup – XML Group

Elements describing SITE.

Table 407 – *SiteGroup* – Group

Classification	Name	Type	Cardinality	Description
«FK»	***TopographicPlaceRef***	*TopographicPlaceRef*	0:1	Reference to a TOPOGRAPHIC PLACE associated with SITE.
«cntd»	***additional-Topographic-Places***	*TopographicPlaceRef*	0:*	Additional Topographic Places in which SITE is located.
«enum»	***SiteType***	*SiteTypeEnum*	0:1	Type of SITE. See allowed values below.
	AtCenter	*xsd:boolean*	0:1	Whether the site is central to the locality, referenced at town centre.
	Locale	*Locale*	0:1	Locale setting time zone, default language etc, for the STOP PLACE.
XGRP	SiteRelationGroup	***xmlGroup***	0:1	Elements describing relations to other SITEs.

«cntd»	***levels***	*Level*	0:*	LEVELs for SITE.
«cntd»	***entrances***	*(Entrance)*	0:*	ENTRANCEs to SITE.
«cntd»	***equipmentPlaces***	*EquipmentPlace*	0:*	EQUIPMENT PLACEs associated with STOP PLACE COMPONENT.
«cntd»	***placeEquipments***	*(PlaceEquipment)*	0:*	Items of fixed EQUIPMENT that may be located in places within the SITE ELEMENT.
«cntd»	***localServices***	*(LocalService)*	0:*	LOCAL SERVICEs that may be located in PLACEs within the SITE ELEMENT.

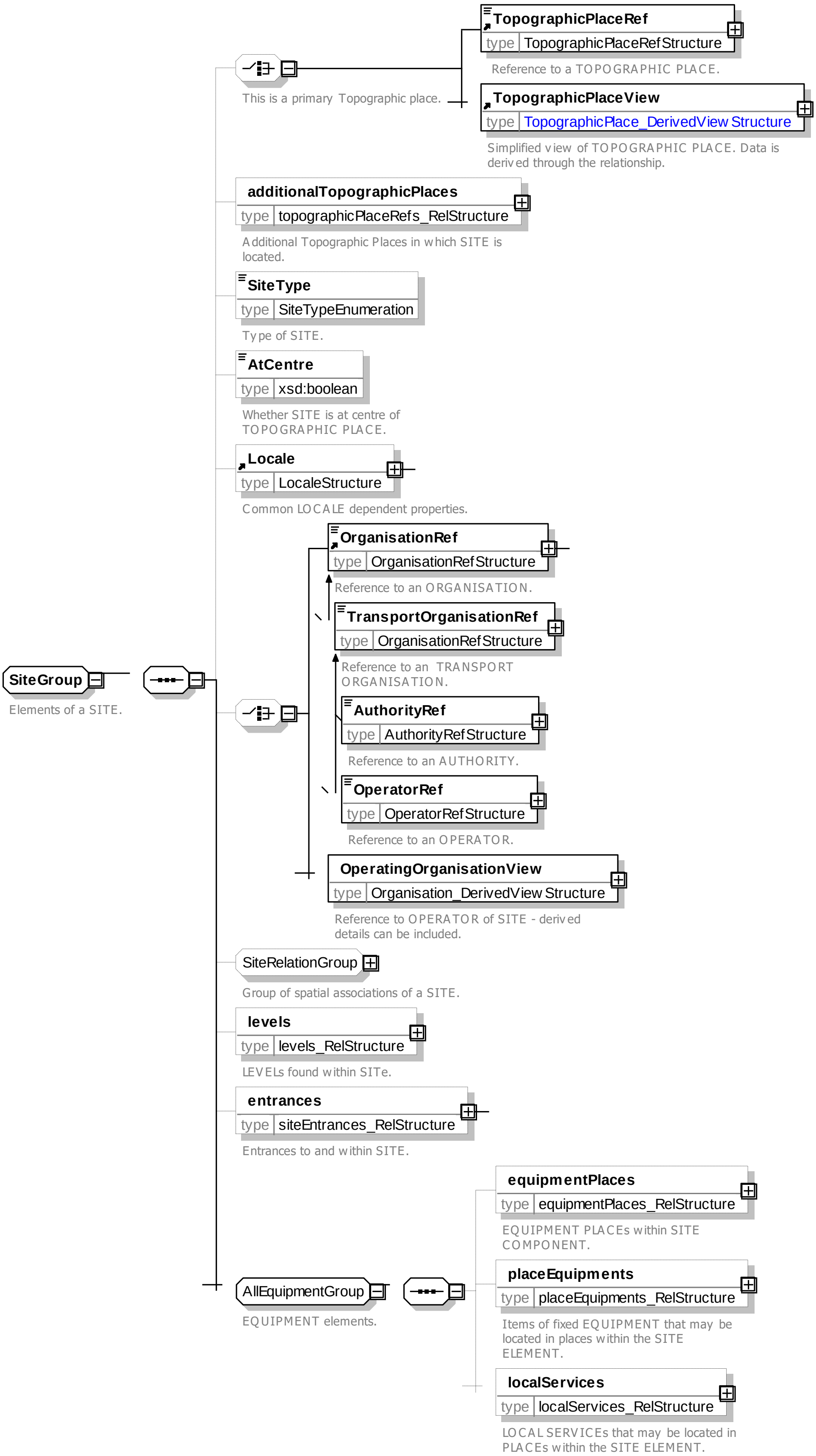


Figure 495 – *SiteGroup* – XSD

***SiteType* – Allowed values**

The following table shows the allowed values for Site (*SiteTypeEnumeration)*.

Table 408 – *SiteType* – Allowed values

Value	Description
school	Site is a School
university	Site is a University
works	Site is a Works
office	Site is an Office
militaryBase	Site is a Military base
retail	Site is a Retail facility
sports	Site is a Sports facility
tourist	Site is a Tourist Attraction facility
other	Other type of site

8.4.3.3.2.2 SiteRelationGroup – XML Group

Elements describing relations to other SITEs.

Table 409 – *SiteRelationGroup* – Group

Classification	Name	Type	Cardinality	Description
«FK»	***ParentSiteRef***	(*SiteRef*)	0:1	Parent SITE which contains this SITE.
«cntd»	***adjacentSites***	(*SiteRef*)	0:*	Adjacent Sites to SITE.
«FK»	***ContainedIn-PlaceRef***	*TopographicPlaceRef*	0:1	Parent SITE which contains this SITE.

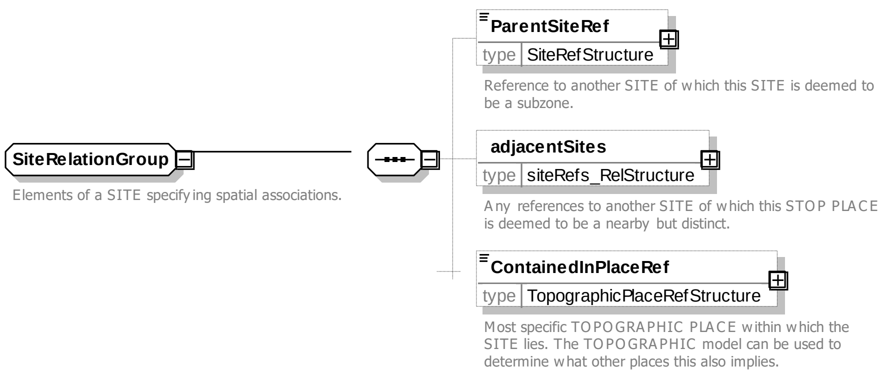


Figure 496 – *SiteRelationGroup* – XSD

8.4.3.3.3 ServiceSite – Model Element

A sub-type of SITE which is of specific interest for the operator (e.g. where a joint service or a joint fee is proposed).

In NeTEx, STOP PLACE is used for SITEs which are for access to transport.

Table 410 – *ServiceSite* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Site*	::>	SERVICE SITE inherits from SITE.
«PK»	id	*ServiceSiteIdType*	1:1	Identifier of SERVICE SITE.

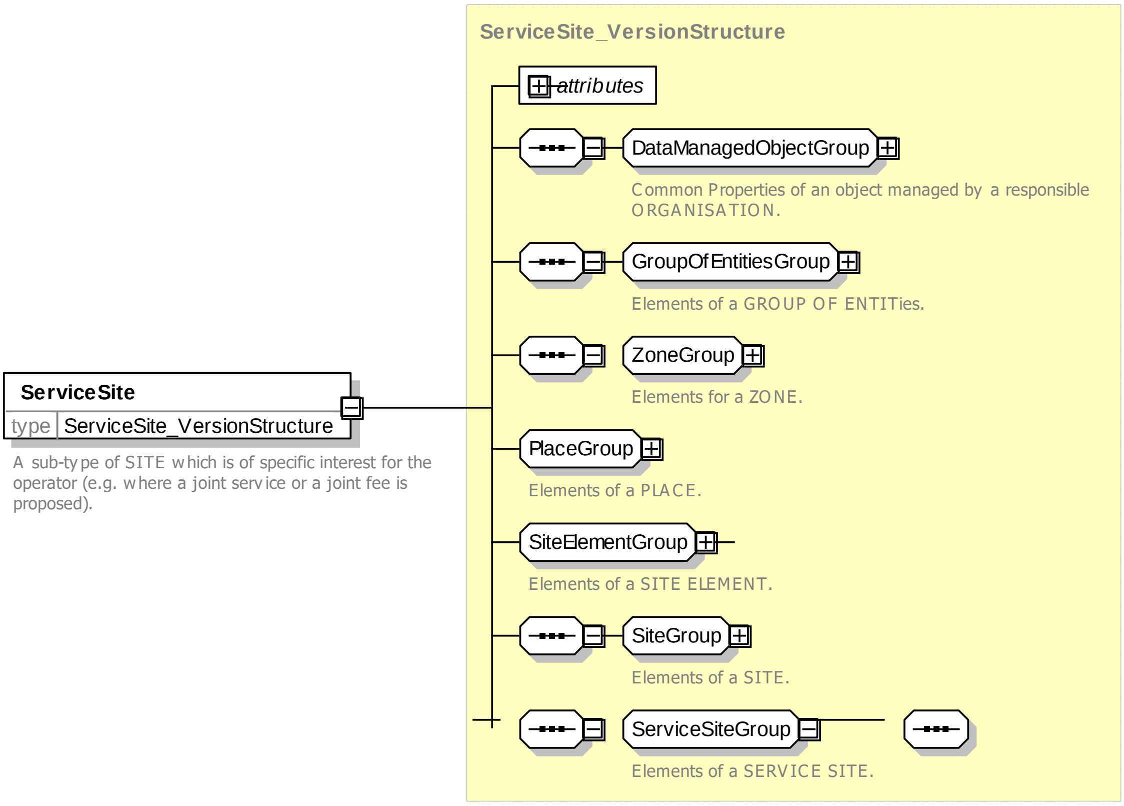


Figure 497 – *ServiceSite* – XSD

8.4.3.3.4 *SiteComponent – Model Element*

An element of a SITE describing part of its structure. SITE COMPONENTs share common properties for EQUIPMENT, management, accessibility and other features.

Table 411 – *SiteComponent* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*SiteElement*	::>	SITE COMPONENT inherits from SITE ELEMENT.
«PK»	***id***	*SiteComponentIdType*	1:1	Identifier of SITE COMPONENT.
«FK»	***SiteRef***	*(SiteRef)*	0:1	Reference to parent of SITE COMPONENT.
«FK»	***LevelRef***	*LevelRef*	0:1	Reference to LEVEL associated with the SITE COMPONENT.

«FK»	***ClassOfUseRef***	*ClassOfUseRef*	0:1	Reference to CLASS OF USE associated with the SITE COMPONENT.
«cntd»	***checkConstraints***	*CheckConstraint*	0:*	CHECK CONSTRAINTs associated with SITE COMPONENT.
«cntd»	***equipmentPlaces***	*EquipmentPlace*	0:*	EQUIPMENT PLACEs associated with STOP PLACE COMPONENT.
«cntd»	***equipments***	(*Equipment*)	0:*	EQUIPMENTs associated with SITE COMPONENT.
«cntd»	***localServices***	*LocalService*	0:*	LOCAL SERVICEs associated with SITE COMPONENT.

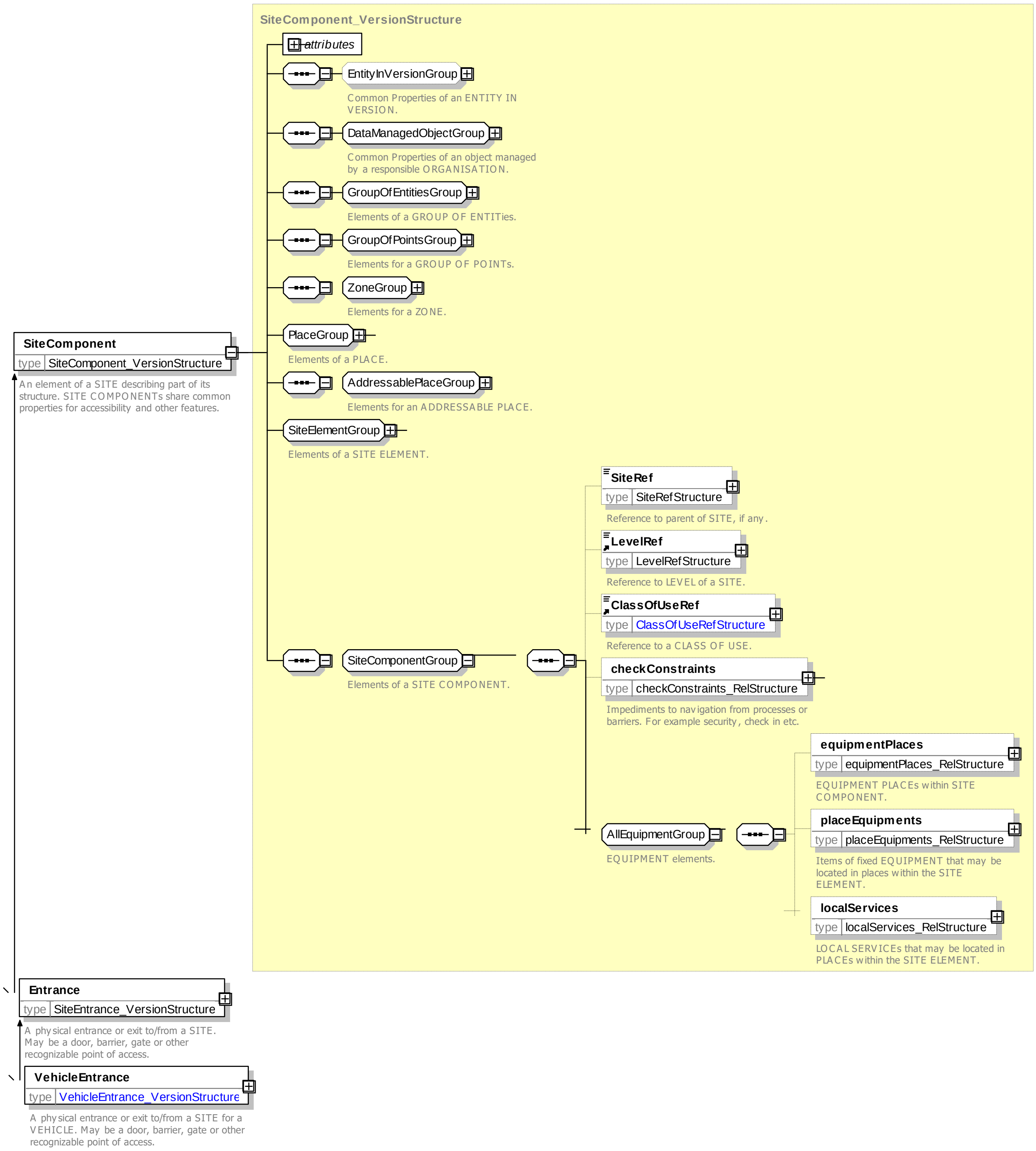


Figure 498 – *SiteComponent* – XSD

8.4.3.3.5 Entrance – Model Element

A physical entrance or exit to/from a SITE. May be a door, barrier, gate or other recognizable point of access.

Table 412 – *Entrance* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*SiteComponent*	::>	ENTRANCE inherits from SITE COMPONENT.
«PK»	***id***	*EntranceIdType*	1:1	Identifier of ENTRANCE.
XGRP	***SiteEntrance-Group***	***xmlGroup***	0:1	Elements describing an ENTRANCE.
XGRP	***ExternalEntrance-Group***	***xmlGroup***	0:1	Elements describing an external ENTRANCE.

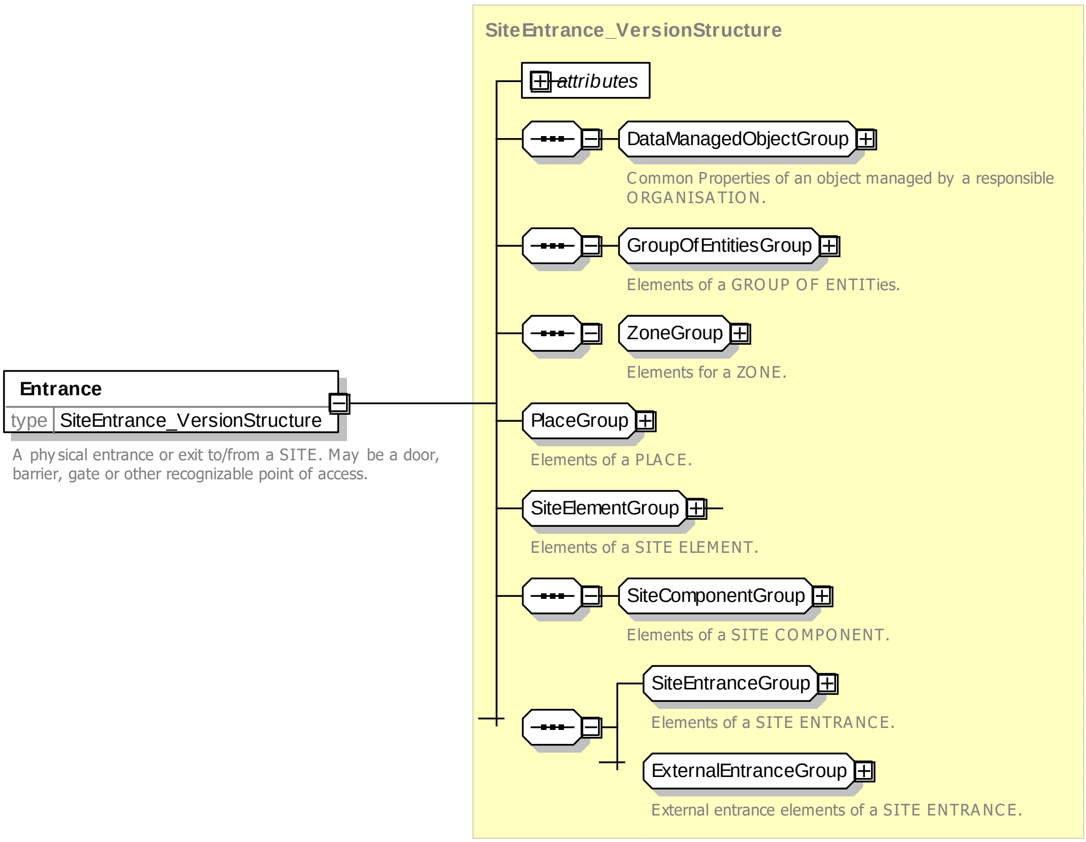


Figure 499 – *SiteEntrance – XSD*

8.4.3.3.5.1 SiteEntranceGroup – XML Group

Elements describing an ENTRANCE to a SITE.

Table 413 – *SiteEntranceGroup* – Group

Classification	Name	Type	Cardinality	Description
	PublicCode	*xsd:normalizedString*	0:1	Public code for ENTRANCE.
	Label	*xsd:normalizedString*	0:1	Label of ENTRANCE.
«enum»	***EntranceType***	*EntranceTypeEnum*	0:1	Type of ENTRANCE. See allowed values below.

	IsExternal	*xsd:boolean*	0:1	Whether ENTRANCE opens externally.
	IsEntry	*xsd:boolean*	0:1	Whether ENTRANCE can be used for entry to SITE.
	IsExit	*xsd:boolean*	0:1	Whether ENTRANCE can be used for exit from SITE.
	Width	*LengthType*	0:1	Width of ENTRANCE.
	Height	*LengthType*	0:1	Height of ENTRANCE.

Figure 500 – *SiteEntranceGroup – XSD*

EntranceType – Allowed values

The following table shows the allowed values for ***EntranceType (****EntranceTypeEnum*eration***).***

Table 414 – *EntranceType* – Allowed values

Value	Description
opening	Opening
openDoor	Open door
door	Door
swingDoor	Swing door
revolvingDoor	Revolving door
automaticDoor	Automatic door
ticketBarrier	Ticket barrier
gate	Gate
other	Other

8.4.3.3.5.2 ExternalEntranceGroup – XML Group

Elements describing an external ENTRANCE to a SITE.

Table 415 – *ExternalEntranceGroup* – Group

Classification	Name	Type	Cardinality	Description
	DroppedKerb-Outside	*xsd:boolean*	0:1	Whether there is a dropped curb outside ENTRANCE.
	DropOffPoint-Close	*xsd:boolean*	0:1	Whether ENTRANCE is near drop-off point.

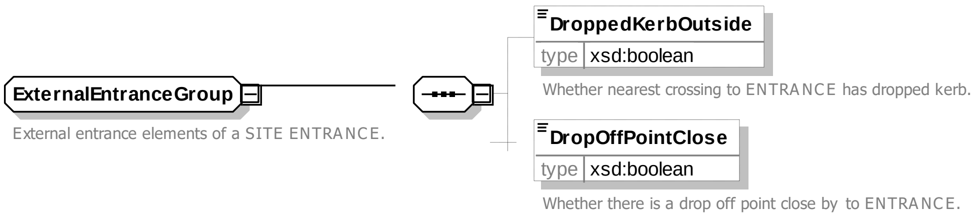


Figure 501 – *ExternalEntranceGroup – XSD*

8.4.3.3.6 VehicleEntrance – Model Element

A physical entrance or exit to/from a STOP PLACE for a Vehicle. May be a door, barrier, gate or other recognizable point of access.

Table 416 – *VehicleEntrance* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Entrance*</u>	::>	VEHICLE ENTRANCE inherits from SITE ENTRANCE.
«PK»	***id***	*VehicleEntranceId*	1:1	Identifier of VEHICLE ENTRANCE.
	Public	*xsd:boolean*	0:1	Whether VEHICLE ENTRANCE can be used by Private VEHICLEs.

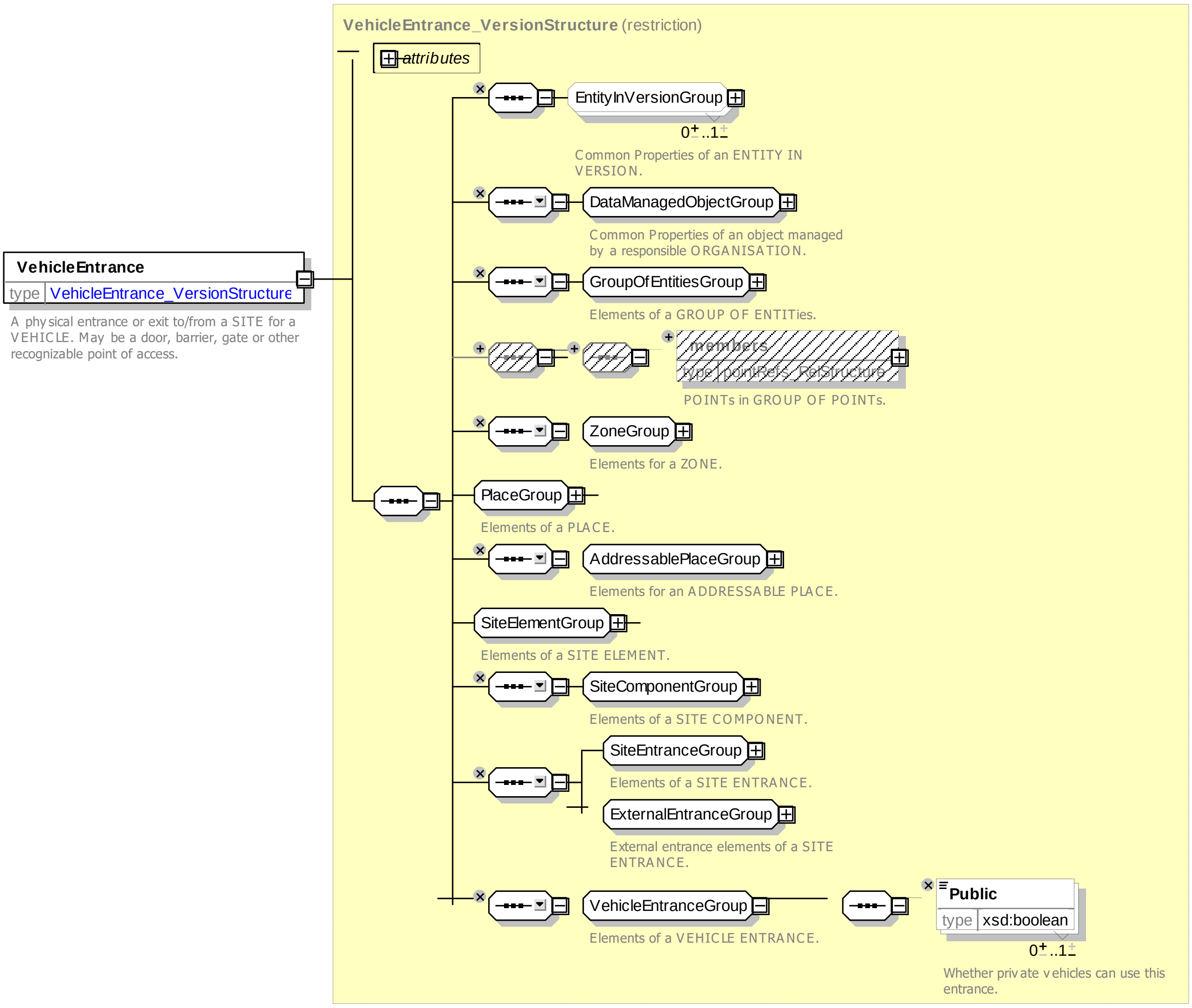


Figure 502 – *VehicleEntrance – XSD*

8.4.3.3.7 Level – Model Element

An identified storey (ground, first, basement, mezzanine, etc.) within an interchange building or SITE on which SITE COMPONENTs reside. A PATH LINK may connect components on different LEVELs.

Complex interchanges are often on multiple levels, each with a name, e.g. 'Arrivals', 'Departures', 'Platform Level', 'Entrance Level', etc. NeTEx allows the definition of named LEVELs, which may be topologically significant. Other elements can then be assigned a LEVEL that indicates their relative position.

Table 417 – *Level* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	LEVEL inherits from DATA MANAGED OBJECT.
«PK»	*id*	*LevelIdType*	1:1	Identifier of LEVEL.
	Name	*MultilingualString*	1:1	Name of LEVEL.

	ShortName	*MultilingualString*	1:1	Short Name of LEVEL.
	Description	*MultilingualString*	0:1	Further descriptive note about LEVEL.
	PublicUse	*xsd:boolean*	0:1	Whether LEVEL is for public use.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Public code of LEVEL.
«cntd»	***AccessibilityAssessment***	*AccessibilityAssessment*	0:1	ACCESSIBILITY of LEVEL.
	AllAreas-Wheelchair-Accessible	*xsd:boolean*	0:1	Whether all areas of LEVEL are accessible in a wheelchair.
«FK»	***SiteRef***	*(SiteRef)*	0:1	Reference to parent SITE of LEVEL.

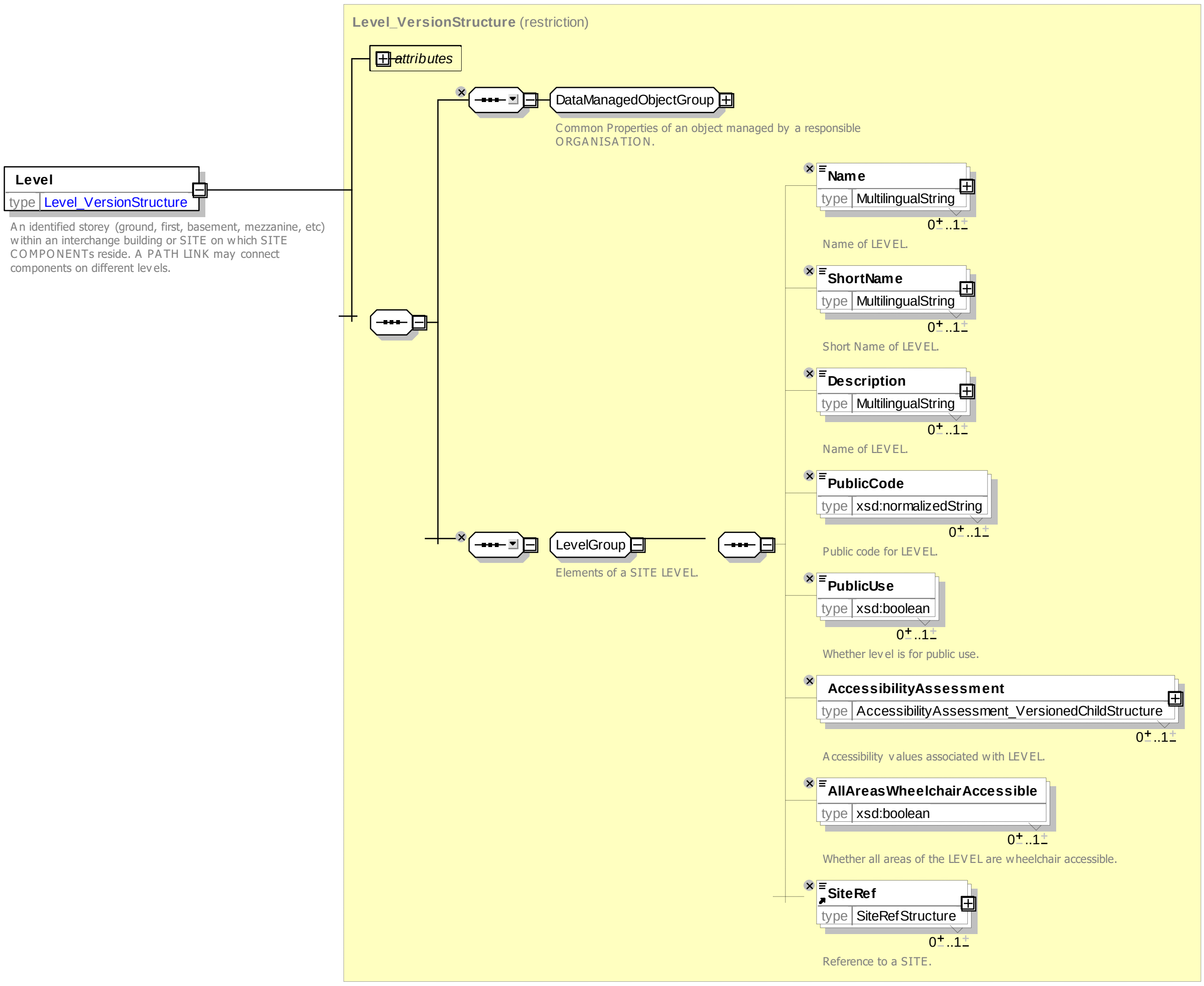


Figure 503 – *Level* – XSD

8.4.3.3.8 AccessZone – Model Element

A ZONE for which the duration to cover any ACCESS to a particular STOP POINT is the same.

In NeTEx A ZONE for which the duration to cover any ACCESS to a particular POINT is the same.

Table 418 – *AccessZone* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Zone*</u>	::>	ACCESS ZONE inherits from ZONE.
«PK»	id	*AccessZoneIdType*	1:1	Identifier of ACCESS ZONE.
«cntd»	***members***	*ScheduledStopPointRef*	0:*	List of references to SCHEDULED STOP POINTs that are members of ACCESS ZONE.
«cntd»	***Accessibility-Assessment***	<u>*AccessibilityAssessment*</u>	0:1	ACCESSIBILITY of ACCESS ZONE.
	AllAreas-Wheelchair-Accessible	*xsd:boolean*	0:1	Whether all areas of ACCESS ZONE are accessible in a wheelchair.

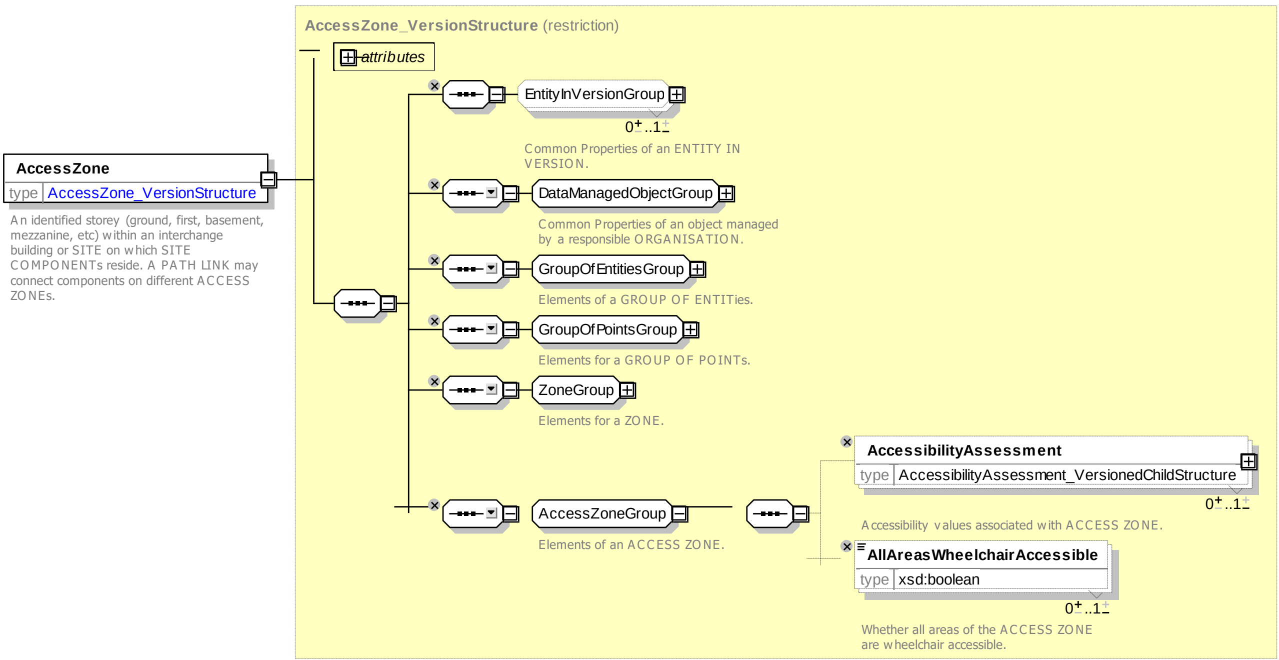


Figure 504 – *AccessZone* – XSD

8.4.3.4 SiteAccessGroup

SITE ACCESS GROUP describes the path and link aspects of a SITE COMPONENT or SITE. It may be included in specific components.

Table 419 – *SiteAccessGroup* – Element

Classification	Name	Type	Cardinality	Description
«cntd»	***pathLinks***	*PathLinkRef*	0:*	PATH LINKs associated with COMPONENT.
«cntd»	***pathJunctions***	*PathJunctionRef*	0:*	PATH JUNCTIONs associated with COMPONENT.
«cntd»	***accesses***	*AccessLinkRef*	0:*	ACCESS LINKs associated with COMPONENT.
«cntd»	***navigationPaths***	*NavigationPathRef*	0:*	NAVIGATION PATHs associated with COMPONENT.

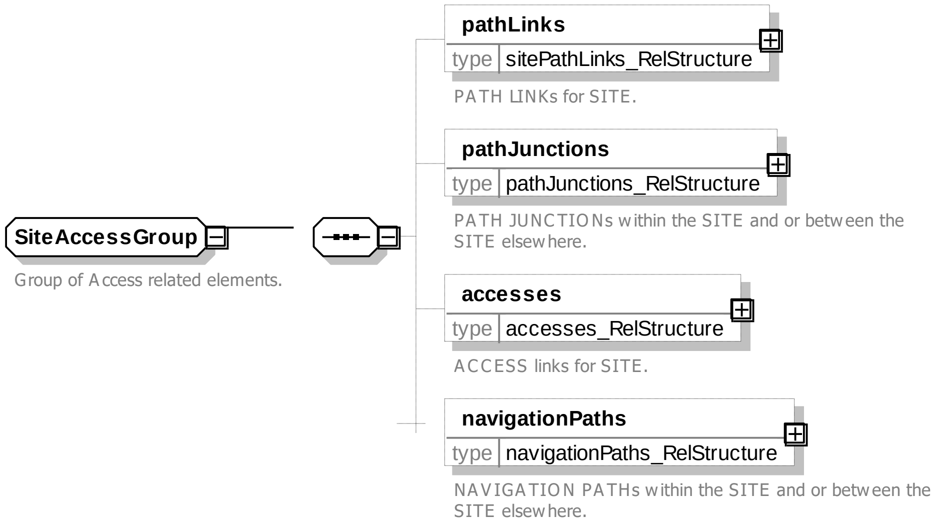


Figure 505 – *SiteAccessGroup – XSD*

8.4.4 Stop Place

8.4.4.1 STOP PLACE – Conceptual MODEL

The STOP PLACE model describes different aspects of a physical point of access to transport, such as a stop or station. For a complex interchange, such as a station, this includes all the component areas of the station; the entrances, concourses, platforms; the levels they are on, the paths through the station and the various types of equipment found in the station, such as ticket machines and lifts, barriers, signs and seating. It also allows detailed accessibility attributes to be recorded at both the element and the station level.

A STOP PLACE represents physical stop or station; that is an interchange, a pair of stops or a cluster of stops on a LINE. A STOP PLACE is a type of SITE. Note that a STOP PLACE is a distinct concept from the representation of the stop in a timetable – the SCHEDULED STOP POINT. The two can be connected using a STOP ASSIGNMENT.

The various spaces of which a STOP PLACE is comprised are described as different types of SITE COMPONENT specific to a STOP PLACE, such as platforms (QUAYs), and concourses (ACCESS SPACEs), etc. The physical point of access to transport is always a QUAY. ENTRANCEs describe the internal and external entrances to the STOP PLACE.

— QUAYs and ACCESS SPACES can be connected to each other using PATH LINKs.

- SITE and SITE COMPONENT inherit common properties from SITE ELEMENT, including ACCESSIBILITY characteristics, and the ability to specify ALTERNATIVE NAMEs, ACCESSIBILITY, PATH LINKs, CHECK CONSTRAINTs and EQUIPMENT, all of which are discussed separately. It is also possible to specify whether the component is indoors or outdoors, or with a gated area.

Furthermore:

- Specific labelled points on a QUAY can be identified as BOARDING POSITIONs, for example the positions to board Eurostar coaches, or the doorways points to an enclosed metro line like the TfL Jubilee Line.

- STOP PLACEs can be organised into a hierarchy so that clusters of transport interchanges, such as a paired rail and tube station, can be described.

- QUAYs can be nested; this allows one to represent composite platforms with two or more sides or named sections. One can thus journey plan to any level of detail (see later below for examples). Similarly ACCESS SPACEs can be nested – within another ACCESS SPACE.

- ENTRANCEs describe points at which a passenger can access a stop place, normally on foot – an ACCESS MODE can be used to identify other permitted modes of entry such as cycle or car. ENTRANCEs can be external, for example the main ENTRANCE, or internal, for example from an entrance concourse to a platform.

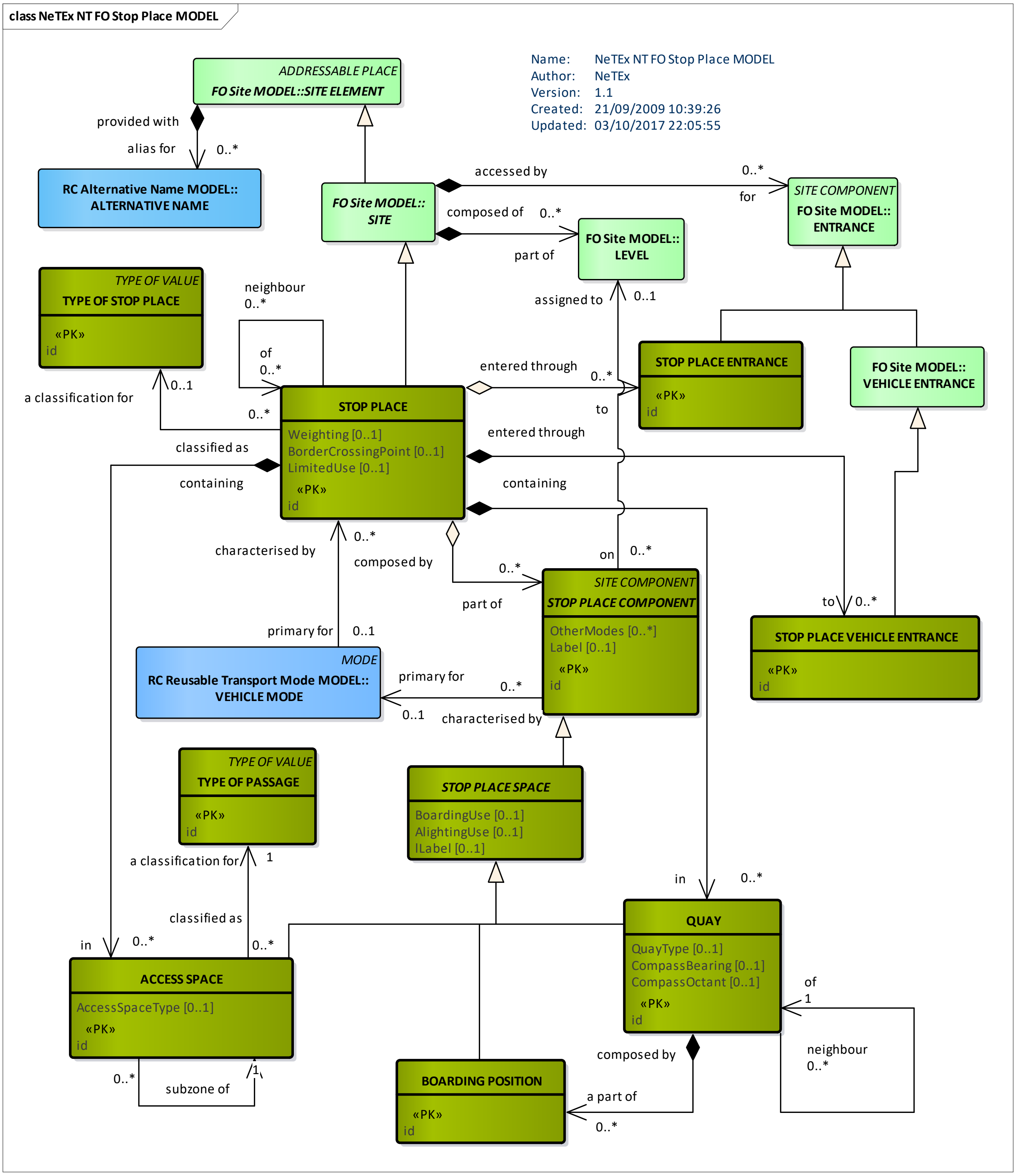


Figure 506 – Stop Place –Conceptual MODEL (UML)

8.4.4.2 Stop Place – Examples

The following table shows how *NeTEx* elements would be used to represent different types of stop.

Table 420 – Common *NeTEx* stop element combinations

	Stop	*NeTEx*	Comment
On street	Single bus stop	1 STOP PLACE + 1 QUAY	EQUIPMENT for stop furniture
	Pair of bus stops on a route bus stop	1 STOP PLACE + 2 QUAYs	
	On street bus cluster	1 STOP PLACE + n QUAYs	
	Hail & Ride Zone	1 STOP PLACE + 1 FLEXIBLE QUAY	
	FlexibleZone	1 STOP PLACE + 1 FLEXIBLE QUAY	Zone projection for flexible area
Off-street	Single mode rail station	1 STOP PLACE + n QUAYs + x ACCESS SPACES + y ENTRANCEs.	Use PATH LINKs + NAVIGATION PATHs for connectivity. EQUIPMENT
	Single mode metro station	1 STOP PLACE + n QUAYs + x ACCESS SPACES + y ENTRANCEs.	
	Bus or Coach station	1 STOP PLACE + n QUAYs + x ACCESS SPACES + y ENTRANCEs.	
	Airport	1 STOP PLACE + n QUAYs + x ACCESS SPACES + y ENTRANCEs.	
Multi modal interchange	Discrete places for each mode	As for single mode, Use PARENT STOP PLACE reference to link to main STOP AREA.	
	Shared use of platforms by different modes	1 STOP PLACE + n QUAYs + x ACCESS SPACES + y ENTRANCEs. Distinct SCHEDULED STOP POINTs for each mode, with STOP ASSIGNMENTs	See example

8.4.4.2.1 Examples of simple on-street Stop Places

In this section we illustrate some of the commonly found combinations.

The following figure shows an on street bus stop as a simple STOP PLACE with a single QUAY for a single direction. Might commonly be found at a terminus of a service.

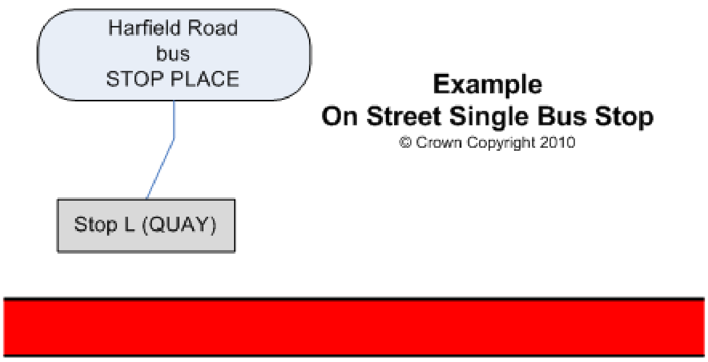


Figure 507 – Example of a single bus stop on street

The following figure shows an on street bus stop pair named '*St George's Road'* as a simple STOP PLACE with two QUAYs, one for each direction.

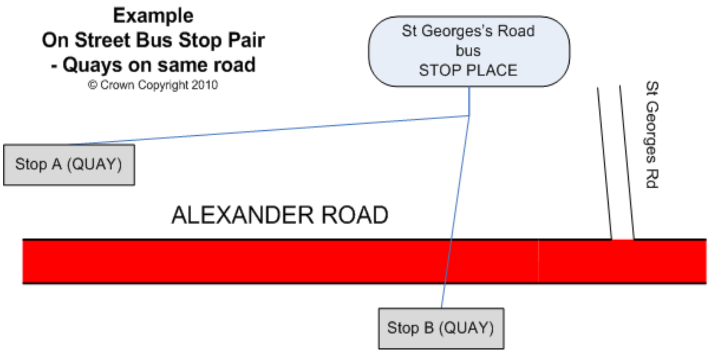


Figure 508 – Example pair of bus stops on street

The following figure shows an on street bus cluster as a simple STOP PLACE with four QUAYs.

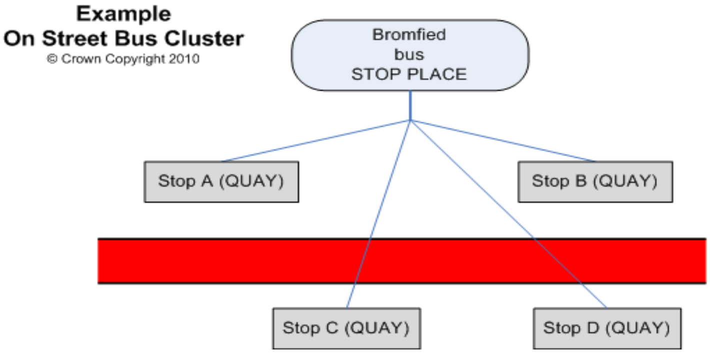


Figure 509 – Example bus cluster on street

8.4.4.2.2 Simple Examples of Stations

The following figure shows a simple rail station with two platforms connected by a barrow crossing represented by two QUAYs and various different types of ACCESS SPACE.

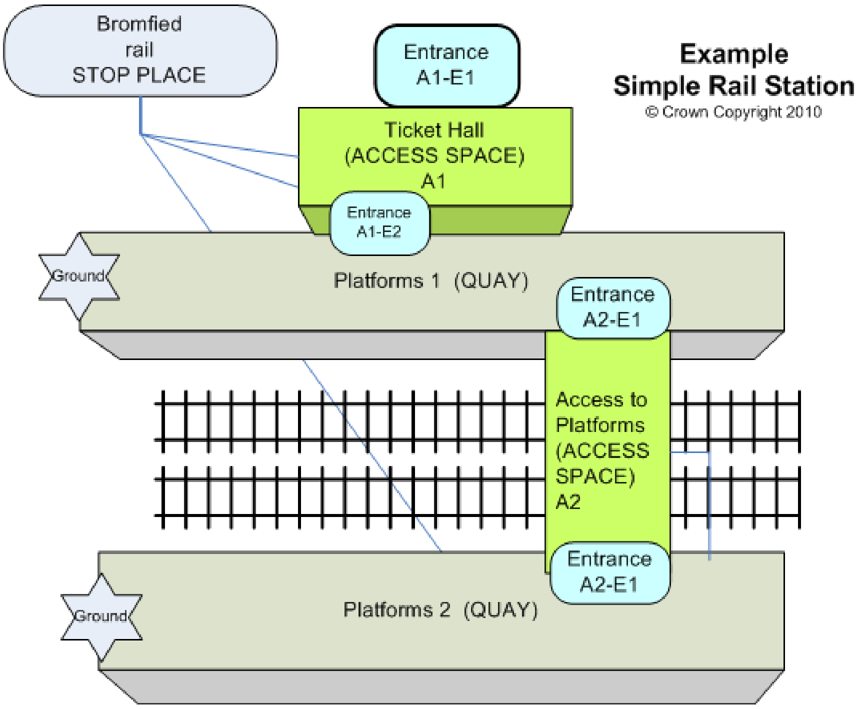


Figure 510 – Simple Rail Station Example – Barrow crossing

The following figure shows a simple rail station with two platforms connected by a bridge (or subway), represented by two QUAYs and various different types of ACCESS SPACE.

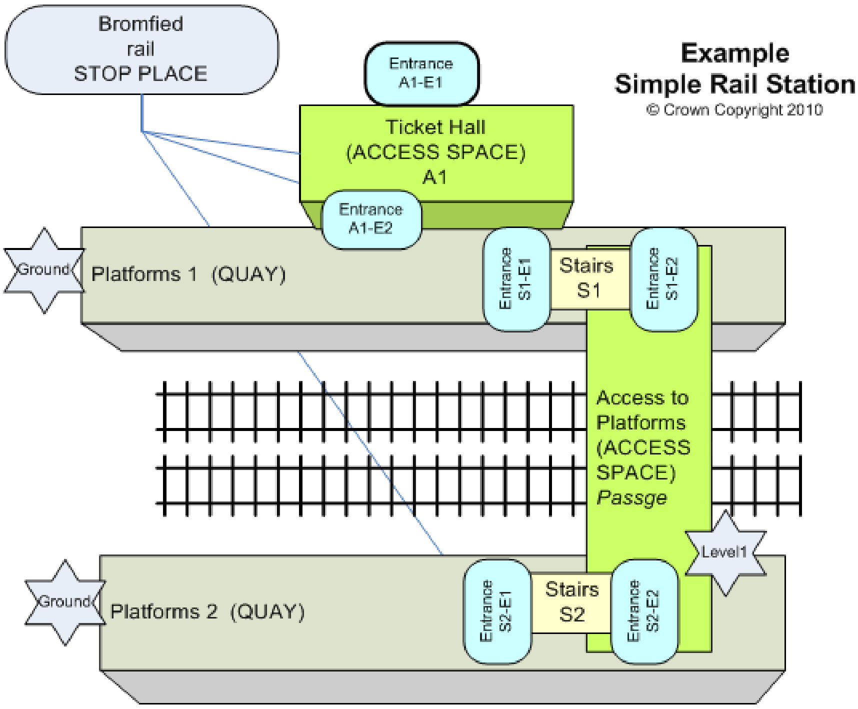


Figure 511 – Simple Rail Station Example – Crossing with stairs

The following figure shows a slightly more complicated example of rail station with four platforms connected by a bridge (or subway) and two ENTRANCEs on different LEVELs.

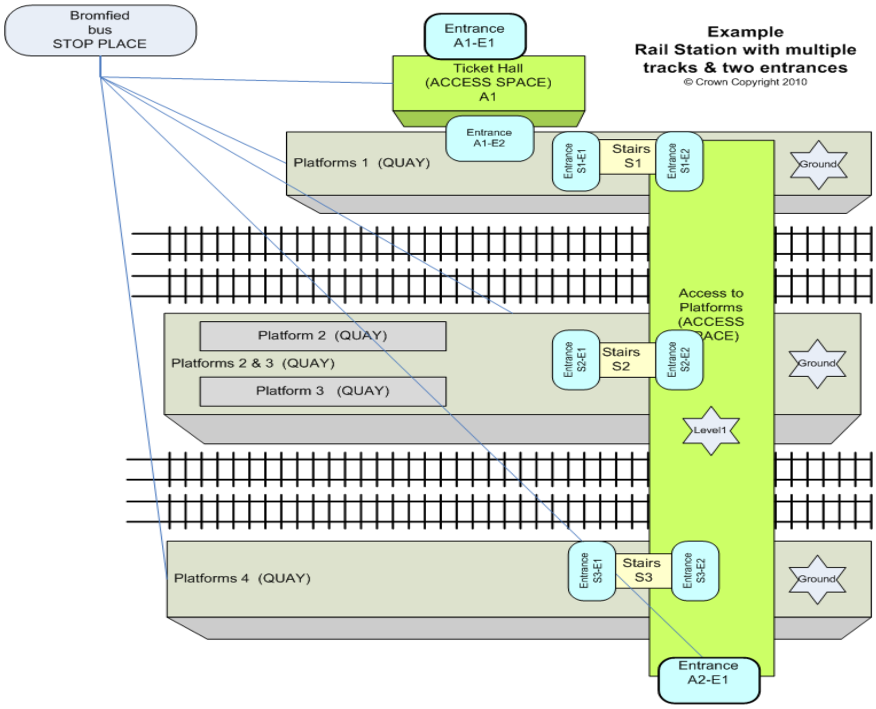


Figure 512 – Rail Station example with multiple platforms

8.4.4.2.3 Nesting Stop Places

Sometimes a complex SITE is made up of a number of different SITEs, for example a large rail STOP PLACE may contain a metro station as a child STOP PLACE and have associated STOP PLACEs for the stops of the bus routes that pass by it – See the following figure.

- There should be a separate STOP PLACE for each transport mode (but see discussion below of shared multimodal use of platforms).
- A separate STOP PLACE should be created if an area of a station can be referenced as a separate station by a timetable or other passenger information usage. For example "*St Pancras Domestic"* and "*St Pancras International*".
- There should be a separate STOP PLACE for each pair of bus or tram stops (or isolated stop) on street.

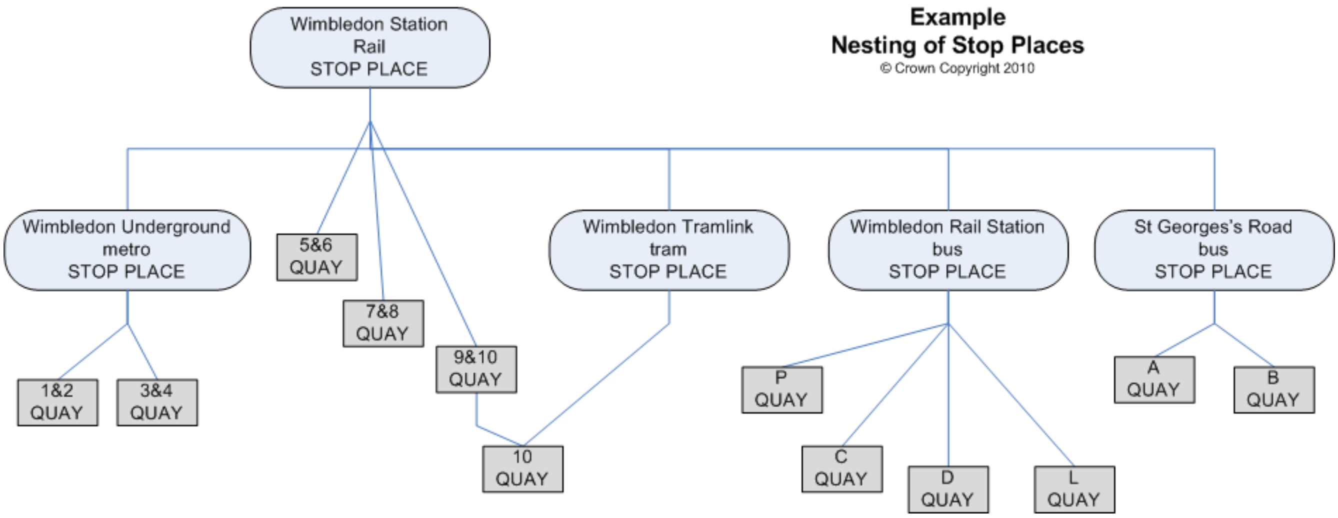


Figure 513 – Example Nesting of Stop Places

8.4.4.2.4 Nesting Quays / Platforms

Several arrangements of composite platforms are commonly found in stations, for example one sided, two sided, etc. The following figure illustrates common configurations.

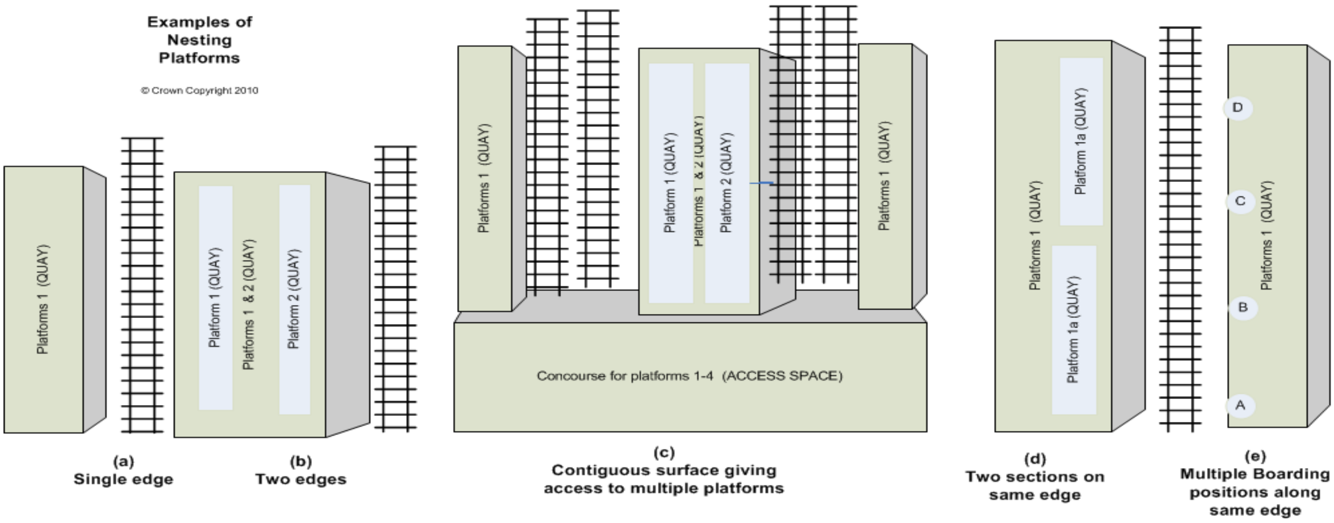


Figure 514 – Common QUAY configurations for station platforms

The following table summarises the common configurations along with a recommended representation using *NeTEx* elements. Further more:

- A nested QUAY is always physically contiguous with its parent and so has the same accessibility characteristics as it parents.
- Nested QUAYs should not be used to mark individual positions on a platform – BOARDING POSITIONs service this function.
- Nested QUAYs and ACCESS PLACES must always be on the same LEVEL as their parent.

Table 421 – Nested QUAY configurations.

Quay Arrangement	Description	Modelling in *NeTEx*
Single edged platform	A single physical platform with a track along a single side	A single QUAY.
Double edged platform	A single physical platform with tracks along both sides. Examples; *Wimbledon 4 & 5*	A parent QUAY with two nested child QUAYs for each side.
Multiple section same edge	A physical platform divided into sections on the same side, for example to indicate train sections that will go to different destinations. Examples: *Cambridge*	A parent QUAY with separate nested child QUAYs for each section. There may be BOARDING POSITIONs to mark points on the platform for each specific carriage.
Multiple sections both edges	A physical platform divided into sections on both sides.	A parent QUAY with nested child QUAYs for each side, and separate nested child QUAYs for each section.
Contiguous surface giving access to multiple platforms	A collection of platforms joined by a contiguous end section: usually found at a terminus or a large station that is a terminus or where some of the tracks terminate. Examples: *Kings Cross, Cambridge, Wimbledon District Line platforms 1-4*	An ACCESS SPACE for the end section and separate QUAYs for each platform as above. **Not** a parent QUAY for the whole contiguous area containing nested QUAYs for each platform.

8.4.4.2.5 Multimodal use of the same platform

Usually there will be a separate STOP PLACE for each transport mode at an interchange, each with its own QUAYs, and with distinct ENTRANCEs. Sometimes however STOP PLACEs for different modes may be intermingled, with the same platform being shared; for example between rail, tram or metro, or between bus and coach.

For example, in the Wimbledon example given earlier there are distinct *Wimbledon Rail Station* and *Wimbledon Tram link* STOP PLACEs, even though they both share a platform 10.

— Where platforms are shared between modes, a single definition of the platform i.e. QUAY can be made. The STOP PLACE for the major mode (e.g. *rail*) can contain the QUAY definition. Two alternative approaches are possible.

  (i) Create a separate STOP PLACE for the additional mode; the STOP PLACE mode can reference the QUAY definition.

  (ii) Simply specify multiple modes for the STOP PLACE and the QUAY (e.g. *rail, metro*).

— The rail STOP PLACE can state tram as another mode and vice versa.

— There will typically be separate SCHEDULED STOP POINTs for the tram and for the rail timetables.

— In addition there can be separate PASSENGER STOP ASSIGNMENTs to assign different SCHEDULED STOP POINTs for each mode to the same QUAY.

8.4.4.2.6 Representing an Entrance between two adjacent spaces

Where there is an ENTRANCE between two adjacent spaces (e.g. ACCESS SPACEs or QUAYs), it is not necessary to create two separate ENTRANCE elements. Instead a single ENTRANCE can be created and shared between the two spaces.

— ENTRANCEs are normally specified as properties of the overall SITE i.e. STOP PLACE.

— ENTRANCEs may be additionally referenced by the ACCESS SPACES and QUAYs which use them. In the example XML fragment below, Entrance '9100WIMBLDN5n6_EL1' is declared as a child of StopPlace '9100WIMBLDN', but also referenced explicitly by Quay '9100WIMBLDN5n6'.

NOTE Provided every ACCESS SPACE and QUAY references all of its own ENTRANCEs, there is sufficient data to infer a basic topology of a SITE even without the specification of PATH LINKs.

8.4.4.3 Stop Place – Physical Model

8.4.4.3.1 Stop Place Overview – Physical Model

The following figure gives an overveiw of the physical model for a STOP PLACE.

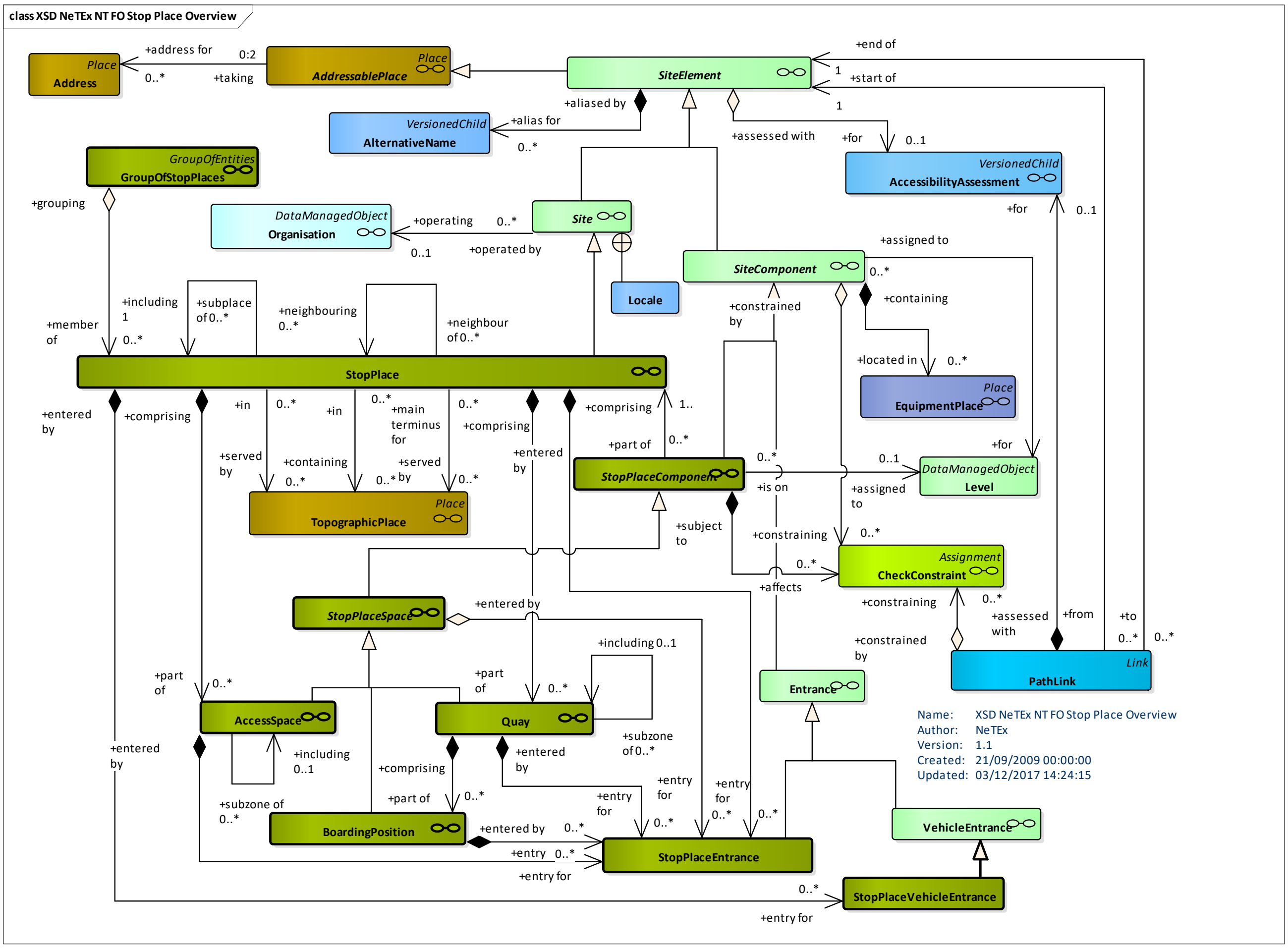


Figure 515 – Stop Place – Physical Model - Overview (UML)

8.4.4.3.2 Stop Place Details – Physical Model

The following figure shows detailed attributes of the STOP PLACE model. A GROUP OF STOP PLACEs aggregates a set of related STOP PLACEs for an arbitrary purpose.

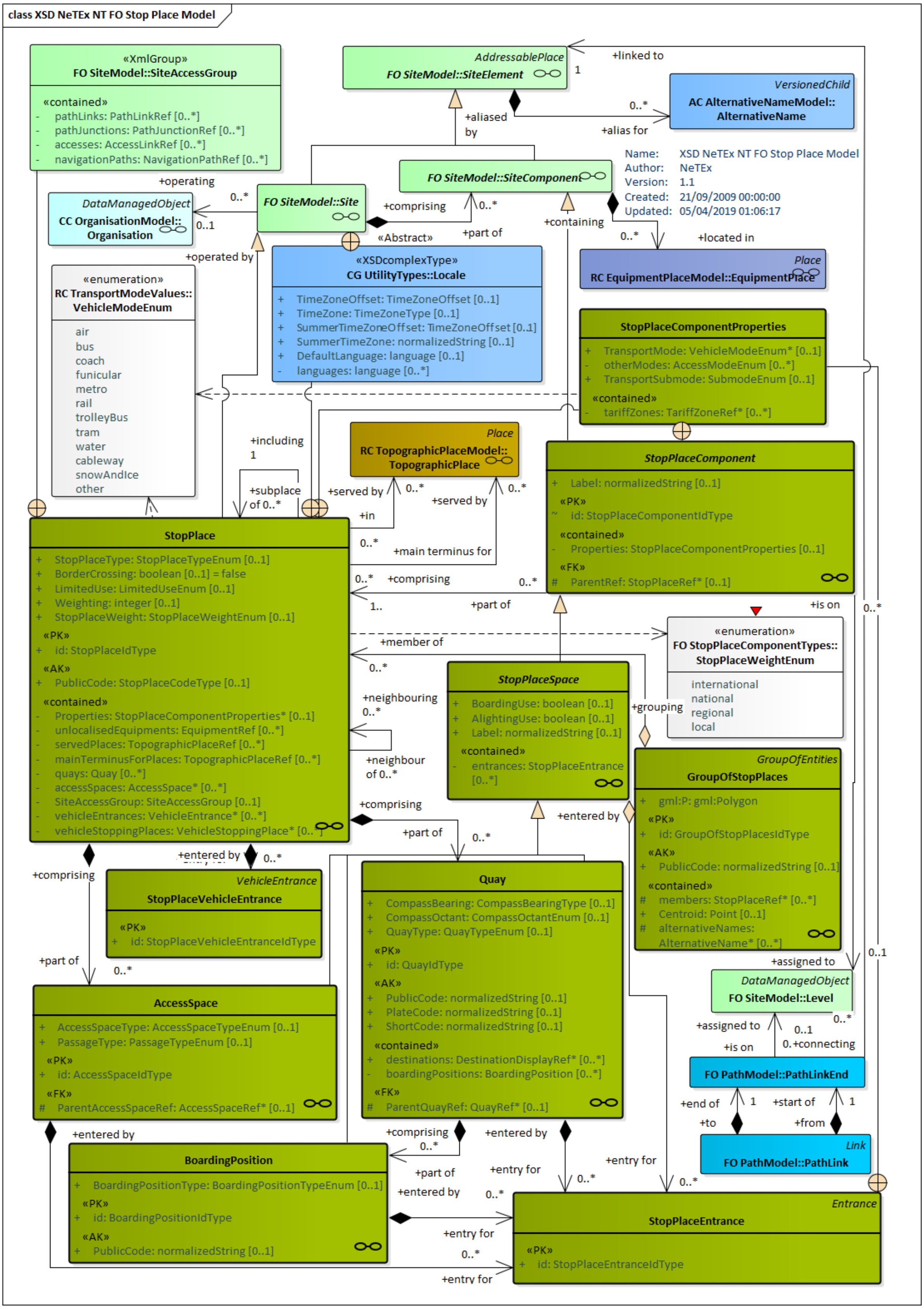


Figure 516 – Stop Place – Physical Model – Details (UML)

8.4.4.3.3 Stop Place Classifications – Physical Model

Mostt of the stop place components can be classified using a standardise set values. The following diagram shows the supported avlies. In addition TYPE OF STOP PLACE, etc can be used to apply other arbitrary classifcation schemes.

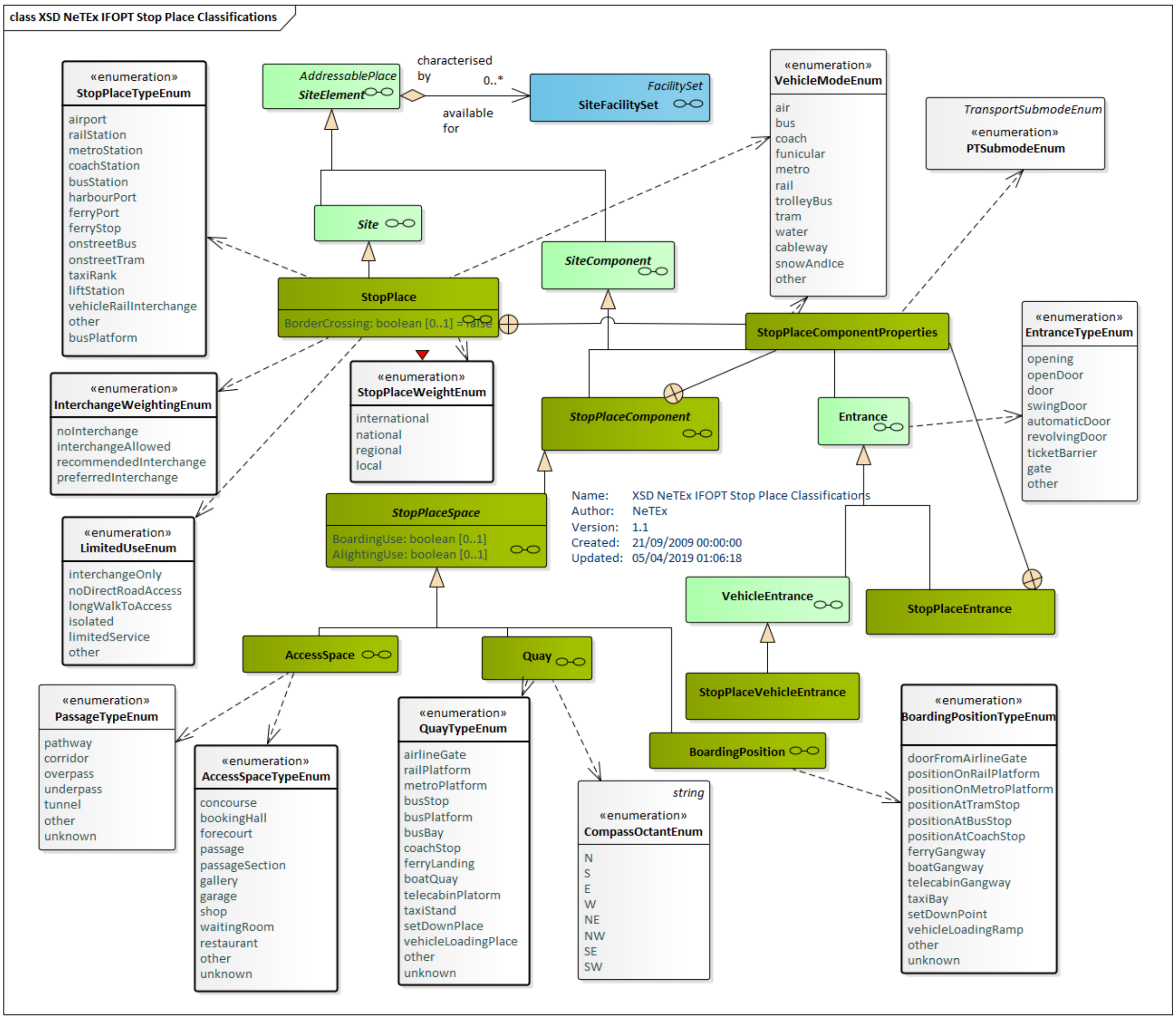


Figure 517 – Stop Place – Physical Model – Classification Values (UML)

8.4.4.4 Stop Place – Attributes and XSD

8.4.4.4.1 StopPlace – Model Element

A place comprising one or more locations where vehicles may stop and where passengers may board or leave vehicles or prepare their trip. A STOP PLACE will usually have one or more well-known names.

Table 422 – *StopPlace* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Site*</u>	::>	STOP PLACE inherits from SITE.

«PK»	***id***	*StopPlaceIdType*	1:1	Identifier of a STOP PLACE.
XGRP	***StopPlaceGroup***	***xmlGroup***	0:1	Elemenst describing a STOP PLACE.

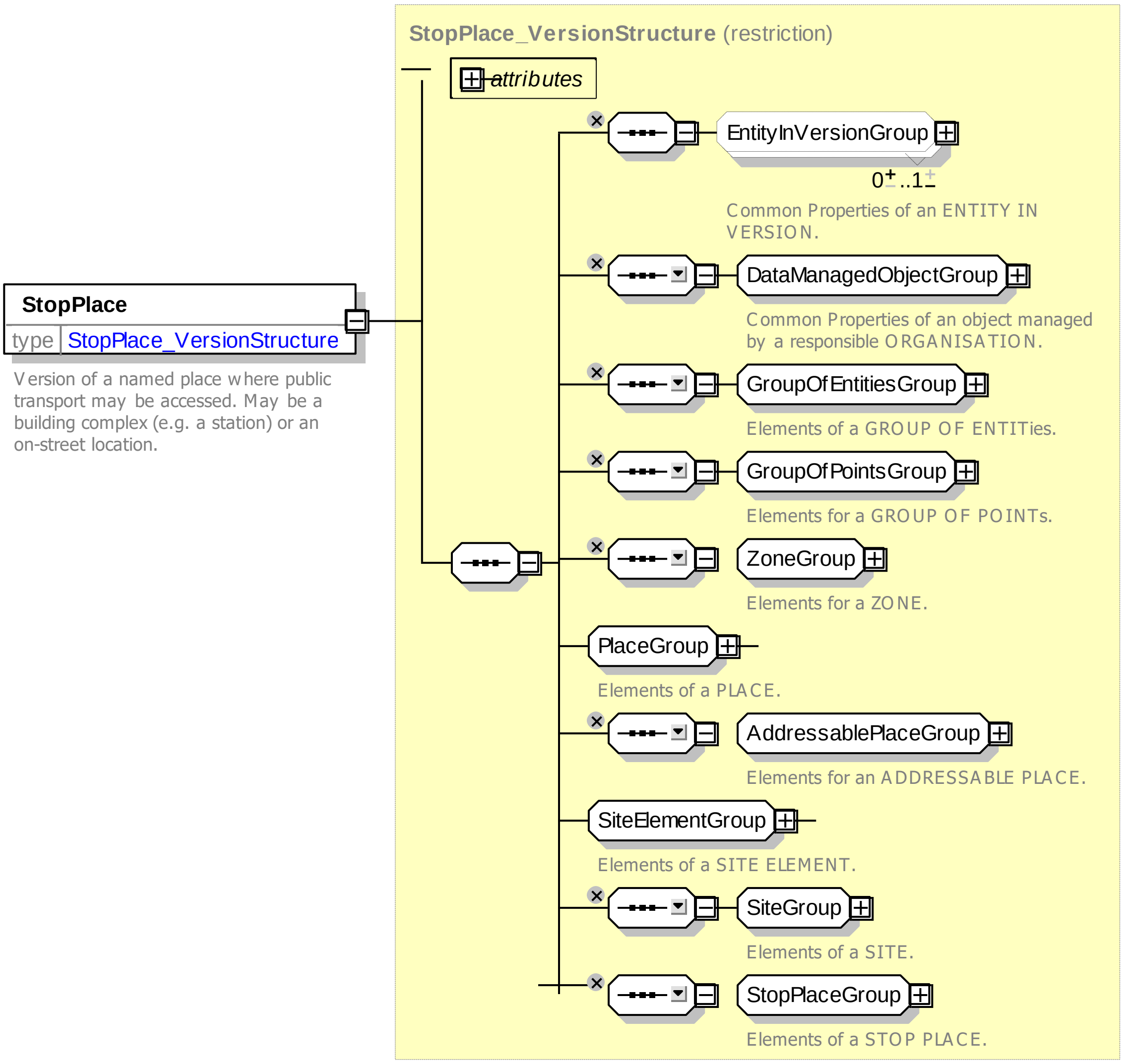


Figure 518 – *StopPlace – XSD*

8.4.4.4.1.1 StopPlaceGroup – XML Group

Elements describing a STOP PLACE.

Table 423 – *StopPlaceGroup* – Group

Classification	Name	Type	Cardinality	Description
«AK»	***PublicCode***	*StopPlaceCodeType*	0:1	Code used to identify a STOP PLACE to the public as an alternative to a name.
XGRP	***StopPlaceComponent-PropertyGroup***	***xmlGroup***	0:1	Common elements relating to all STOP PLACE components.

XGRP	***StopPlaceProperty-Group***	***xmlGroup***	0:1	Common elements desribing STOP PLACE properties.
XGRP	***StopPlace-TopographicalGroup***	***xmlGroup***	0:1	Elements relating to TOPOGRAPHICAL PLACEs that the STOP PLACE serves.
«enum»	***Weighting***	*InterchangeWeighting Enum*	0:1	STOP PLACEs can be classified for their relative desirability (weighting) as an interchange. See allowed values below.
«enum»	***StopPlaceWeight***	*StopPlaceWeightEnum*	0:1	Type of expected INTERCHANGE at a STOP PLACE for use in journey planners and also for possible legal classification. See allowed values below. +v1.1
«cntd»	***quays***	*Quay*	0:*	The QUAYs contained in the STOP PLACE, that is platforms, jetties, bays, taxi ranks, and other points of physical access to VEHICLEs.
«cntd»	***accessSpaces***	*AccessSpace*	0:*	ACCESS SPACEs within the STOP PLACE, i.e. STOP PLACE COMPONENTs that are not QUAYs, BOARDING POSITIONS, or ENTRANCEs.
XGRP	***SiteAccessGroup***	***xmlGroup***	0:1	Common elements relating to SITE ACCESS; PATH LINKs, PATH JUNCTIONs, ACCESS LINKs, NAVIGATION PATHs.
«cntd»	***vehicleStoppingPlaces***	*VehicleStoppingPlace*	0:*	VEHICLE STOPPING PLACEs within the STOP PLACE. See VEHICLE STOPPING Model later ater below

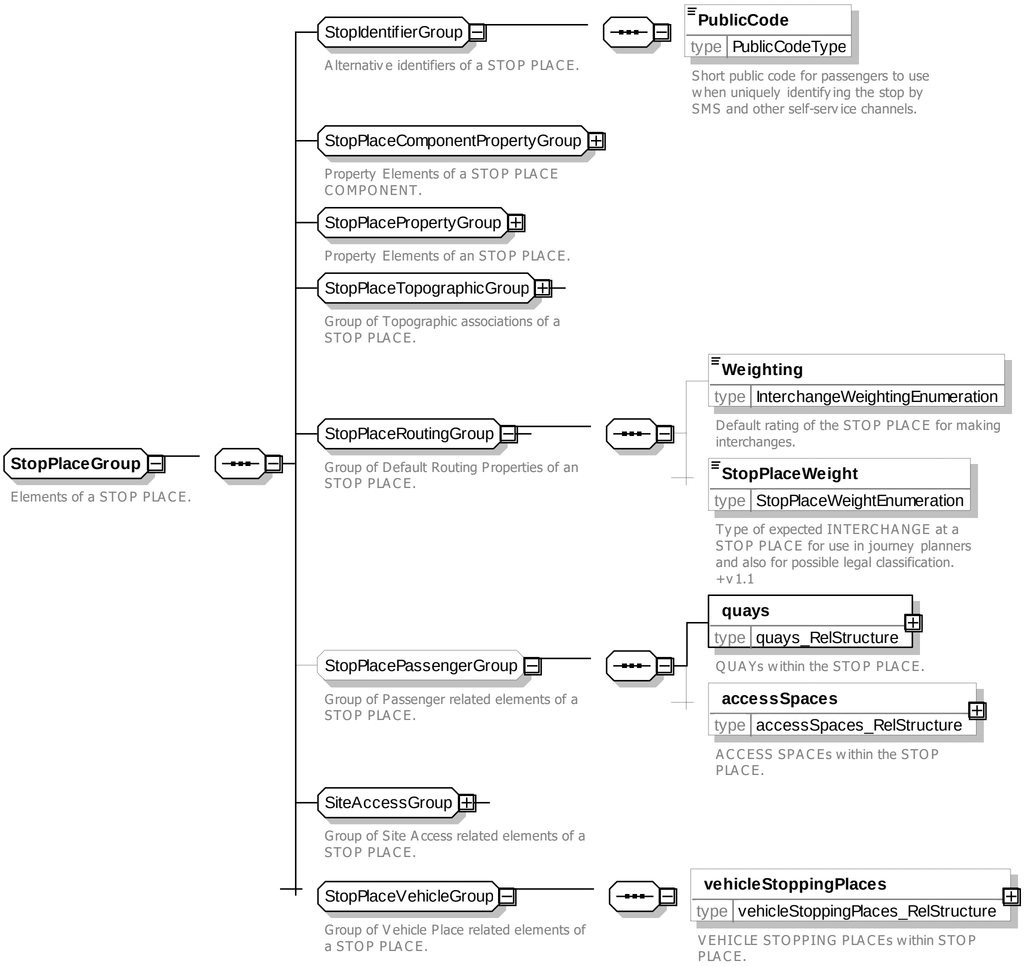


Figure 519 – *StopPlaceGroup – XSD*

Interchange Weighting – Allowed values

STOP PLACEs can be classified for their relative desirability as an interchange spot. The following table shows the allowed values for ***InterchangeWeighting*** (*InterchangeWeightingEnumeration*).

Table 424 – *InterchangeWeighting* – Allowed values

Value	Description
preferredInterchange	Interchange is a preferred point for transfers by Journey Planners
recommendedInterchange	Interchange is recommended for use by Journey planners
interchangeAllowed	Interchange may be used for transfers by Journey Planners but is not desirable
noInterchange	No Interchange should be made by Journey Planners

StopPlaceWeight – Allowed values

STOP PLACEs can be classified for their relative importance. The following table shows the allowed values for ***StopPlaceWeight*** (*StopPlaceWeightEnumeration*).

Table 425 – *StopPlaceWeight* – Allowed values

Value	Description
international	Place for international transfers.
national	Place for long distance national transfers.
regional	Place for regional transfers.
local	Place for local transfers.

8.4.4.4.1.2 StopPlaceComponentPropertyGroup – XML Group

Common elements relating to all STOP PLACE components.

Table 426 – *StopPlaceComponentGroup* – Group

Classification	Name	Type	Cardinality	Description
«enum»	***TransportMode***	*VehicleModeEnum*	0:1	The main TRANSPORT MODE of the STOP PLACE. See TRANSPORT MODE for allowed values.
«enum»	***submode***	*TransportSubmode-Enum*	0:1	Submode enumeration associated with MODE.
«cntd»	***OtherTransport-Modes***	*VehicleModeEnum*	0:*	The other TRANSPORT MODE available the STOP PLACE.
«cntd»	***tariffZones***	*TariffZoneRef*	0:*	The TARIFF ZONES associated with the STOP PLACE.

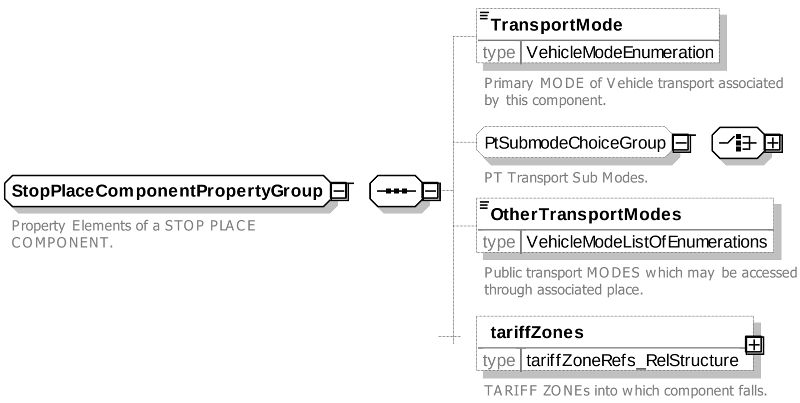


Figure 520 – *StopPlaceComponentPropertyGroup – XSD*

8.4.4.4.1.3 *StopPlacePropertyGroup* – XML Group

Elements describing specific properties of a STOP PLACE.

Table 427 – *StopPlacePropertyGroup* – Group

Classification	Name	Type	Cardinality	Description
«enum»	***StopPlaceType***	*StopPlaceTypeEnum*	1:1	The type of the STOP PLACE. See allowed values below.
	BorderCrossing	*xsd:boolean*	0:1	Whether STOP PLACE is a border crossing.
«cntd»	***unlocalised-Equipments***	*(EquipmentRef) | (Equipment)*	0:*	Equipment available at STOP PLACEs but not specifically located.

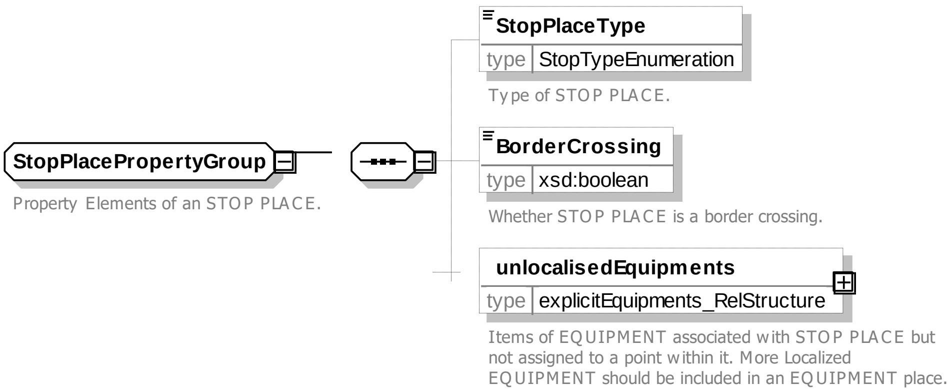


Figure 521 – *StopPlacePropertyGroup – XSD*

Classifying Stop Places – Allowed values

STOP PLACEs can be given a type showing the allowed values for ***StopPlace*** (*StopPlaceTypeEnumeration)*

Table 428 – *StopPlaceType* – Allowed values.

Value	Description
onstreetBus	On street Bus Stop
onstreetTram	On street Tram Stop
airport	Airport
railStation	Rail Station
metroStation	Metro Station
coachStation	Coach Station
ferryPort	Ferry Port
harbourPort	Harbour Port
ferryStop	Ferry Stop
liftStation	Lift, Ski Lift, etc
tramStation	Tram Station
vehicleRailInterchange	Motorrail access point +v1.1
other	Other

Use restrictions– Allowed values

STOP PLACEs can be classified for restrictions on use. These can be used to filter or annotate the stop in Journey Planners and other applications. The following table shows the allowed values for ***LimitedUse*** (*LimitedUseTypeEnumeration*).

Table 429 – *LimitedUse* – Allowed values

Value	Description
interchangeOnly	Stop may only be used for interchange, not for entrance or exit.
noDirectRoadAccess	Stop may not be reached from road by a paved path.
longWalkToAccess	Stop may only be accessed by a long (200m) walk from road.
isolated	Stop is an island or ferry stop that does not connect to road network.
limitedService	Stop has a very limited service.

8.4.4.4.1.4 StopPlaceTopographicGroup – XML Group

Elements describing topographic aspects of STOP PLACE.

Table 430 – *StopPlaceTopographicGroup* – Group

Classification	Name	Type	Cardinality	Description
«cntd»	***servedPlaces***	*AccessRef*	0:*	TOPOGRAPHICAL PLACEs that the STOP PLACE serves.
«cntd»	***mainTerminusFor-Places***	*TopographicalPlace-Ref*	0:*	TOPOGRAPHICAL PLACEs for which the STOP PLACE is a main terminus. Only certain stations will be deemed the main STOP PLACEs points. For example London has many rail stations but only some are main line termini. Geographic containment is not necessarily implied. For example London Gatwick and London Stansted airports are not in London, but are designated airports for London. Norwich station is not in Norwich, etc.
«enum»	***LimitedUse***	*LimitedUseEnum*	0:1	Categorisation of the stop as having topographic limitations which may affect its use in journey planners. See allowed values below.

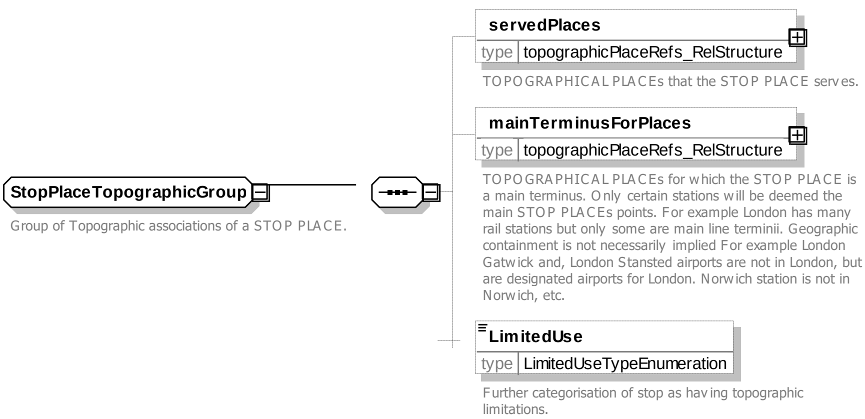


Figure 522 – *StopPlaceTopographicGroup – XSD*

8.4.4.4.2 StopPlaceComponent

An abstract element of a STOP PLACE describing part of its structure. STOP PLACE COMPONENTs share common properties for data management, accessibility and other features.

Table 431 – *StopPlaceComponent* – Element

Classifi cation	Name	Type	Cardina lity	Description
::>	*::>*	*SiteComponent*	*::>*	STOP PLACE COMPONENT inherits from SITE COMPONENT.
«PK»	***id***	*StopPlaceComponen-tIdType*	1:1	Identifier of STOP PLACE COMPONENT.
«enum	***TransportMode***	*VehicleModeEnum*	0:1	TRANSPORT MODE for STOP PLACE COMPONENT.
«enum	***otherTransport-Modes***	*VehicleModeEnum*	0:*	Other TRANSPORT MODEs for STOP PLACE COMPONENT.
«cntd	***tariffZones***	*TariffZoneRef*	0:*	TARIFF ZONEs for STOP PLACE COMPONENT.

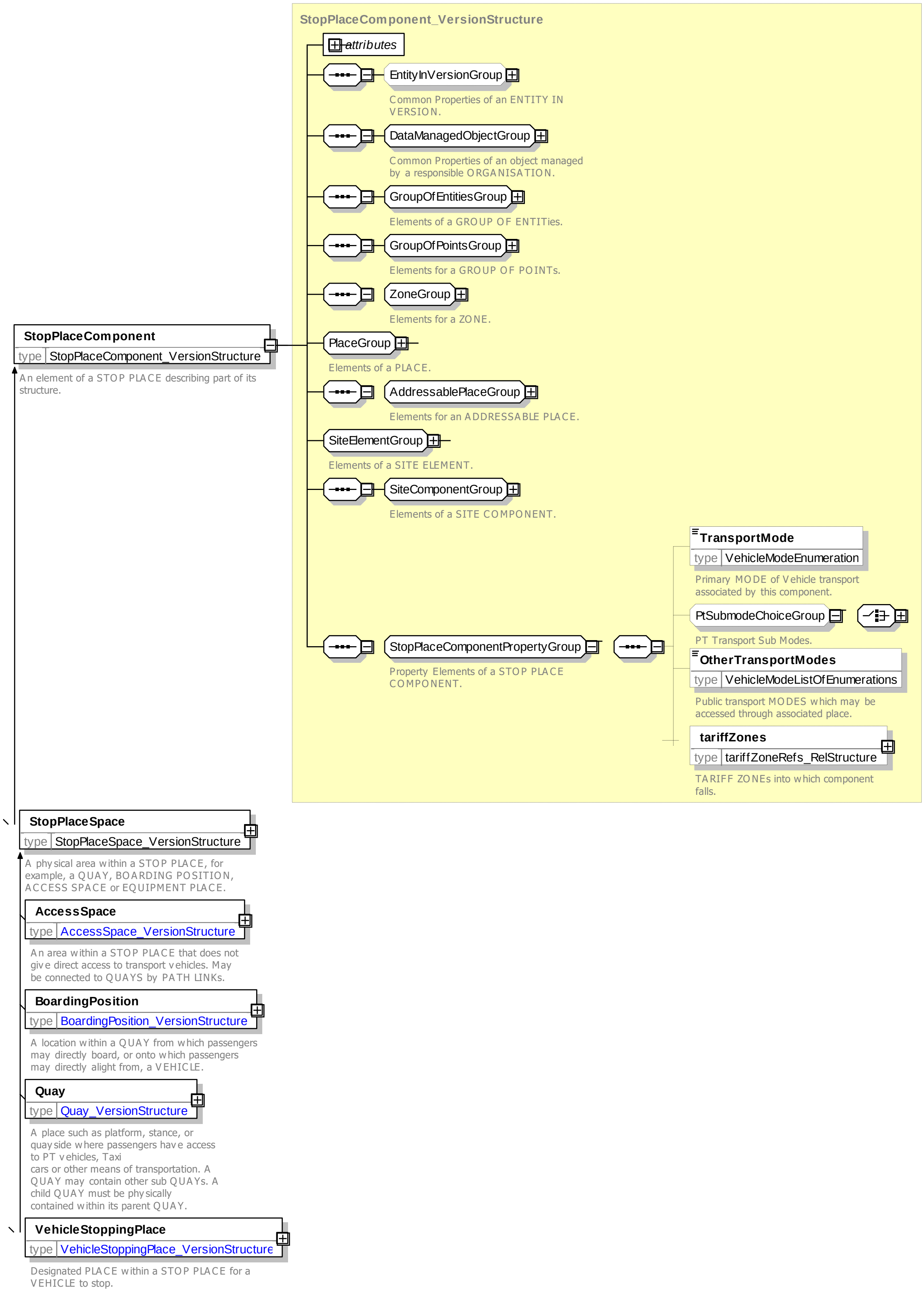


Figure 523 – *StopPlaceComponent – XSD*

8.4.4.4.3 StopPlaceSpace – Model Element

A physical area within a STOP PLACE, for example, a QUAY, BOARDING POSITION, ACCESS SPACE or EQUIPMENT PLACE.

Table 432 – *StopPlaceSpace* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SitopPlaceComponent*	*::>*	STOP PLACE SPACE inherits from STOP PLACE COMPONENT.
	BoardingUse	*xsd:boolean*	0:1	Whether Passengers may use the component for Boarding vehicle transport.
	AlightingUse	*xsd:boolean*	0:1	Whether Passengers may use the component when Alighting from vehicle transport.
	Label	*xsd:normalizedString*	0:1	Local Label given to Component, e.g. a Point Letter on a stop.
«cntd»	***entrances***	*StopPlaceEntrance*	0:*	ENTRANCES to STOP PLACE COMPONENT.

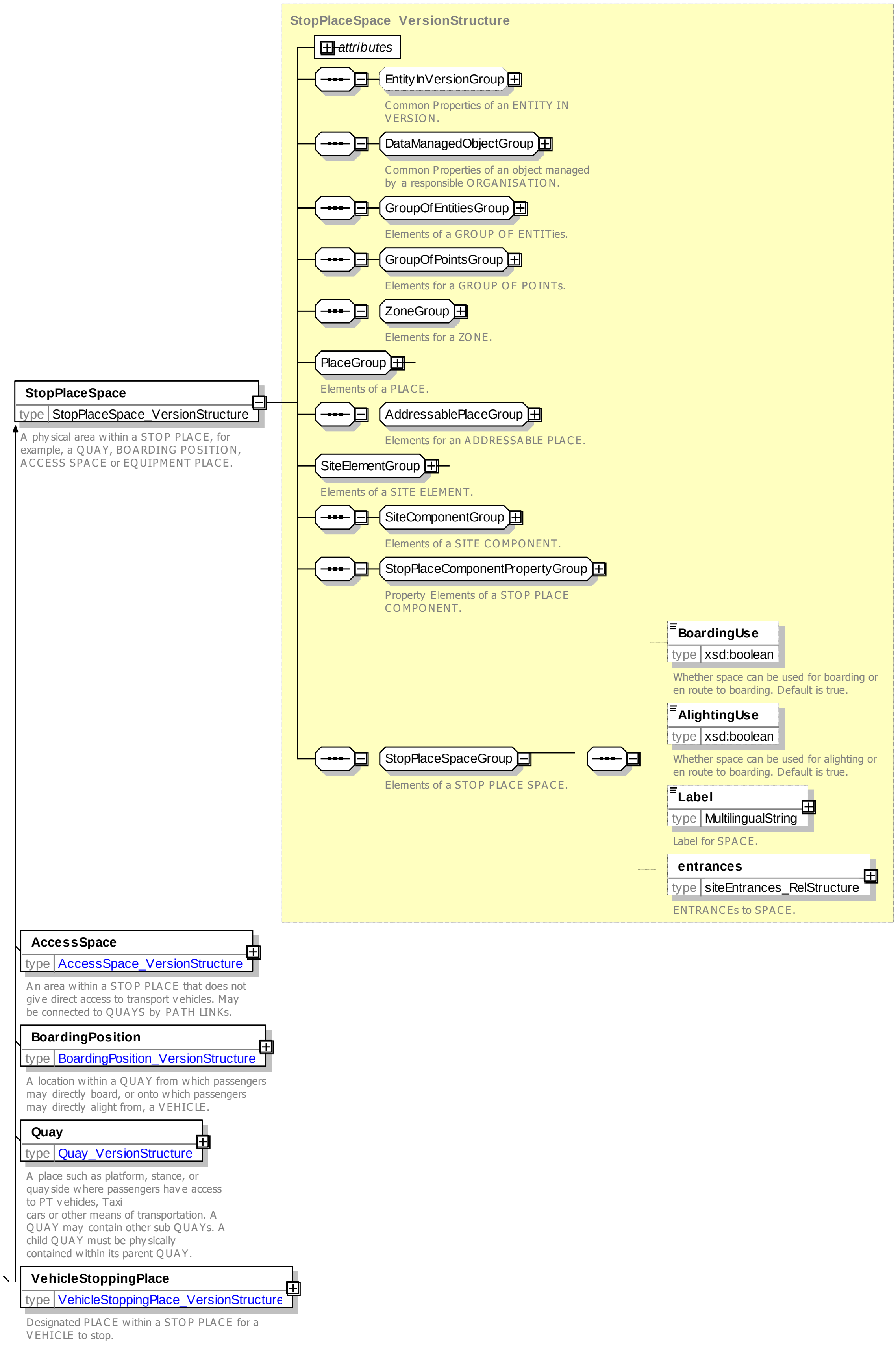


Figure 524 – *StopPlaceSpace – XSD*

8.4.4.4.4 StopPlaceEntrance – Model Element

A physical entrance or exit to/from a STOP PLACE for a Passenger. May be a door, barrier, gate or other recognizable point of access.

Table 433 – *StopPlaceEntrance* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SiteEntrance*	*::>*	STOP PLACE ENTRANCE. inherits from SITE ENTRANCE.
«PK»	***id***	*StopPlaceEntranceIdType*	1:1	Identifier of STOP PLACE ENTRANCE.
XGRP	S***topPlaceComponentGroup***	***xmlGroup***	0:1	Common properties of a STOP PLACE COMPONENT – See earlier.

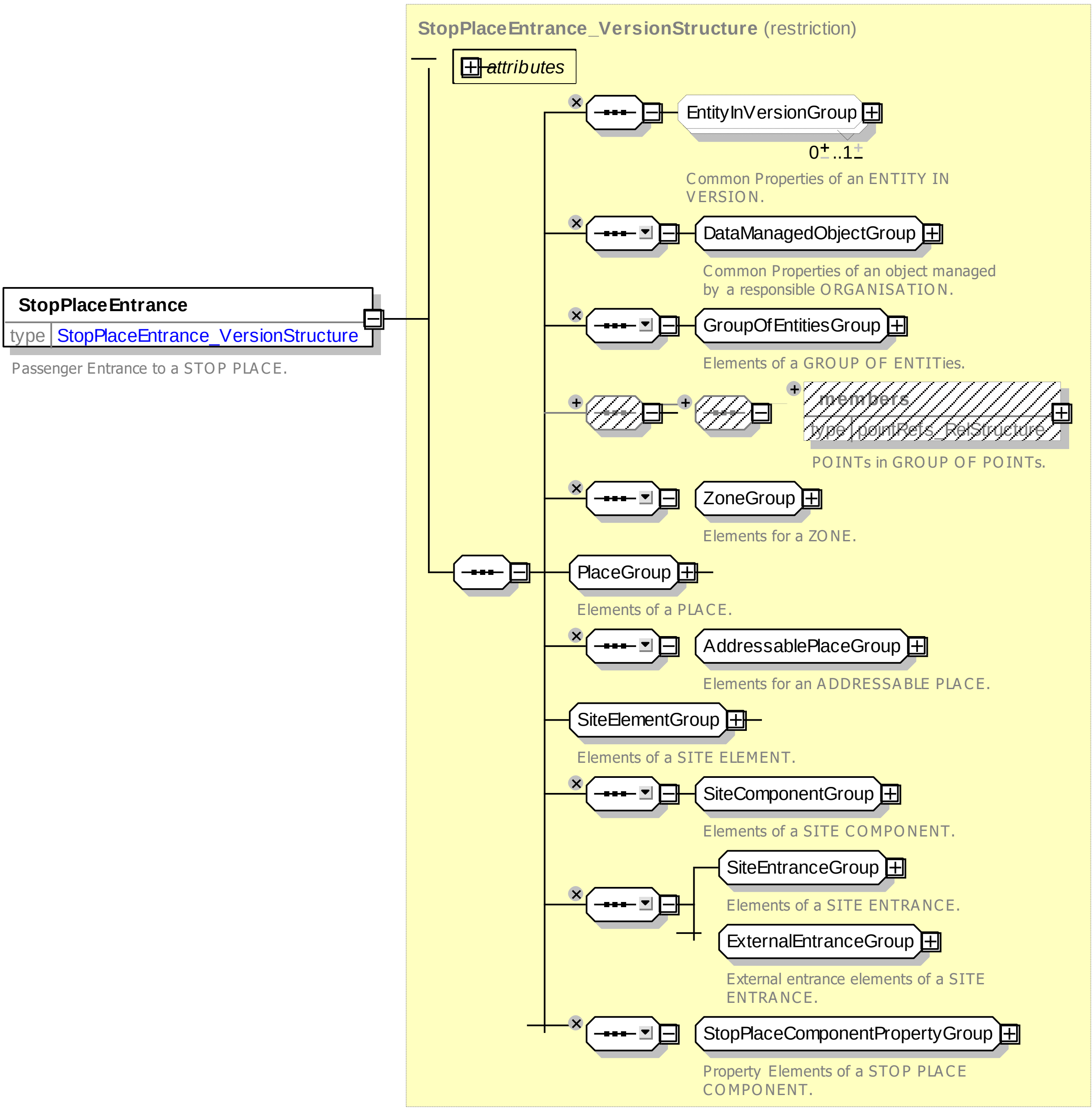


Figure 525 – *StopPlaceEntrance – XSD*

8.4.4.4.5 StopPlaceVehicleEntrance – Model Element

A physical entrance or exit to/from a STOP PLACE for a Vehicle.

Table 434 – *VehicleEntrance* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*VehicleEntrance*	*::>*	STOP PLACE VEHICLE ENTRANCE inherits from VEHICLE ENTRANCE.

«PK»	***id***	*StopPlaceVehicleEntrance-IdType*	1:1	Identifier of STOP PLACE VEHICLE ENTRANCE.
XGRP	S***topPlaceComponentGroup***	***xmlGroup***	0:1	Common properties of a STOP PLACE COMPONENT – See earlier.

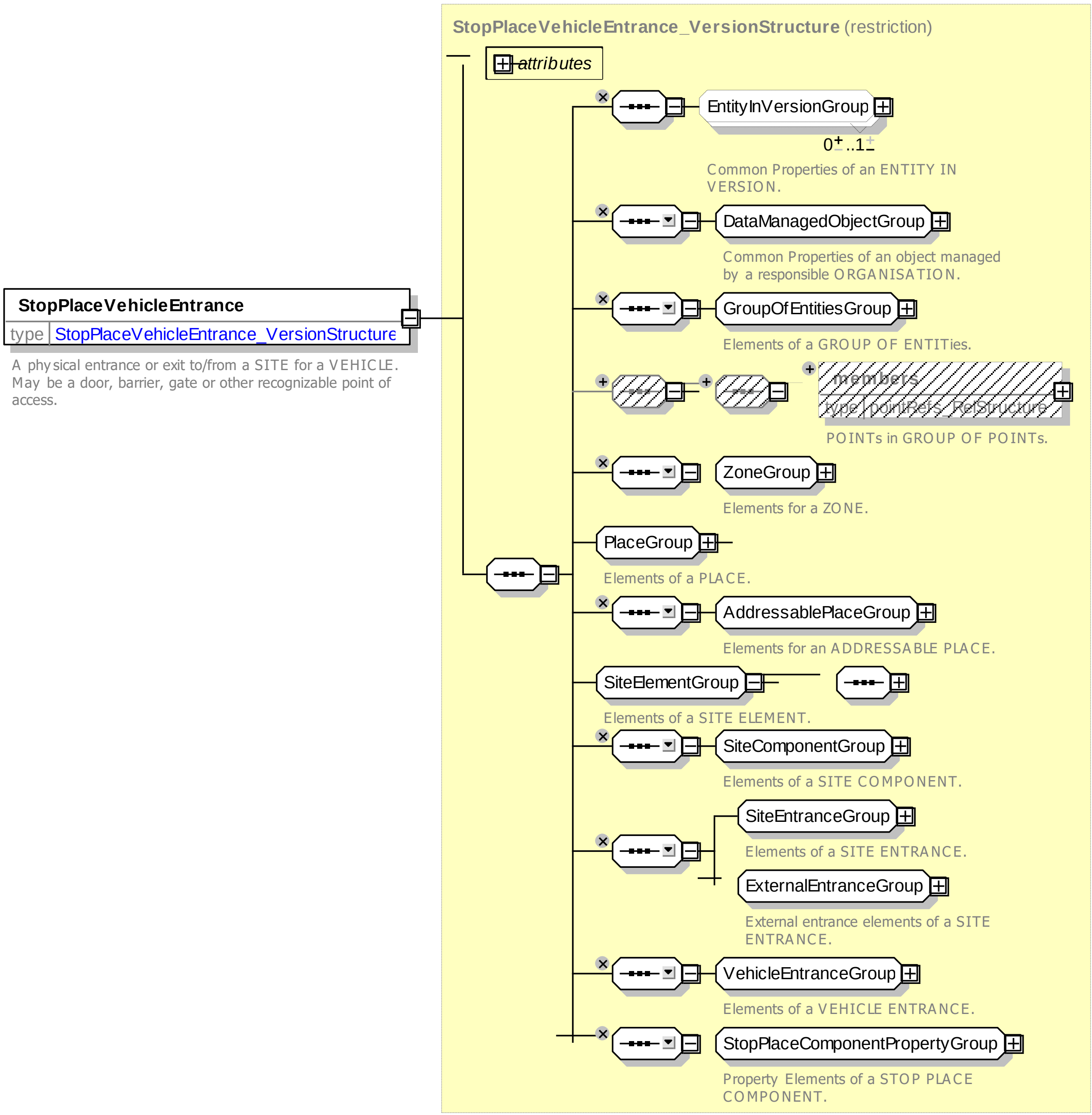


Figure 526 – *StopPlaceVehicleEntrance – XSD*

8.4.4.4.6 Quay – Model Element

A place such as platform, stance, or quayside where passengers have access to PT vehicles, taxi, cars or other means of transportation. A QUAY may serve one or more VEHICLE STOPPING PLACEs and be associated with one or more STOP POINTs.

A QUAY may contain other sub QUAYs. A child QUAY must be physically contained within its parent QUAY.

Table 435 – *Quay* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*StopPlaceSpace*	::>	QUAY inherits from STOP PLACE SPACE.
«PK»	***id***	*QuayIdType*	1:1	Identifier of QUAY.
XGRP	***QuayGroup***	***xmlGroup***	0:1	Elements describing QUAY.

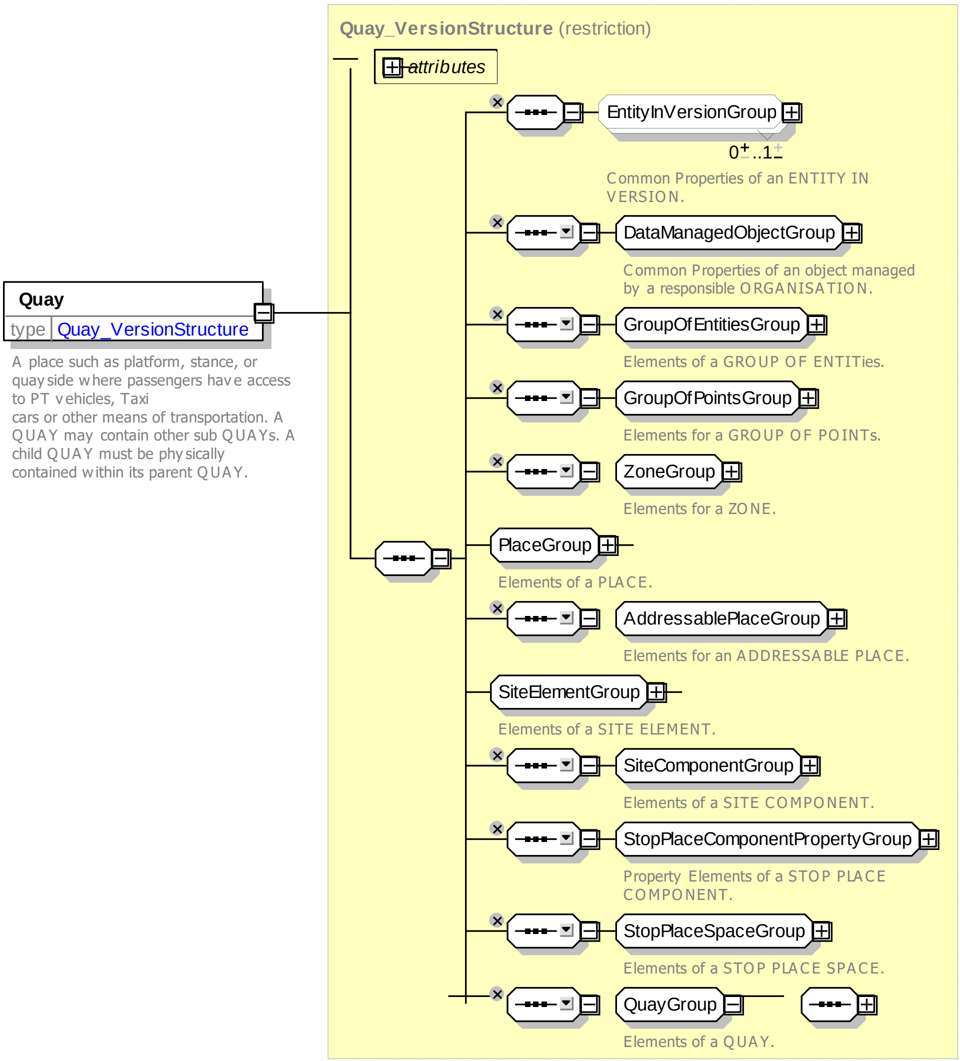


Figure 527 – *Quay* – XSD

8.4.4.4.6.1 QuayGroup – XML Group

Elements describing of a QUAY.

Table 436 – *QuayGroup* – Group

Classification	Name	Type	Cardinality	Description

«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Code use to identify QUAY to the public.
«AK»	***PlateCode***	*xsd:normalizedString*	0:1	Asset Code use to identify QUAY.
«AK»	***ShortCode***	*xsd:normalizedString*	0:1	Short Code use to identify QUAY for near band wireless.
«cntd»	***destinations***	*DestinationDisplayRef*	0:*	Destinations associated with on QUAY.
«enum»	***CompassBearing***	*CompassBearingType*	0:1	Bearing of street relative to QUAY in degrees.
«enum»	***CompassOctant***	*CompassOctantcntd*	0:1	Bearing of street relative to QUAY in compass quadrant. See utility types for allowed values.
«enum»	***QuayType***	*QuayTypeEnum*	0:1	Type of QUAY. See allowed values below.
«FK»	***ParentQuayRef***	*QuayRef*	0:1	Reference to parent of QUAY that wholly contains it.
«cntd»	***boardingPositions***	*BoardingPosition*	0:*	BOARDING POSITIONs within QUAY.

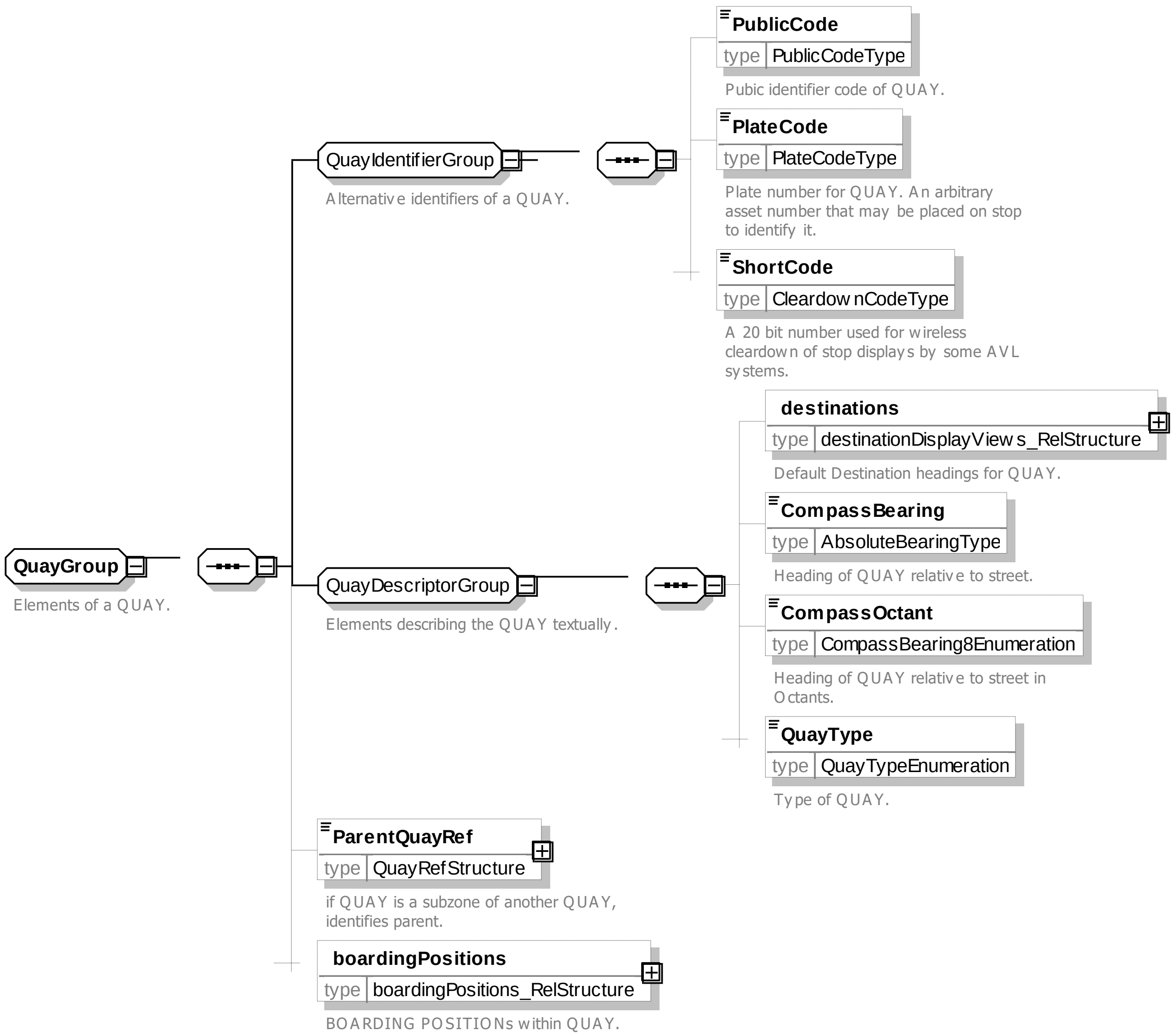


Figure 528 – *QuayDescriptorGroup – XSD*

Quay Type – Allowed values

The following table shows the allowed values for ***QuayType (****QuayTypeEnumeration)*.

Table 437 – *QuayType* – Allowed values

Value	Description
airlineGate	Airline Gate
railPlatform	Rail Platform
metroPlatform	Metro Platform
busStop	Bus Stop
busPlatform	Bus Platform +v1.1
tramPlatform	Tram Platform
boatQuay	Boat Quay
telecabinePlatform	Telecabine Platform
setDownPlace	Set Down or Pick up Place
coachStop	Coach Stop
busBay	Bus Bay
tramStop	Tram Stop
ferryLanding	Ferry Landing
taxiStand	Taxi Stand
vehicleLoadingPlace	Vehicle Loading Place
other	Other

8.4.4.4.7 BoardingPosition – Model Element

A location within a QUAY from which passengers may directly board, or onto which passengers may directly alight from, a VEHICLE.

Table 438 – *BoardingPosition* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*StopPlaceSpace*	*::>*	BOARDING POSITION inherits from STOP PLACE SPACE.
«PK»	***id***	*BoardingPositionIdType*	1:1	Identifier of BOARDING POSITION.
XGRP	***BoardingPosition Group***	***xmlGroup***	0:1	Elements describing BOARDING POSITION.

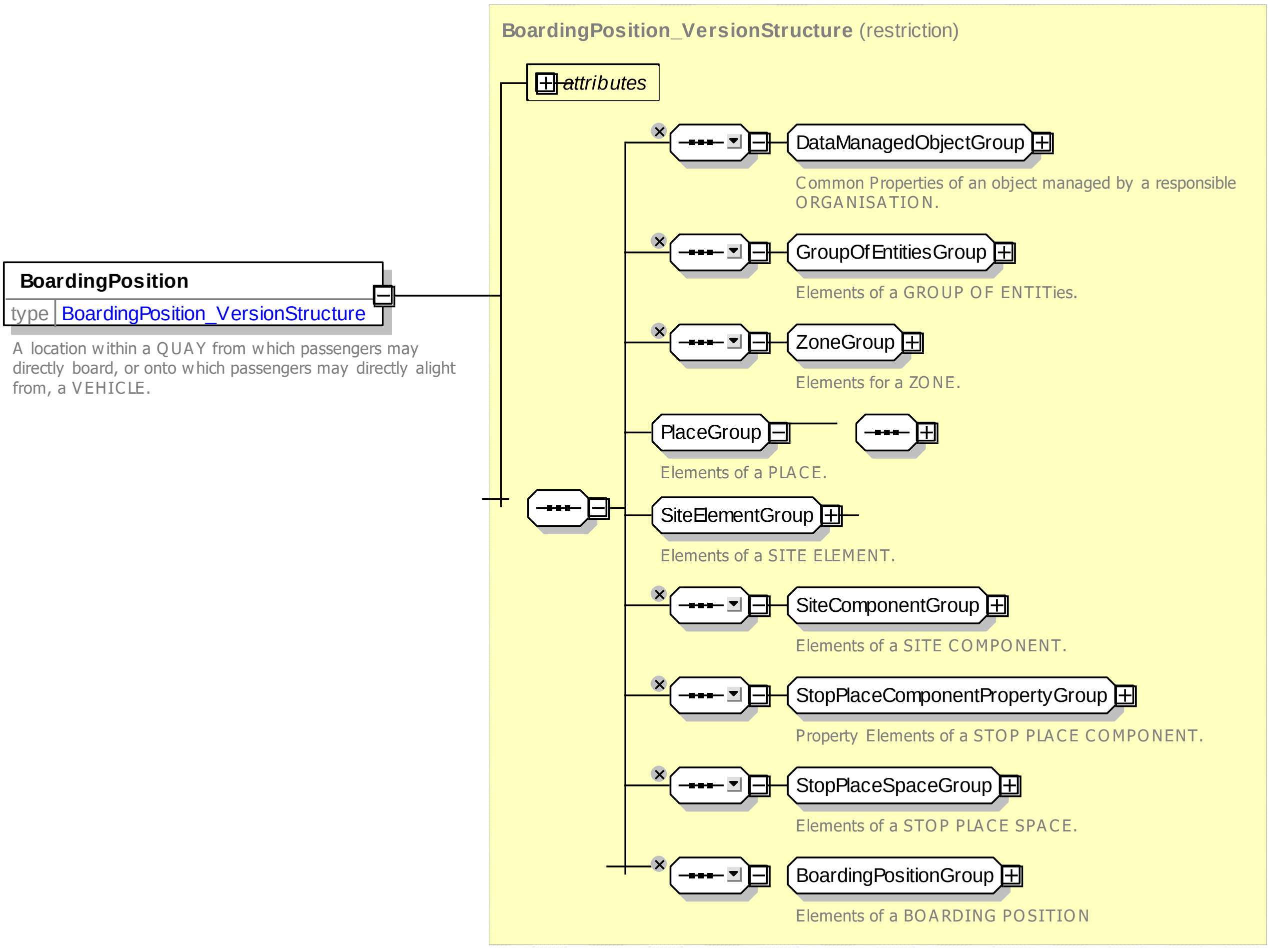


Figure 529 – *BoardingPosition* – *XSD*

8.4.4.4.7.1 **BoardingPositionGroup – XML Group**

Table 439 – *BoardingPositionGroup* – Group

Classification	Name	Type	Cardinality	Description
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Code used to identify BOARDING POSITION to the public.
	Label	*MultilingualString*	0:1	Label associated with BOARDING POSITION.
«enum»	***BoardingPosition Type***	*BoardingPositionType-Enum*	0:1	Type of BOARDING POSITION. See allowed values below.
«cntd»	***boardingPosition Entrances***	*EntranceRef*	0:*	Entrances to BOARDING POSITIONs.

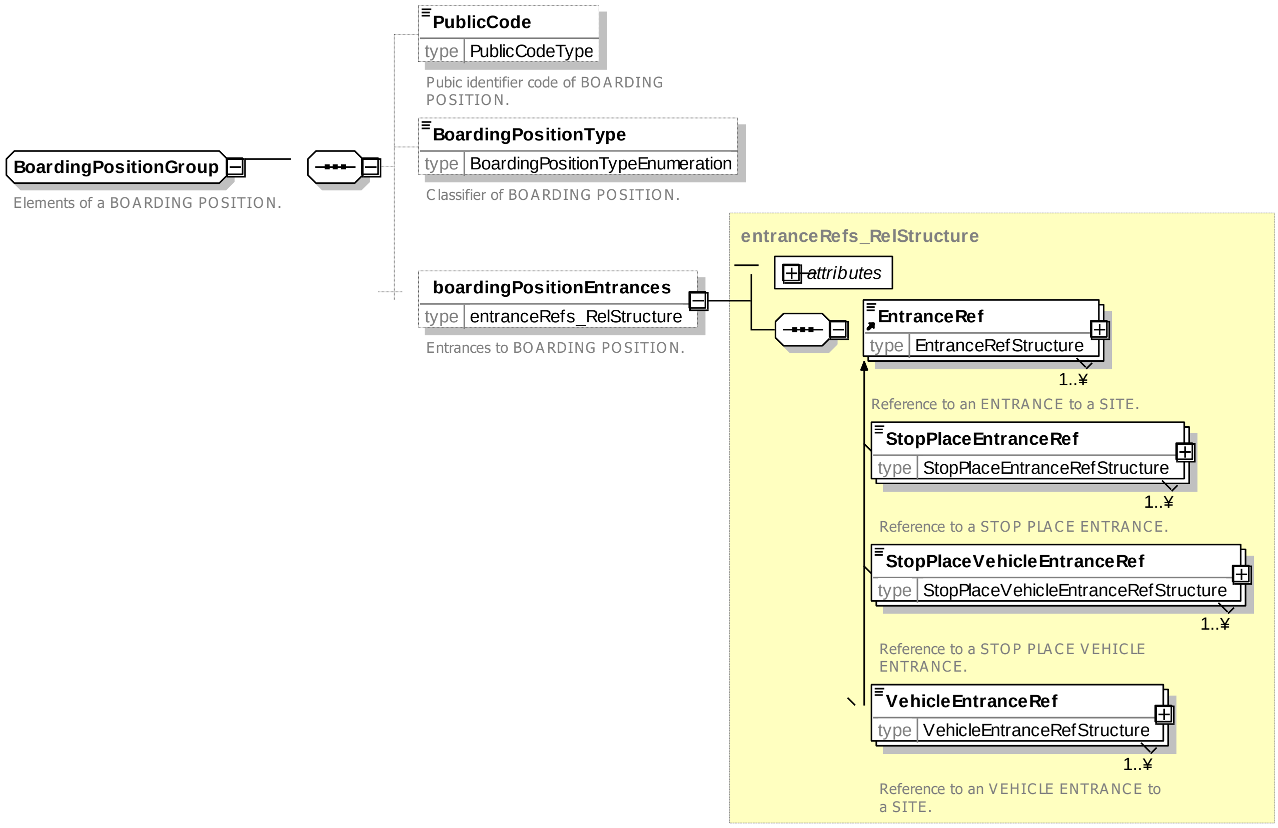


Figure 530 – *BoardingPositionGroup – XSD*

BoardingPositionType – Allowed values

The following table shows the allowed values for ***BoardingPositionType*** (*BoardingPositionTypeEnumeration).*

Table 440 – *BoardingPositionType* – Allowed values

Value	Description		
doorFromAirlineGate	Door from airline gate.	*ferryGangway*	Ferry Gangway.
positionOnRailPlatform	Position on rail platform.	*telecabineplatform*	Telecabine platform.
positionOnMetroPlatform	Position on metro platform.	*setDownPoint*	Set-down point.
positionAtCoachStop	Position at coach stop.	*taxiBay*	Taxi bay.
positionAtBusStop	Position at bus stop.	*vehicleLoadingRamp*	Vehicle loading ramp.
boatGangway	Boat Gangway.	*unknown*	Unknown.
		other	Other.

8.4.4.4.8 *AccessSpace – Model Element*

A passenger area within a STOP PLACE such as a concourse or booking hall, immigration hall or security area that is accessible by passengers, but without a direct access to vehicles. Direct access to a VEHICLE is always from a QUAY and/or BOARDING POSITION. An ACCESS SPACE may be a Room, Hall, Concourse, Corridor, or bounded open space within a STOP PLACE.

Table 441 – *AccessSpace* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*StopPlaceSpace*	::>	ACCESS SPACE inherits from STOP PLACE SPACE.
«PK»	***id***	*AccessSpaceIdType*	1:1	Identifier of ACCESS SPACE.
«enum»	***AccessSpaceType***	*AccessSpaceTypeEnum*	0:1	Type of ACCESS SPACE. See allowed values below.
«enum»	***PassageType***	*PassageTypeEnum*	0:1	Type of passage associated with ACCESS SPACE. See allowed values below.
«FK»	***ParentAccess-SpaceRef***	*AccessSpaceRef*	0:1	Reference to parent ACCESS SPACE within which this space is contained.

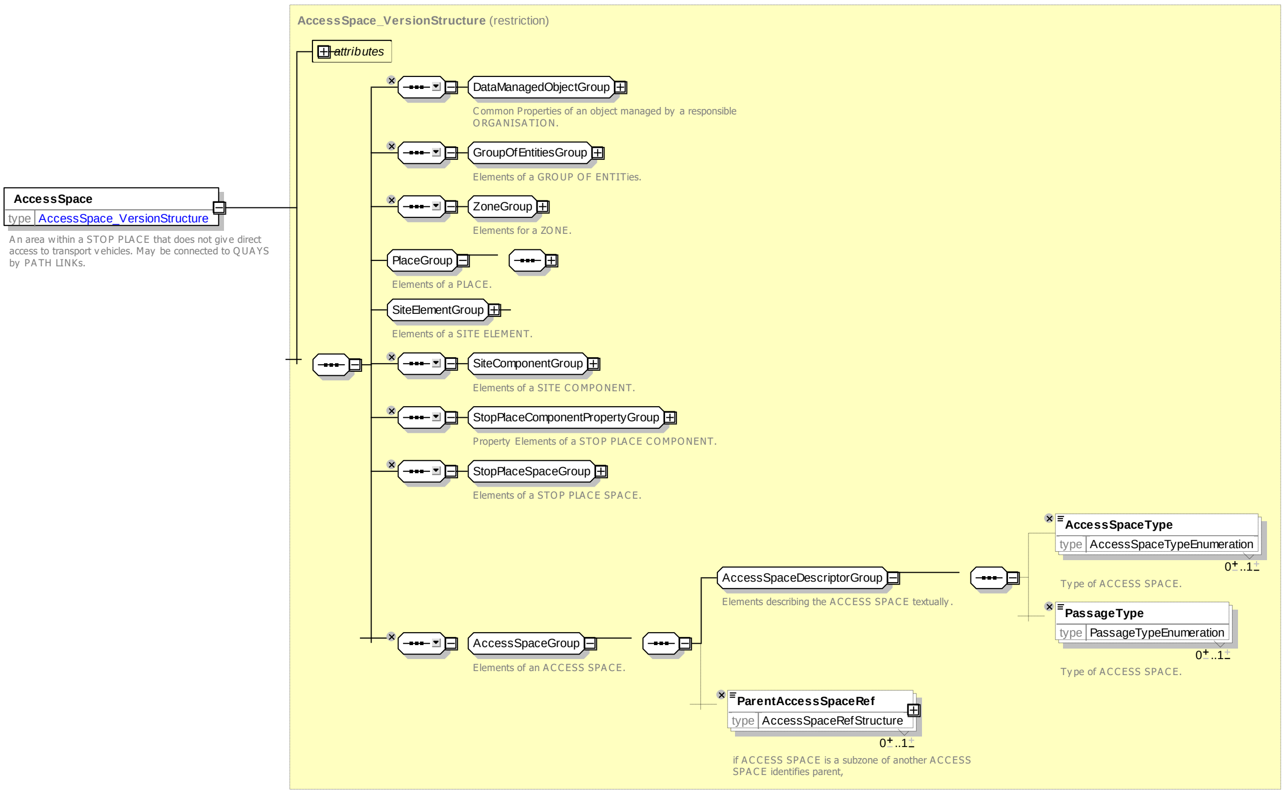


Figure 531 – *AccessSpace – XSD*

8.4.4.4.8.1 AccessSpaceType – Allowed values

The following table shows the allowed values for ***AccessSpaceType (****AccessSpaceTypeEnumeration****).***

Table 442 – *AccessSpaceType* – Allowed values

Value	Description
concourse	Concourse
bookingHall	Booking hall
forecourt	Forecourt
underpass	Under pass
overpass	Overpass

passage	Passage	*gallery*	Gallery	*restaurant*	Restaurant
passageSection	Passages Section	*garage*	Garage	*other*	Other
lift	Lift	*shop*	Shop	*staircase*	Staircase
		waitingRoom	Waiting room	*wc*	WC

8.4.4.4.8.2 **PassageType – Allowed values**

The following table shows the allowed values for ***PassageType (****PassageTypeEnumeration)****.***

Table 443 – *PassageType* – Allowed values

Value	Description
none	None
pathway	Pathway
corridor	Corridor
overpass	Overpass
underpass	Underpass
tunnel	Tunnel
other	Other

8.4.4.4.9 GroupOfStopPlaces – Model Element

A grouping of STOP PLACEs which will be commonly referenced for a specific purpose.

Table 444 – *GroupOfStopPlaces* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	***GroupOfStopPlaces*** inherits from ***GroupOfEntities.***
«PK»	***id***	*GroupOfStopPlaces-IdType*	1:1	Identifier of a GROUP of STOP PLACEs.
	PublicCode	*PublicCodeType*	0:1	Public code for GROUP OF STOP PLACEs +v1.1
«cntd»	***members***	*StopPlaceRef*	0:*	Members of a GROUP of STOP PLACEs.
«cntd»	***alternativeNames***	*AlternativeName*	0:*	ALTERNATIVE NAMEs for GROUP of STOP PLACEs.
«cntd»	***Centroid***	*SimplePoint*	0:1	Centre Coordinates of GROUP of STOP PLACEs. +v1.1
«cntd»	***gml:Polygon***	*gml:PolygonType*	0:1	GML polygon defining bounding outline of GROUP OF STOP PLACEs. +v1.1

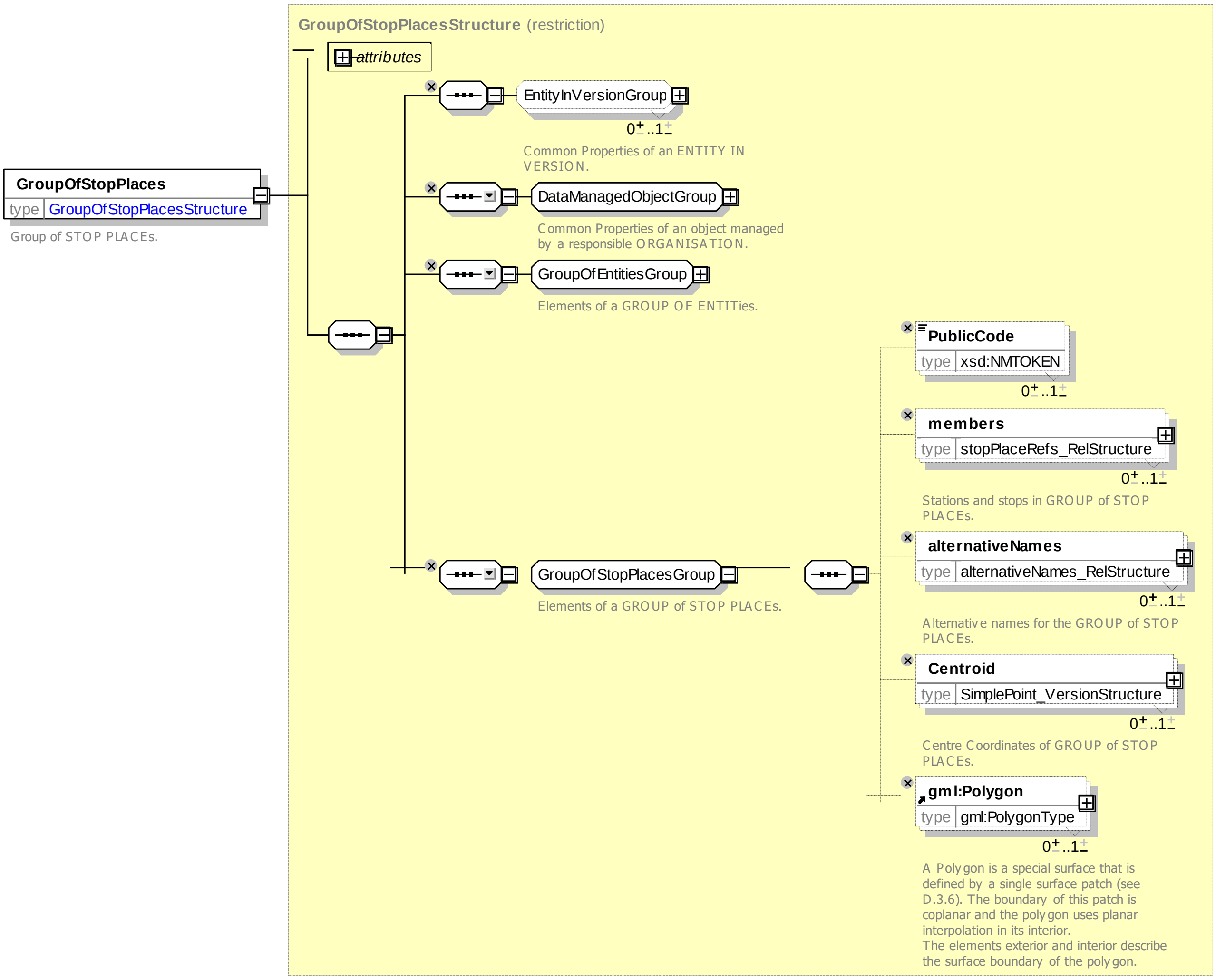


Figure 532 – *GroupOfStopPlaces – XSD*

8.4.4.5 StopPlace – XML Examples

8.4.4.5.1 XML Example of Stop Place and Quay for an On-Street stop

The following XML code fragment shows an on street bus stop as a simple STOP PLACE with two QUAYs (only one of which is shown), using data from an equivalent NaPTAN 3.x representation.

EXAMPLE XML Example of StopPlace

```
<StopPlace created="2006-09-11T15:42:00" id=”napt:490G0019043”>
    <Name>St George's Road (SW19)</Name>
    <Location srsName="UKOS">
        <Coordinates>524811  170666 </Coordinates>
    </Location>
    <types>
        <TypeOfPointRef>GPBS</TypeOfPointRef>
    </types>
    <ShortName>Wimbledon </ShortName>
    <PublicUse>true </ PublicUse >
    <TopographicPlaceRef>nptg:E0034695</TopographicPlaceRef>
    <TransportMode>bus</TransportMode>
    <TypeOfStopPlace>onstreetBus</TypeOfStopPlace>
    <ParentStopPlaceRef>napt:490G00272P</ParentStopPlaceRef>
    <quays>
        <Quay created="2010-04-17T09:30:47Z" dataSourceRef="NaPTAN" id=”napt:490014734A”>
            <Name>Alexandra Road, Stop A</Name>
            <Centroid>
                <Location>
                    <Longitude>-0.2067466166</Longitude>
                    <Latitude>51.4222367962</Latitude>
                </Location>
            </Centroid>
            <types>
                <TypeOfPointRef>BCT</TypeOfPointRef>
            </types>
            <zoneTypes>
                <TypeOfZoneRef>MKD</TypeOfZoneRef>
            </zoneTypes>
            <ShortName>Alexandra Road</ShortName>
            <Covered>outdoors</Covered>
            <RoadAddress id=”Rd_Addr_08”>
                <RoadName>Alexandra Road</RoadName>
                <BearingCompass>N</BearingCompass>
            </RoadAddress>
            <StopPlaceRef>napt:490G0019043</StopPlaceRef>
            <LevelRef>tbd:9100WIMBLDN_Lvl_S0</LevelRef>
            <Description>Stop A is paired with Stop B on Alexandra Road St Georges Road</Description>
            <BoardingUse>true</BoardingUse>
            <AlightingUse>true</AlightingUse>
            <PublicCode>1-2345</PublicCode>
            <Label>Stop A</Label>
            <CompassOctant>N</CompassOctant>
            <QuayType>busStop</QuayType>
        </Quay>
        <Quay>          ::>
    </quays>
</StopPlace>
```

8.4.4.5.2 XML Example of a Stop Place for a Station with platforms

The following XML code fragment shows part of a STOP PLACE for a station, including a definition of a single ENTRANCE (further ones have been omitted from the fragment shown) and one of its platforms. The platform is two sided, so is described as a pair of QUAY instances nested inside another QUAY – see discussion of nested QUAYs in Section 8.4.4.2.4. The station is on two LEVELs (only the definition of the first is shown).

Some other points of remark in the example code:

The station's overall accessibility rating is described by an ACCESSIBILITY ASSESSMENT.

— The station has an ALTERNATIVE NAME.

— The station is tagged as being the main terminus for a TOPOGRAPHIC PLACE (corresponding to a reference to a NTPG Locality as the NPTG main locality).

— Individual ENTRANCEs and QUAY's are also tagged with an ACCESSIBILITY ASSESSMENT.

EXAMPLE XML Example of StopPlace – Rail Station with Platforms

```xml
<StopPlace created="2006-09-11T15:42:00" modification="revise" dataSourceRef="NaPTAN">
    <ResponsibilitySetRef>napt:RS_110</ResponsibilitySetRef id="napt:910GWIMBLDN">
    <Name>Wimbledon Rail Station</Name>
    <Location srsName="UKOS">
        <Coordinates>524811 170666 </Coordinates>
    </Location>
    <types>
        <TypeOfPointRef>GRLS</TypeOfPointRef>
    </types>
    <AccessibilityAssessment>
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation created="2010-05-17T09:30:47Z">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
                <AudibleSignalsAvailable>false</AudibleSignalsAvailable>
                <VisualSignsAvailable>true</VisualSignsAvailable>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <ShortName>Wimbledon Station</ShortName>
    <alternativeNames modificationSet="all">
        <AlternativeName created="2010-05-17T09:30:47Z" modification="new">
            <NameType>label</NameType>
            <Name>Wimbledon+ </Name>
        </AlternativeName>
    </alternativeNames>
    <Covered>mixed</Covered>
    <TopographicPlaceRef>E0034695</TopographicPlaceRef>
    <mainTerminusForPlaceRefs>
        <TopographicPlaceRef created="2005-04-08T00:00:00">E0034695</TopographicPlaceRef>
    </mainTerminusForPlaceRefs>
    <RoadAddress created="2010-05-17T09:30:47Z" modification="new"id="tbd:RdAddr_01">
        <RoadName>Wimbledon Bridge +</RoadName>
    </RoadAddress>
    <!-- ============LEVELS ===================- -->
    <levels>
```

```xml
    <Level created="2010-04-17T09:30:47Z" id="tbd:9100WIMBLDN_Lvl_G0">
        <Name>Ground </Name>
        <LevelCode>G</LevelCode>
    </Level>
:::::
</levels>
<!-- ============ENTRANCEs ====================- -->
<entrances>
    <Entrance created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_A3_EE1">
        <Name>External Entrance to Centre Court Ticket Hall from forecourt</Name>
        <validityConditions>
            <AvailabilityConditionRef>AC_01_Main_Opening</AvailabilityConditionRef>
        </validityConditions>
        <ParentZoneRef>tbd:9100WIMBLDN_A3</ParentZoneRef>
        <AccessibilityAssessment>
            <MobilityImpairedAccess>true</MobilityImpairedAccess>
            <limitations>
                <AccessibilityLimitation>
                    <WheelchairAccess>true</WheelchairAccess>
                    <StepFreeAccess>true</StepFreeAccess>
                </AccessibilityLimitation>
            </limitations>
        </AccessibilityAssessment>
        <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
        <placeEquipments>
            <EntranceEquipment>
                <Door>true</Door>
                <KeptOpen>true</KeptOpen>
                <WheelChairPassable>true</WheelChairPassable>
            </EntranceEquipment>
        </placeEquipments>
        <EntranceType>openDoor</EntranceType>
        <isExternal>true</isExternal>
        <isEntry>true</isEntry>
        <isExit>true</isExit>
        <Width>1.0</Width>
        <Height>3.0</Height>
    </Entrance>
::::::::
</entrances>
    <!-- ============QUAYs ====================- -->
<quays>
    <Quay created="2010-04-17T09:30:47Z" id="tbd:9100WIMBLDN5n6">
        <Name>Platforms 5 &amp; 6</Name>
        <Location srsName="UKOS">
            <Coordinates>524811 170666 </Coordinates>
        </Location>
        <AccessibilityAssessment created="2010-05-17T09:30:47Z" modification="new">
            <MobilityImpairedAccess>true</MobilityImpairedAccess>
            <limitations>
                <AccessibilityLimitation created="2010-05-17T09:30:47Z" modification="new">
                    <WheelchairAccess>true</WheelchairAccess>
                    <StepFreeAccess>true</StepFreeAccess>
                    <EscalatorFreeAccess>true</EscalatorFreeAccess>
                    <LiftFreeAccess>true</LiftFreeAccess>
                    <AudibleSignalsAvailable>false</AudibleSignalsAvailable>
                    <VisualSignsAvailable>true</VisualSignsAvailable>
                </AccessibilityLimitation>
            </limitations>
```

```xml
            </AccessibilityAssessment>
            <Covered>covered</Covered>
            <LevelRef>tbd:9100WIMBLDN_Lvl_U1</LevelRef>
            <Description>Platforms 5 &amp; 6 </Description>
            <BoardingUse>true</BoardingUse>
            <AlightingUse>true</AlightingUse>
            <Label>5 and 6</Label>
            <destinations>
                <DestinationDisplay>Clapham Junction</DestinationDisplay>
                <DestinationDisplay>Waterloo</DestinationDisplay>
            </destinations>
            <QuayType>railPlatform</QuayType>
            <quayEntrances>
                <EntranceRef>tbd:9100WIMBLDN5n6_EL1</EntranceRef>
                <EntranceRef>tbd:9100WIMBLDN5n6_ES1</EntranceRef>
            </quayEntrances>
        </Quay>
        <Quay created="2010-04-17T09:30:47Z" id="napt:9100WIMBLDN5">
            <Name>Platform 5</Name>
            <Description>Platform 5 is paired with platform 6 with separate lift and stair access
</Description>
            <Label>5</Label>
            <QuayType>railPlatform</QuayType>
            <ParentQuayRef>tbd:9100WIMBLDN5n6</ParentQuayRef>
        </Quay>
        <Quay created="2010-04-17T09:30:47Z" id="napt:9100WIMBLDN6">
            <Name>Platform 6</Name>
            <Description>Platform 5 is paired with platform 6 with separate lift and stair
access</Description>
            <Label>5</Label>
            <QuayType>railPlatform</QuayType>
            <ParentQuayRef>tbd:9100WIMBLDN5n6</ParentQuayRef>
        </Quay>
        :::::
</StopPlace>
```

8.4.4.5.3 XML Example of a Nested Stop Place

The following XML code fragment shows a STOP PLACE for a *metro* station that is itself a subsidiary part of another *rail* STOP PLACE (defined in the previous example).

EXAMPLE XML Example of Nested StopPlaces

```xml
<StopPlace created="2006-09-11T15:42:00" modification="revise" changed="2009-02-26T15:47:00"
id="napt:940GZZLUWIM">
    <Name>Wimbledon Underground Station </Name>
    <Centroid>
        <Location>
            <Longitude>-0.2065219984</Longitude>
            <Latitude>51.4213610557</Latitude>
        </Location>
    </Centroid>
    <types>
        <TypeOfPointRef>GTMU</TypeOfPointRef>
    </types>
    <ShortName>Wimbledon</ShortName>
    <TopographicPlaceRef>nptg:E0034695</TopographicPlaceRef>
    <entrances>
:::::::
```

```
    </entrances>
    <PrivateCode>86286</PrivateCode>
    <TypeOfStopPlace>metroStation</TypeOfStopPlace>
    <TransportMode>metro</TransportMode>
    <ParentStopPlaceRef>napt:910GWIMBLDN</ParentStopPlaceRef>
    <quays>
        ::>
    </quays>
    <accessSpaces>
::>
    </accessSpaces>
</StopPlace>
```

8.4.4.5.4 XML Example of a Multimodal use of the same platform

The following XML code fragment shows a rail STOP PLACE with a shared use QUAY.

EXAMPLE

```
<StopPlace    created="2006-09-11T15:42:00"    modification="revise"    changed="2009-02-26T15:47:00"
id="napt:910GWIMBLDN">
    <Name>Wimbledon Rail Station</Name>
    <Location srsName="UKOS">
        <Coordinates>524811  170666 </Coordinates>
    </Location>
    <types>
        <TypeOfPointRef>GRLS</TypeOfPointRef>
    </types>
::>……
    <TypeOfStopPlace>railStation</TypeOfStopPlace>
    <TransportMode>rail</TransportMode>
    <otherModes>
        <OtherTransportMode>metro</OtherTransportMode>
        <OtherTransportMode>tram</OtherTransportMode>
    </otherModes>
:::::
    <quays>
::>……
        <Quay created="2010-04-17T09:30:47Z" id="napt:9100WIMBLDN10">
            <Name>Platform 10</Name>
            <TransportMode>rail</TransportMode>
            <otherModes>
                <VehicleMode>tram</VehicleMode>
            </otherModes>
            <Description>Platform 10 is paired with platform 9 with separate lift and stair access. It
has shared use for tram</Description>
            <Label>10</Label>
            <destinations>
                <DestinationDisplay>London</DestinationDisplay>
            </destinations>
            <QuayType>tramPlatform</QuayType>
            <ParentQuayRef>tbd:9100WIMBLDN9n10</ParentQuayRef>
        </Quay>
        <QuayRef>tbd:9100WIMBLDN10</QuayRef>
    </quays>
```

The following XML code fragment shows an additional STOP PLACE for a tram station that references the same shared use rail platform defined above.

EXAMPLE XML Example of Shared Quay in a Rail StopPlace

```xml
<StopPlace    created="2006-09-11T15:42:00"    modification="revise"    changed="2009-02-26T15:47:00"
id="napt:940GZZCRWIM">
    <Name>Wimbledon Tramlink Station </Name>
    <Centroid>
        <Location>
            <Longitude>-0.2065219984</Longitude>
            <Latitude>51.4213610557</Latitude>
        </Location>
    </Centroid>
    <types>
        <TypeOfPointRef>GTMU</TypeOfPointRef>
    </types>
    <ShortName>Wimbledon</ShortName>
    <TopographicPlaceRef>nptg:E0034695</TopographicPlaceRef>
    <TypeOfStopPlace>tramStation</TypeOfStopPlace>
    <TransportMode>tram</TransportMode>
    <ParentStopPlaceRef>napt:910GWIMBLDN</ParentStopPlaceRef>
    <quays>
        <QuayRef>tbd:9100WIMBLDN10</QuayRef>
    </quays>
</StopPlace>
```

8.4.5 Flexible Stop Place

8.4.5.1 FLEXIBLE STOP PLACE – Conceptual MODEL.

Hail and ride or zone based flexible services which serve a section of road or an area can be represented in *NeTEx* using FLEXIBLE STOP PLACEs and FLEXIBLE QUAYs. This allows the sections or zones to appear as named “stops” in their own right in a journey planner. It has to be noted that flexible services can also use regular STOP PLACEs and QUAYs.

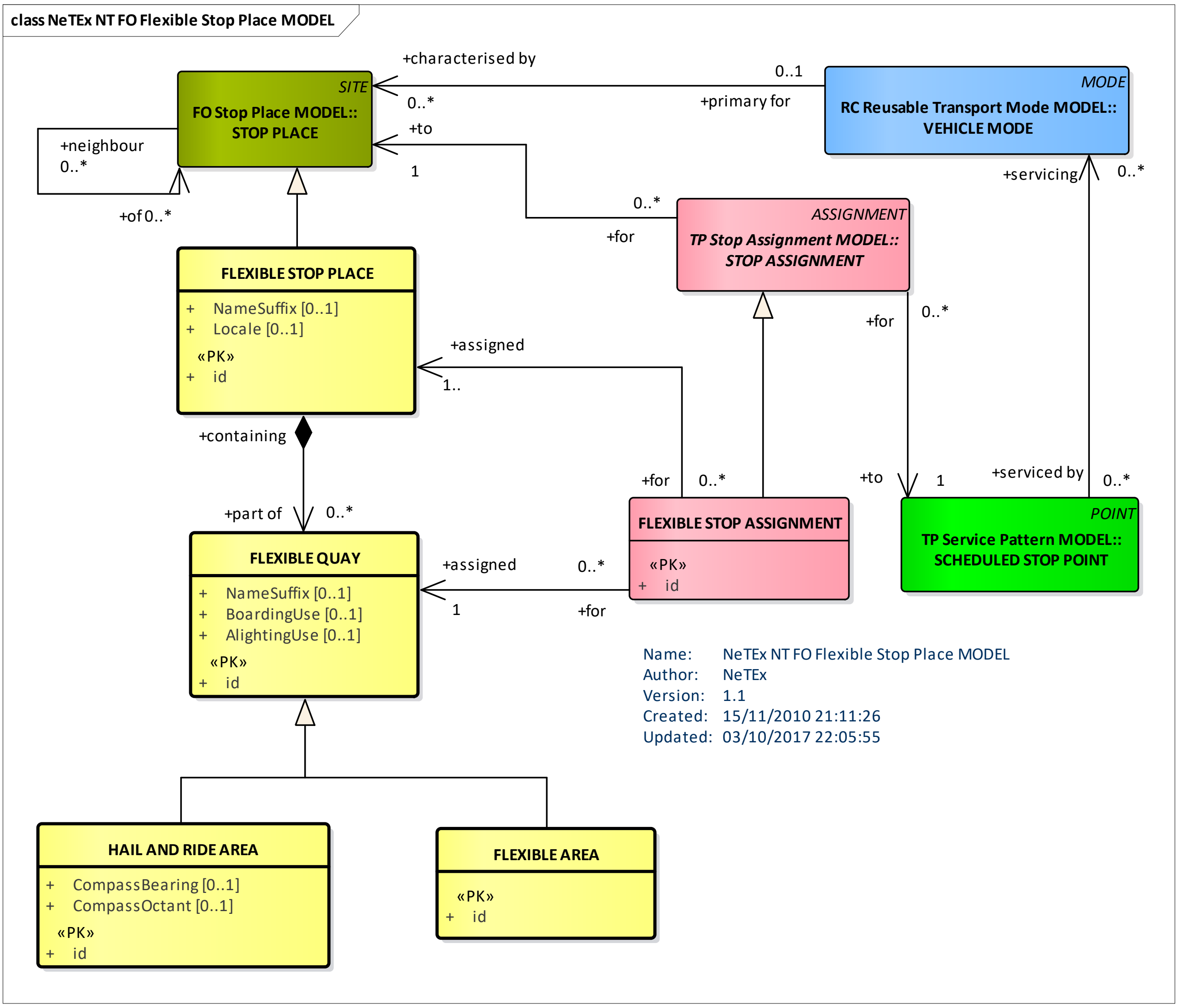


Figure 533 – FLEXIBLE STOP PLACE – Conceptual MODEL (UML)

8.4.5.2 Flexible Stop Place – Examples

The following diagrams show examples of flexible stops.

8.4.5.2.1 Simple Examples Hail and Ride Stop

The following figures describe a Hail and Ride section along a bus line (in red). It has two separated start and end points, one for each direction. Between start and stop point, the vehicle can be stopped on any point (on passenger demand) for boarding or alighting.

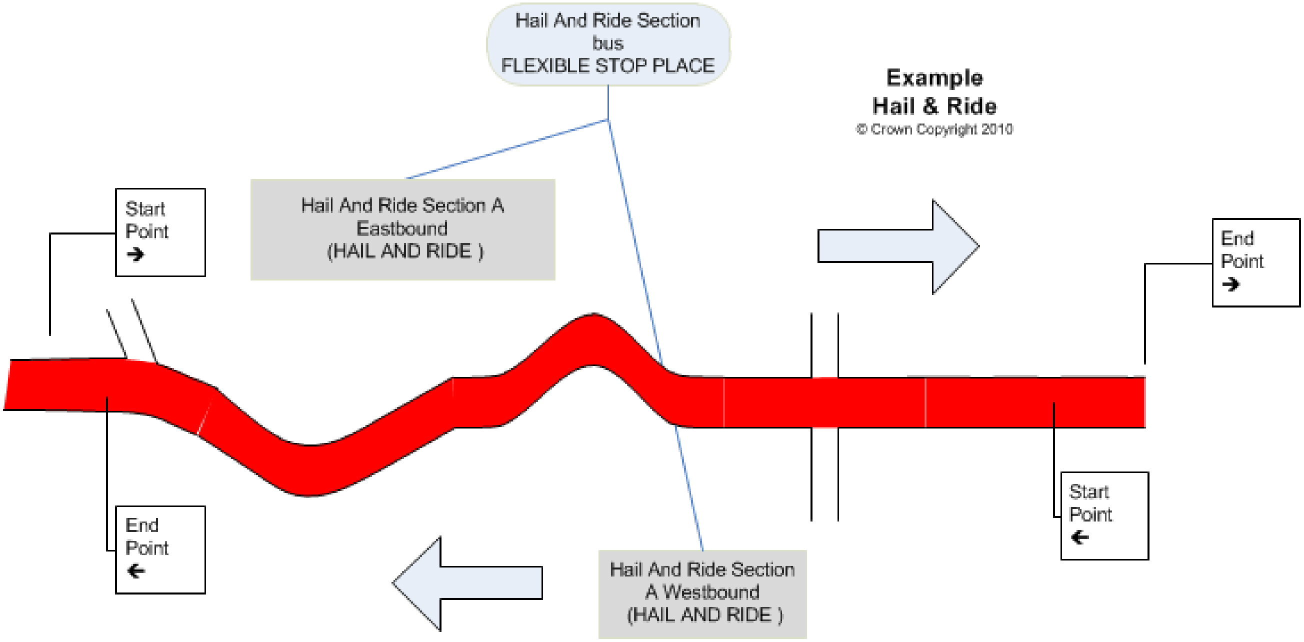


Figure 534 – Example of Hail and Ride Stop

8.4.5.2.2 Simple Examples of Flexible Stop

The following figure describes two flexible zones on some bus lines (main route in red). In this example, Hail and Ride is available for any part of the bus lines inside the defined flexible zones.

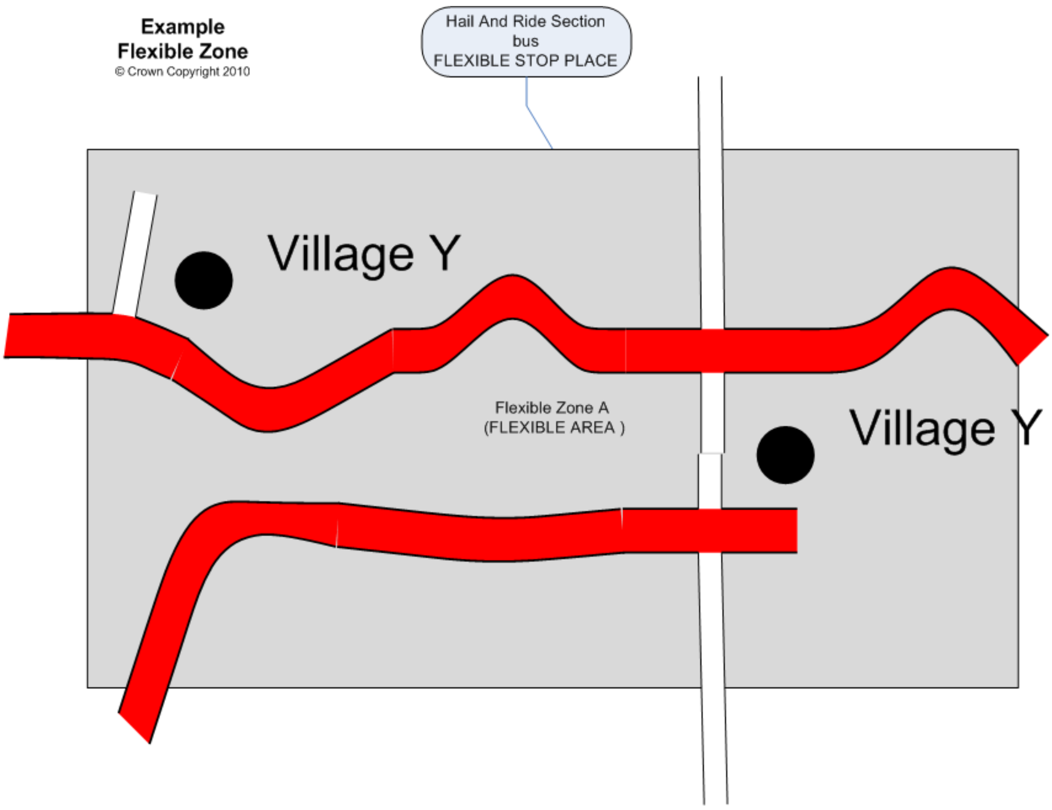


Figure 535 – Example of Flexible Zone

8.4.5.3 Flexible Stop Place – Physical Model

The following figure shows detailed attributes of the FLEXIBLE STOP PLACE model.

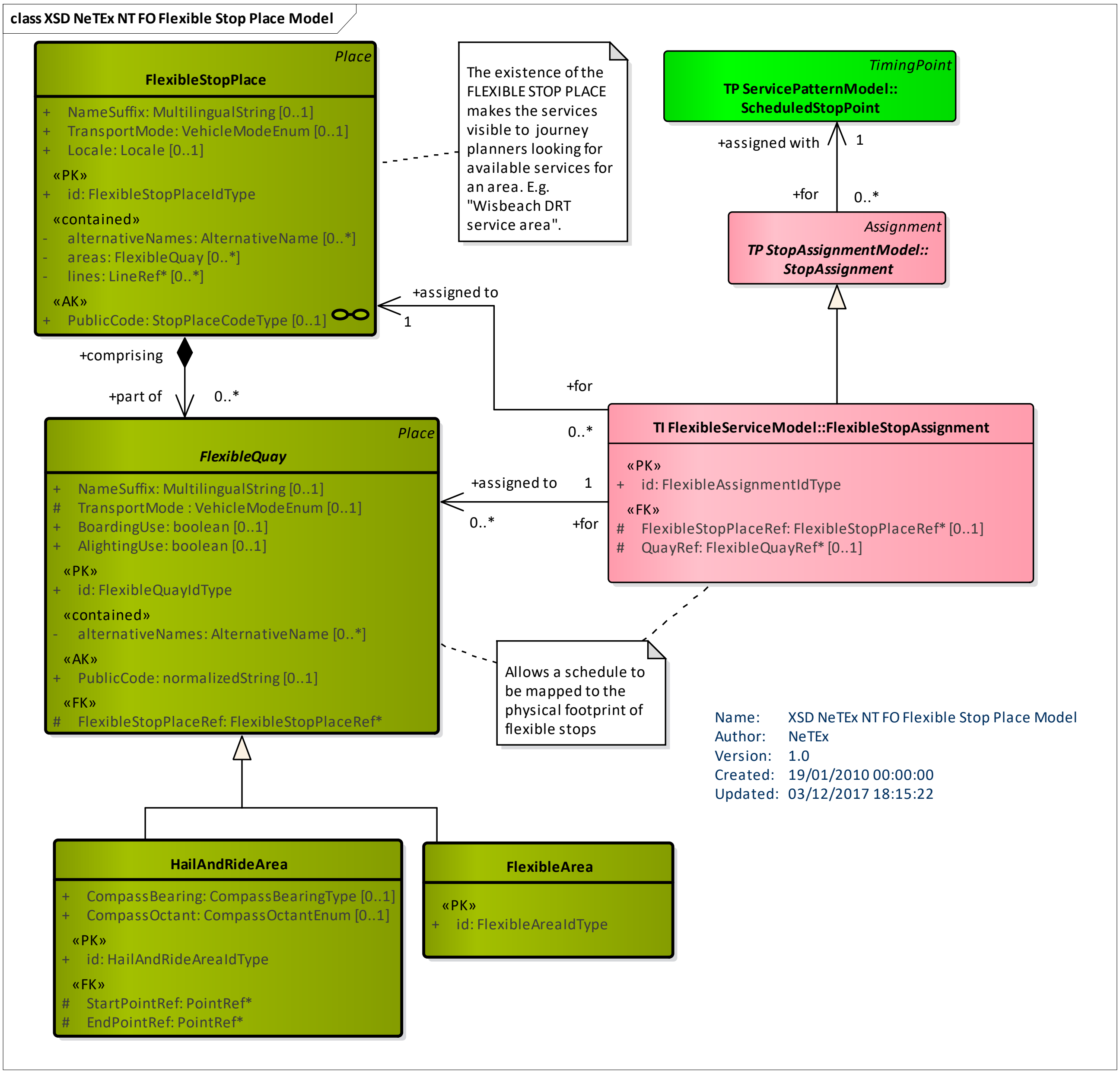


Figure 536 – Flexible StopPlace – Physical Model (UML)

8.4.5.4 Flexible Stop Place – Attributes and XSD

8.4.5.4.1 FlexibleStopPlace – Model Element

A type of STOP PLACE for FLEXIBLE services comprising one or more flexible zones where vehicles may stop and where passengers may board or leave vehicles.

Table 445 – *FlexibleStopPlace* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Place*	::>	FLEXIBLE STOP PLACE inherits from PLACE.
«PK»	***id***	*FlexibleStopPlaceIdType*	1:1	Identifier of a FLEXIBLE STOP PLACE.
	NameSuffix	*MultilingualString*	1:1	Suffix to use on Name.
«cntd»	***alternativeNames***	*AlternativeName*	0:*	Alternative names for SITE ELEMENT.
«enum»	***TransportMode***	*VehicleModeEnum*	1:1	The main TRANSPORT MODE of the FLEXIBLE STOP PLACE. Ee TRANSPORT MODE for allowed values.
«cntd»	***Locale***	*Locale*	0:1	Locale setting time zone, default language etc., for the FLEXIBLE STOP PLACE. See Reusable components.
«AK»	***PublicCode***	*PublicCodeStructure*	0:1	Code used to identify a FLEXIBLE STOP PLACE to the public as an alternative to a name.
«cntd»	***areas***	*FlexibleQuay*	0:*	FLEXIBLE QUAYs of a FLEXIBLE STOP PLACE.

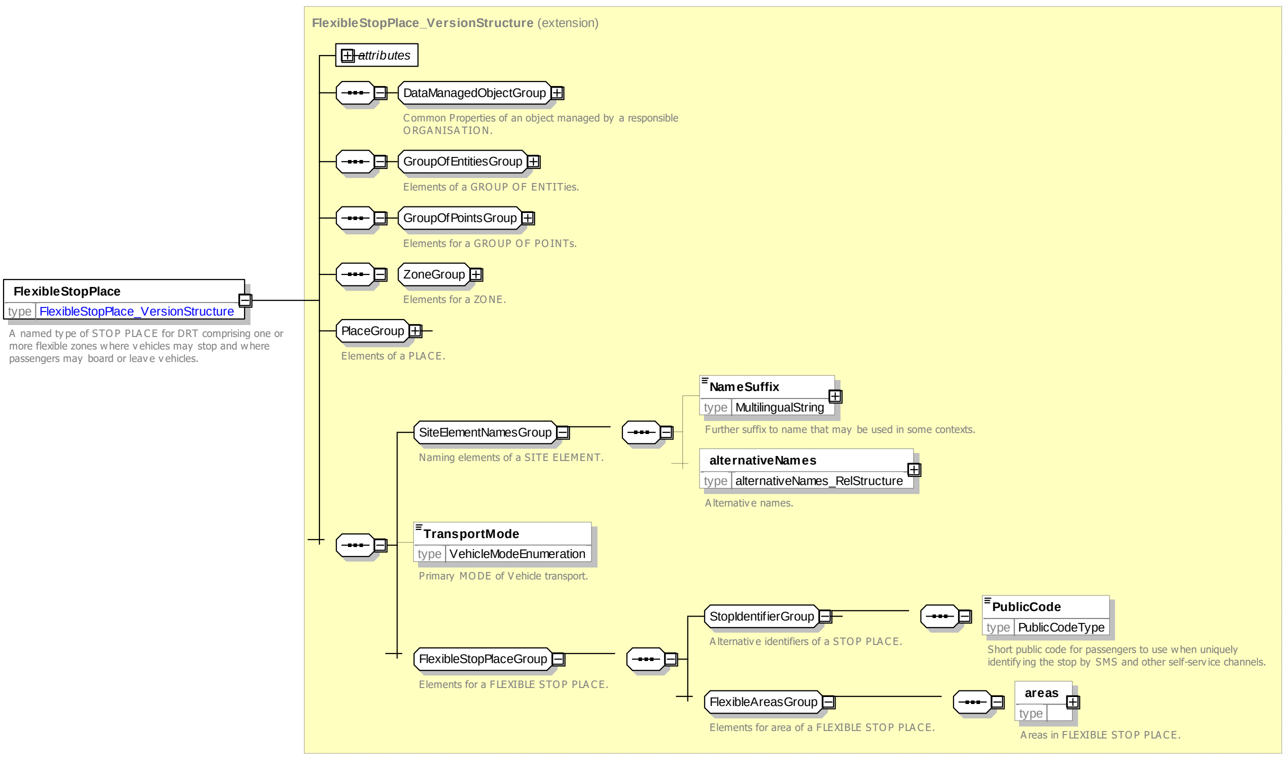


Figure 537 – *FlexibleStopPlace – XSD*

8.4.5.4.2 FlexibleQuay – Model Element

A physical ZONE such as a section of a road where a flexible service is available on demand. The existence of the zone makes the services visible to journey planners looking for available services for an area.

Table 446 – *FlexibleQuay* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Place*	::>	FLEXIBLE QUAY inherits from PLACE.
«PK»	***id***	*FlexibleQuayIdType*	1:1	Identifier of a FLEXIBLE QUAY.
	NameSuffix	*MultilingualString*	1:1	Suffix to use on Name.
«cntd»	***alternativeNames***	*AlternativeName*	0:*	Alternative names for SITE ELEMENT.
«enum»	***TransportMode***	*VehicleModeEnum*	0:1	Main TRANSPORT MODE of FLEXIBLE STOP PLACE. If not specified, same as parent STOP PLACE.
	BoardingUse	*xsd:boolean*	1:1	Whether Passengers may use the FLEXIBLE QUAY for Boarding vehicle transport.
	AlightingUse	*xsd:boolean*	1:1	Whether Passengers may use the FLEXIBLE QUAY for Alighting from vehicle transport.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Public code of a FLEXIBLE STOP PLACE.
«FK»	***FlexibleStopPlaceRef***	*FlexibleStopPlaceRef*	0:1	Reference to a parent FLEXIBLE STOP PLACE.

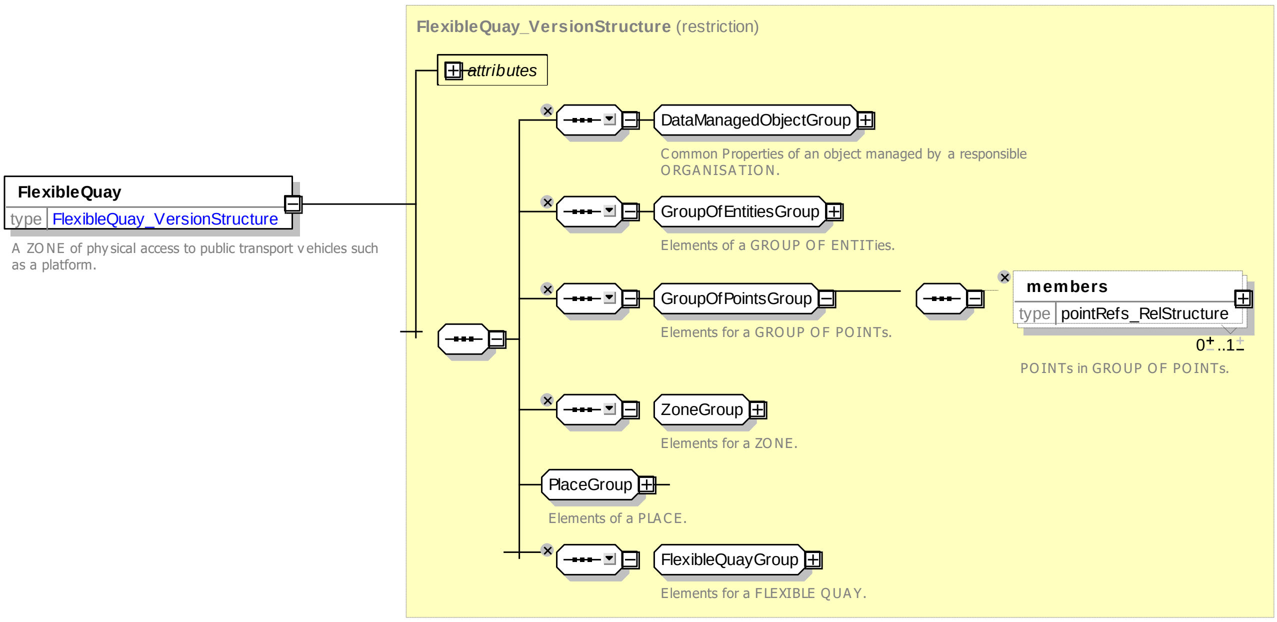


Figure 538 – *FlexibleQuay – XSD*

8.4.5.4.3 FlexibleArea – Model Element

A FLEXIBLE AREA is the physical ZONE within which a flexible service is available for pickup or drop off. Allows the projection of the zone onto a map.

Table 447 – *FlexibleArea* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*FlexibleQuay*	::>	FLEXIBLE AREA inherits from FLEXIBLE QUAY.
«PK»	***id***	*FlexibleAreaIdType*	1:1	Identifier of a Flexible AREA.

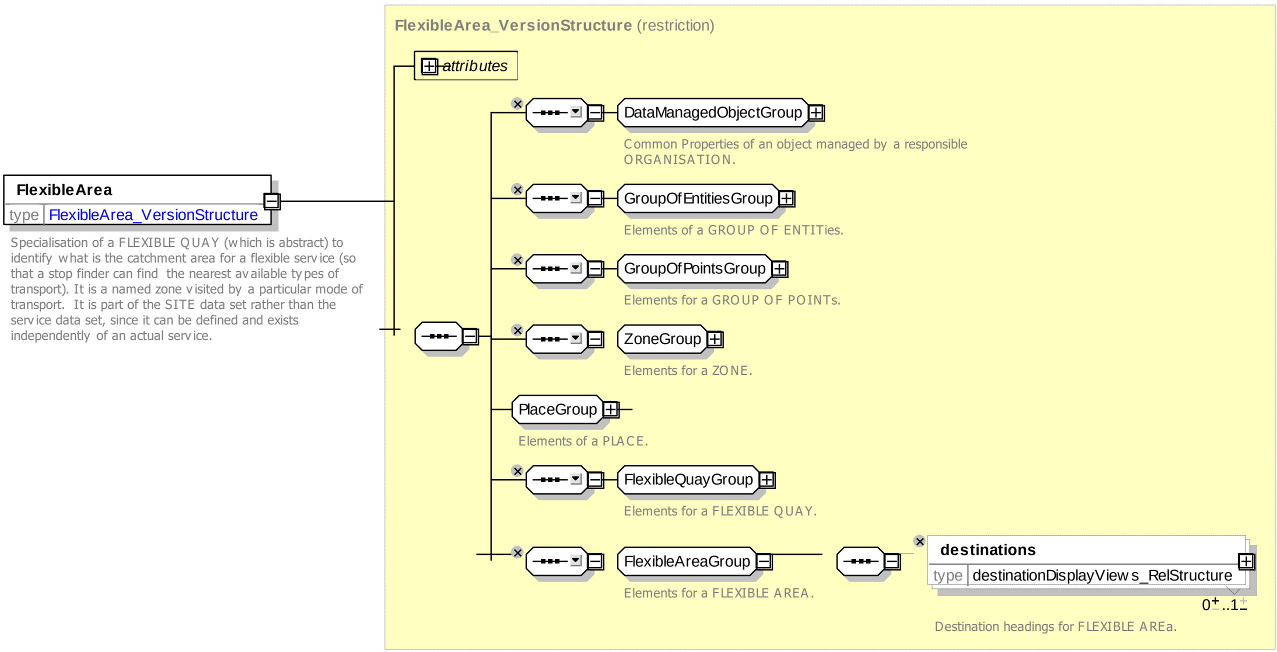


Figure 539 – *FlexibleArea – XSD*

8.4.5.4.4 *HailAndRideArea – Model Element*

A physical section of road between a start and end point within which a Hail and Ride service is available

Table 448 – *HailAndRideArea* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*FlexibleQuay*	::>	HAIL AND RIDE AREA inherits from FLEXIBLE QUAY.
«PK»	***id***	*HailAndRideAreaIdType*	1:1	Identifier of a HAIL AND RIDE AREA.
	CompassBearing	*CompassBearingType*	0:1	Compass Bearing associated with a HAIL AND RIDE AREA in segrees

«enum»	***CompassOctant***	*CompassOctantEnum*	0:1	Compass Octant associated with a HAIL AND RIDE AREA (N, S, E W etc.). See utility types for allowed values.
«FK»	***StartPointRef***	(*PointRef*)	1:1	Start Point on road of the HAIL AND RIDE AREA.
«FK»	***EndPointRef***	(*PointRef*)	1:1	End Point on road of the HAIL AND RIDE AREA.

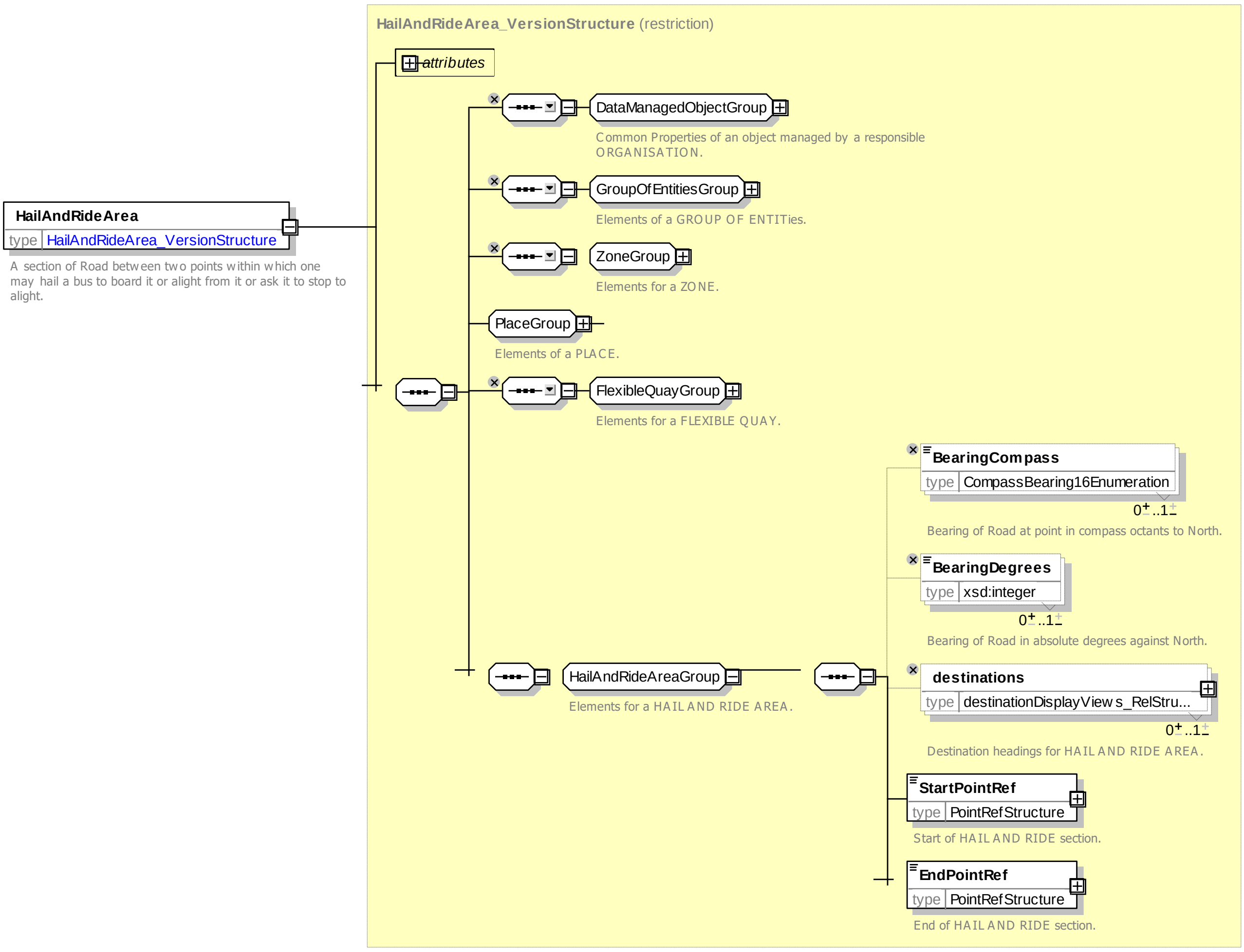


Figure 540 – *HailAndRideArea – XSD*

8.4.6 Point Of Interest

8.4.6.1 POINT OF INTEREST – Conceptual MODEL

STOP PLACEs are a type of SITE that provides access to public transport. A POINT of INTEREST is another type of SITE used to represent a well-known attraction such as a museum, stadium, park, venue, etc. Like a STOP PLACE a POINT of INTEREST may comprise SITE COMPONENTs such as designated ENTRANCEs and SITE SPACEs. PATH LINKs and NAVIGATION PATHs may be used to connect to it and within it and ACCESSIBILITY properties may be assigned. Some types of EQUIPMENT are also relevant, for examples lifts, stairs.

In addition a POINT OF INTEREST may be described by one or more POINT OF INTEREST CLASSIFICATIONs. These can be organised into a POINT OF INTEREST CLASSIFICATION HIERARCHY which provides a way of exchanging categorisations of the point of interest.

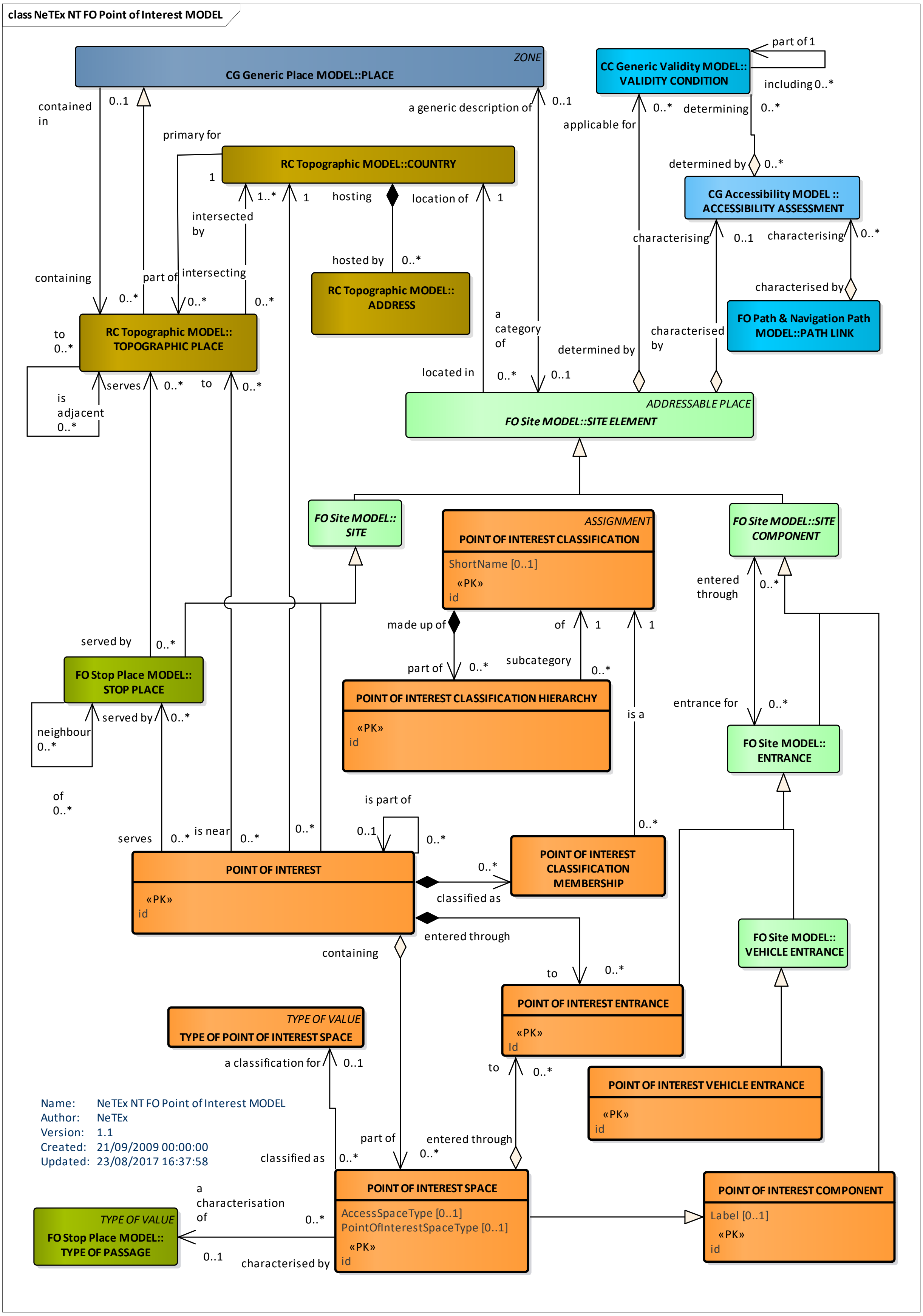


Figure 541 – POINT OF INTEREST – Conceptual MODEL

8.4.6.2 Point of Interest – Examples

8.4.6.2.1 Example of Point of Interest – Basic Stadium

The following figure shows the use of SITE COMPONENTs to describe just the entrances and existence of a stadium with four separate ENTRANCEs – similar to a NaPTAN 3.0 level of detail.

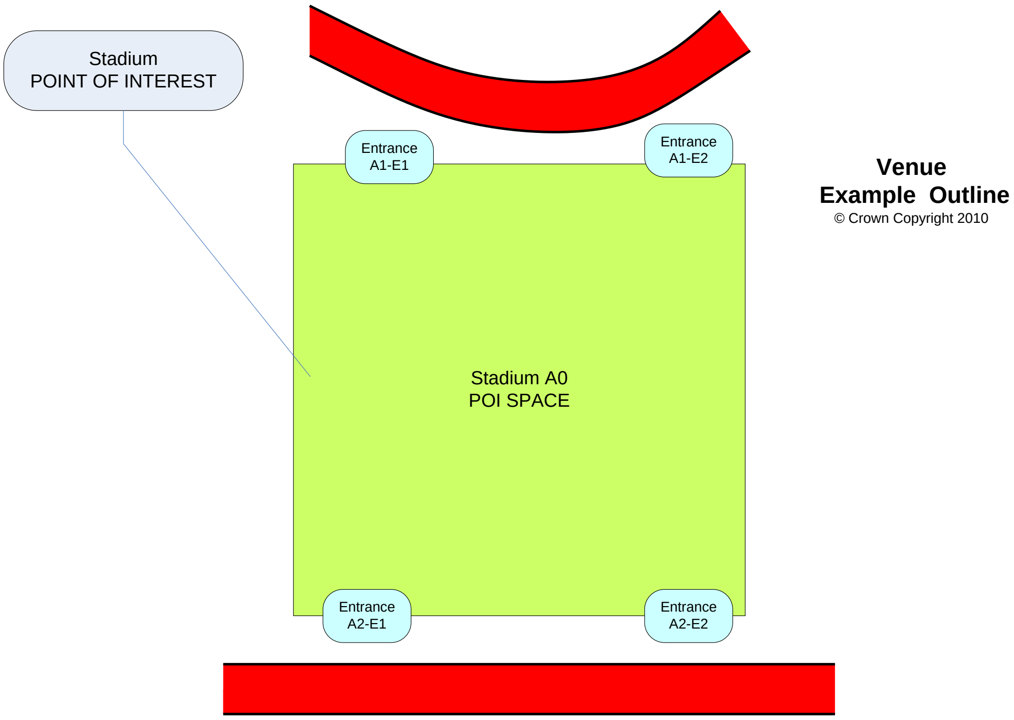


Figure 542 – Example Basic Point of interest

8.4.6.2.2 Example of Point of Interest – Stadium with Stands

The following figure shows the use of SITE COMPONENTs to describe access details to the same stadium shown in slightly more detail with two separate ENTRANCEs and security check areas. PATH LINKs describe the connectivity within the site.

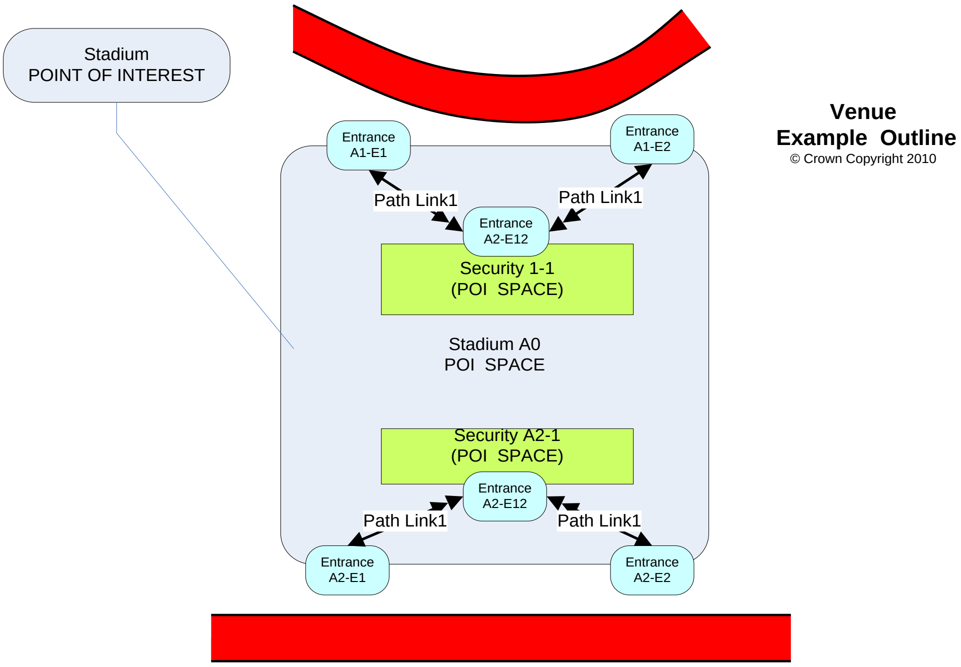


Figure 543 – Example Point of Interest – Stadium Outline

8.4.6.2.3 Example of Point of Interest – Stadium with detail

The following figure shows the use of SITE COMPONENTs to further describe the internal parts of the stadium sufficient to indicate the Entrance to use to reach a given set of seats.

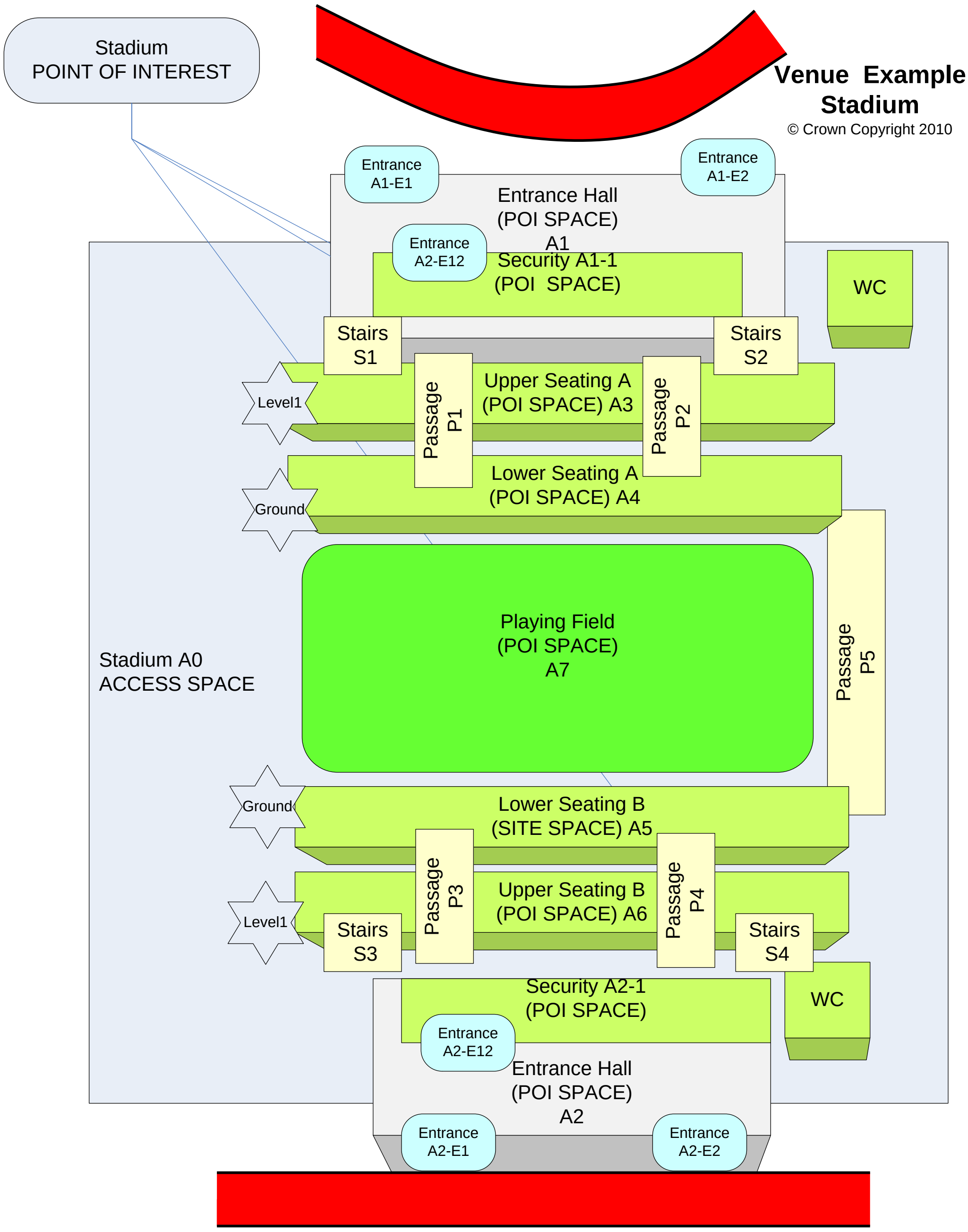


Figure 544 – Example Point of Interest – Stadium

8.4.6.2.4 Example of Point of Interest – Large Stadium

The following figure shows the use of SITE COMPONENTs to describe the seat number ranges within a large stadium so that the appropriate entry can be selected.

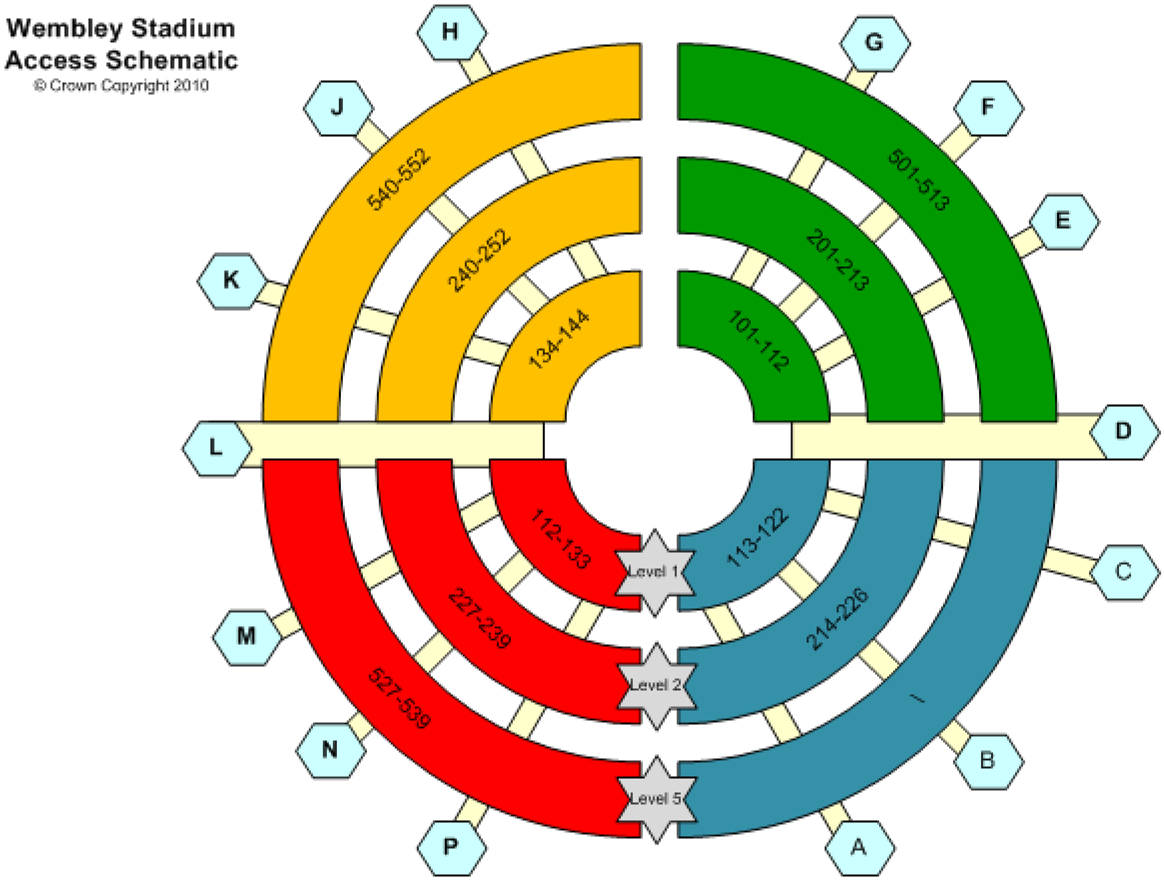


Figure 545 – Example Point of Interest – Stadium with Numbered sections

8.4.6.3 Point Of Interest – Physical Model

The following figure shows detailed attributes of the POINT OF INTEREST model.

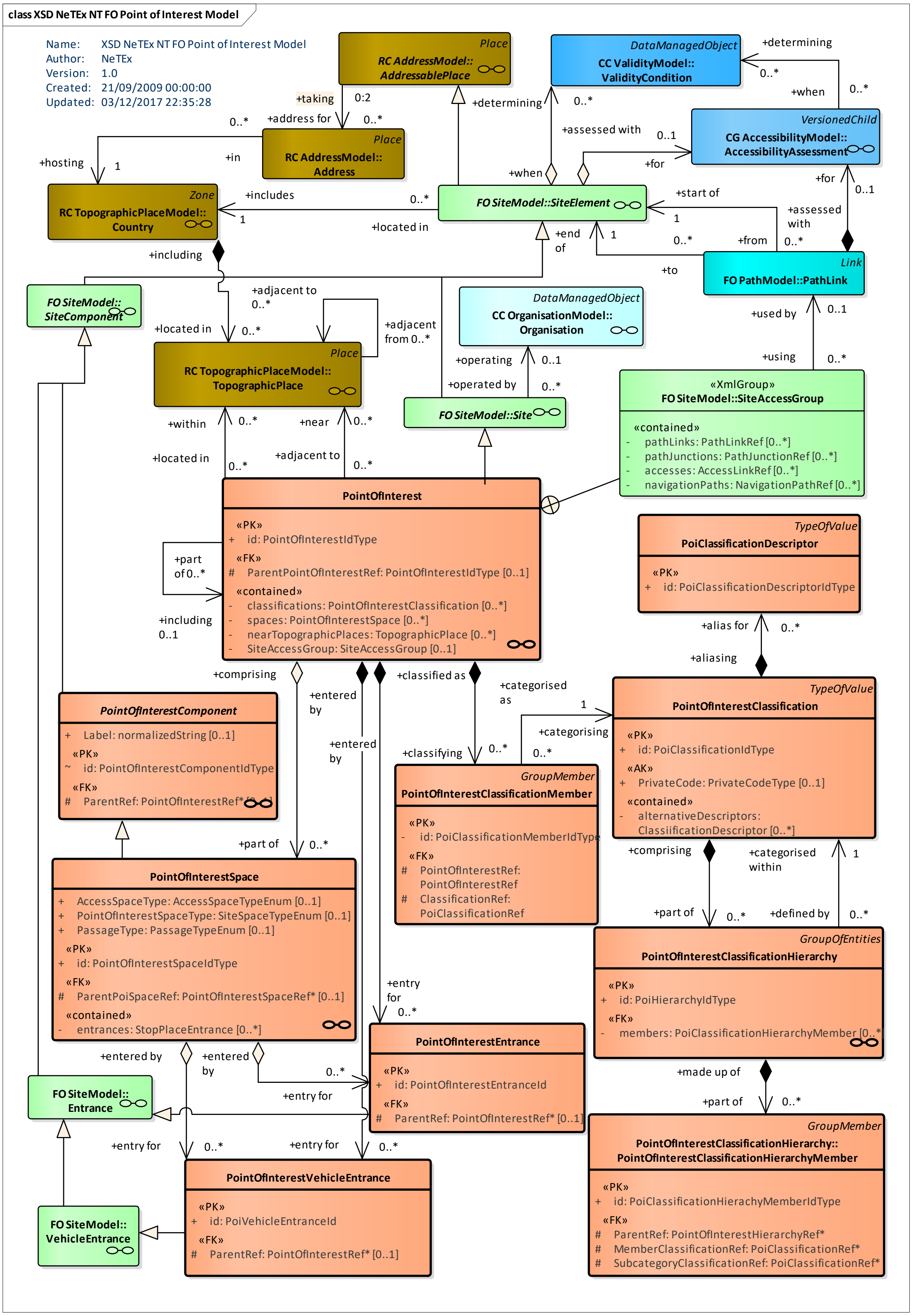


Figure 546 – Point Of Interest – Physical Model (UML)

8.4.6.4 Point of Interest – Attributes and XSD

8.4.6.4.1 PointOfInterest – Model Element

A type of PLACE to or through which passengers may wish to navigate as part of their journey and which is modelled in detail by journey planners.

A POINT OF INTEREST may have a complex spatial substructure with designated POINT OF INTEREST ENTRANCEs as well as access pathways described using PATH LINKs. A journey planner will normally provide an optimised route from a STOP PLACE to a POINT OF INTEREST ENTRANCE using a NAVIGATION PATH comprising one or more PATH LINKs IN SEQUENCE.

Table 449 – *PointOfInterest* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Site*	*::>*	POINT OF INTEREST inherits from SITE.
«PK»	***id***	*PointOfInterestIdType*	1:1	Identifier of POINT OF INTEREST.
«FK»	***ParentPointOfInterestRef***	*PointOfInterestRef*	0:1	Reference to parent POINT OF INTEREST of which this is part.
«cntd»	***classifications***	*PointOfInterest-Classification*	0:*	Classification of the POINT OF INTEREST.
«cntd»	***spaces***	*PointOfInterestSpace*	0:*	POINT OF INTEREST SPACEs within the POINT OF INTEREST.
«cntd»	***nearTopographicPlaces***	*TopographicPlace*	0:*	TOPOGRAPHIC PLACEs near the POINT OF INTEREST.
XGRP	***SiteAccessGroup***	***xmlGroup***	0:1	ACCESS SPACEs within the POINT OF INTEREST, i.e. STOP PLACE COMPONENTs that are not QUAYs, BOARDING POSITIONS, or ENTRANCEs. See above.

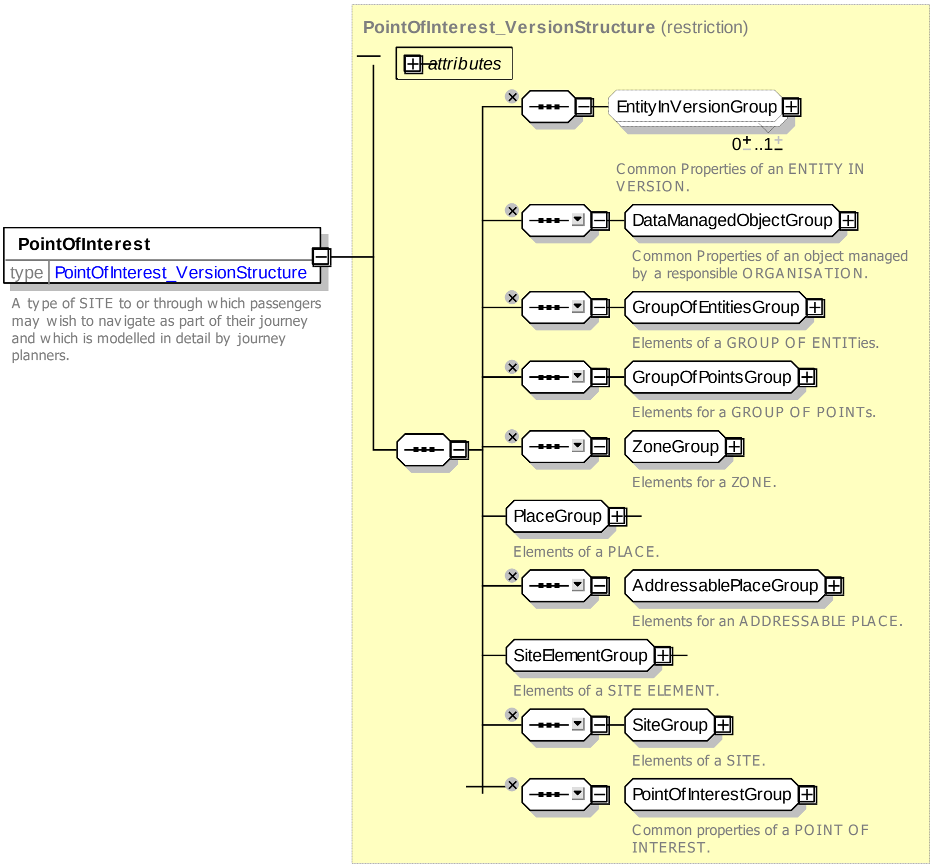


Figure 547 – *PointOfInterest – XSD*

8.4.6.4.1.1 PointOfInterestGroup – XML Group

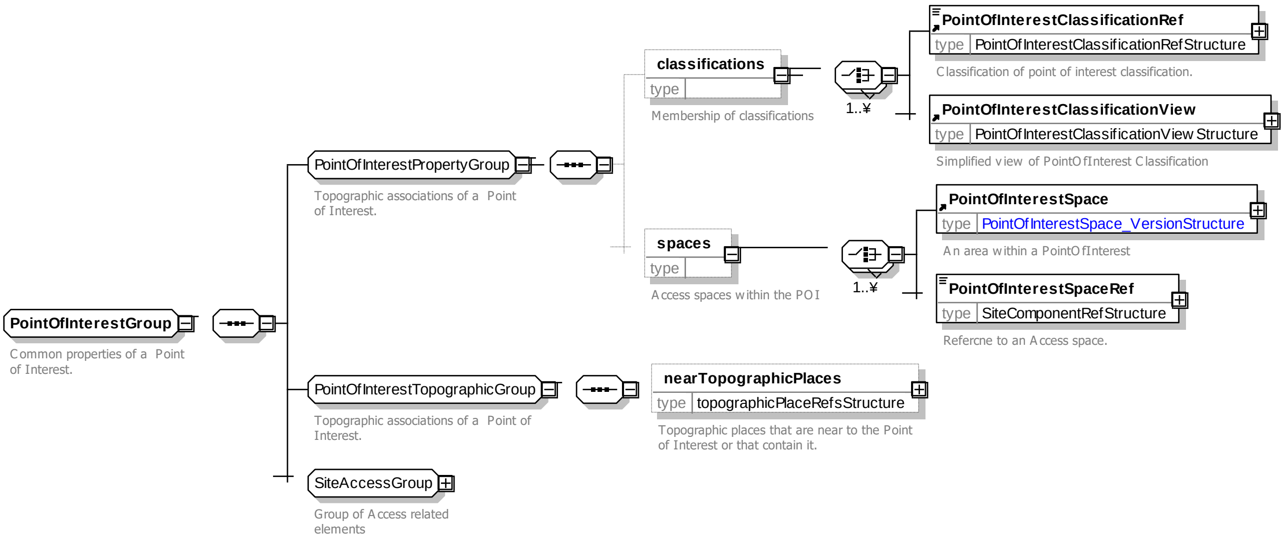


Figure 548 – *PointOfInterestGroup – XSD*

8.4.6.4.2 PointOfInterestComponent – Model Element

A part of the physical structure of a POINT OF INTEREST.

Table 450 – *PointOfInterestComponent* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*SiteComponent*	::>	POINT OF INTEREST COMPONENT inherits from SITE COMPONENT.
«PK»	***id***	*PointOfInterestComponent-IdType*	1:1	Identifier of POINT OF INTEREST COMPONENT.
«FK»	***ParentRef***	*PointOfInterestRef*	0:1	Reference to a parent POINT OF INTEREST.

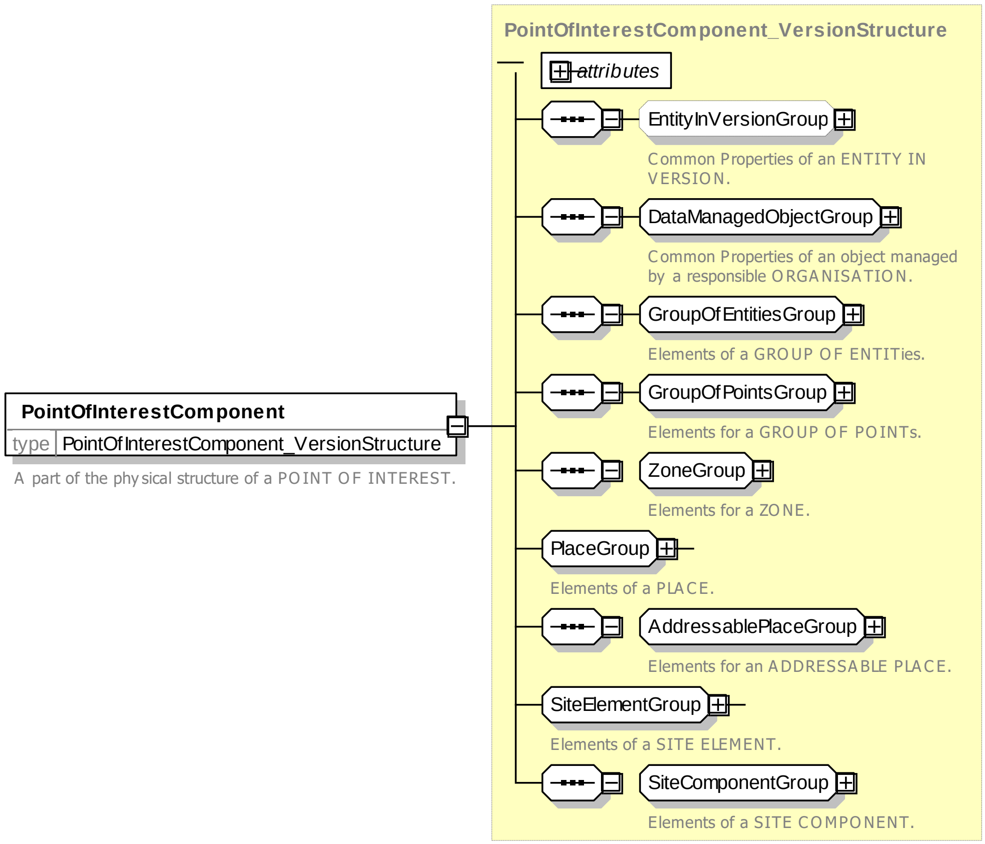


Figure 549 – *PointOfInterestComponent – XSD*

8.4.6.4.3 PointOfInterestSpace – Model Element

An area within a POINT OF INTEREST such as a concourse.

Table 451 – *PointOfInterestSpace* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PointOfInterestComponent*	::>	POINT OF INTEREST ACCESS SPACE inherits from POINT OF INTEREST COMPONENT.

«PK»	***id***	*PointOfInterestSpace-IdType*	1:1	Identifier of POINT OF INTEREST ACCESS SPACE.
«enum»	***AccessSpaceType***	*AccessSpaceTypeEnum*	0:1	Type of ACCESS SPACE. See allowed values below.
«enum»	***PointOfInterest-SpaceType***	*PointOfInterestSpaceType-Enum*	0:1	Type of POINT of INTEREST SPACE. See allowed values below.
«enum»	***PassageType***	*PassageTypeEnum*	0:1	Type of passage associated with ACCESS SPACE. See allowed values below.
«FK»	***ParentPoi-SpaceRef***	*PointOfInterestSpaceRef*	0:1	Reference to parent ACCESS SPACE within which this space is contained.
«cntd»	***entrances***	*StopPlaceEntrance*	0:*	ENTRANCES to POINT OF INTEREST.

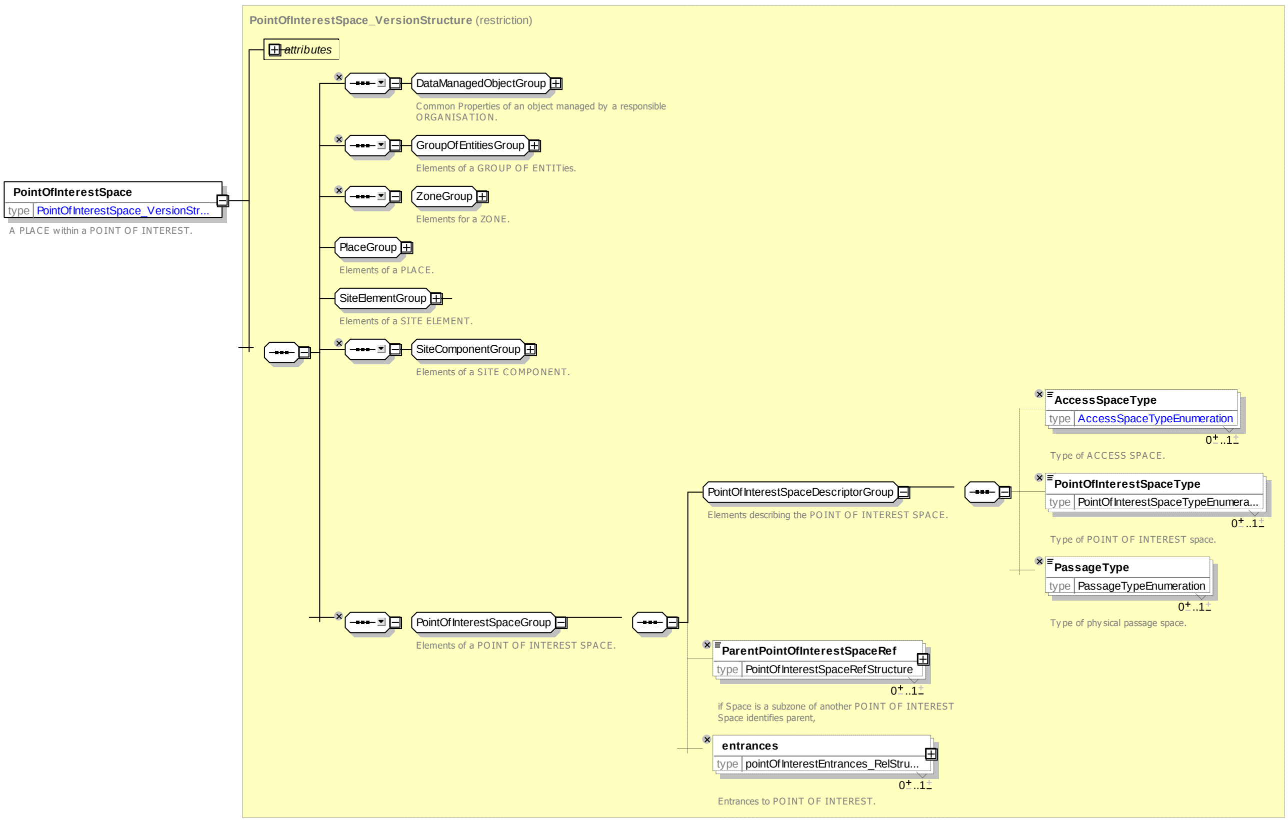


Figure 550 – *PointOfInterestSpace – XSD*

8.4.6.4.3.1 PointOfInterestSpaceType – Allowed values

The following table shows the allowed values for ***PointOfInterestSpaceType*** ***(****PointofInterestType-Enumeration).*

Table 452 – *PointOfInterestSpaceType* – Allowed values

Value	Description
arena	Arena.
archeryArena	Archery arena.
athleteArea	Athlete area.

auditorium	Auditorium.
changingRoom	Changing room.
court	Court.
downhillSkiingCourse	Downhill skiing Course.
freestyleSkiingCourse	Freestyle Skiing Course.
skiboardingArea	Skiboarding area.
gates	Gates.
greenRoom	Green room.
hospitalityZone	Hospitality zone.
iceRink	Ice rink.
orchestralPit	Orchestral pit.
playingField	Playing field.
podium	Podium.
pool	Pool.
divingPool	Diving pool.
pressArea	Press area.
queuingAreaForEntrance	Queuing area for Entrance.

ridingArea	Riding area.
rowingArea	Rowing area.
securityScreeningArea	Security Screening Area.
sledRun	Sled run.
spectatorTerrace	Spectator terrace.
spectatorSeating	Spectator seating.
spectatorStandingArea	Spectator Standing Area.
sportsArea	Sports area.
stabling	Stabling.
stage	Stage.
ring	Ring.
ticketing	Ticketing.
track	Track.
trackside	Trackside.
velodrome	Velodrome.
warmUpArea	Warm up area.
waterside	Waterside.
other	Other.

8.4.6.4.4 *PointOfInterestEntrance* – Model Element

A physical entrance or exit to/from a POINT OF INTEREST for passengers not in a vehicle. May be a door, barrier, gate or other recognizable point of access.

Table 453 – *PointOfInterestEntrance* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*SiteEntrance*	::>	POINT OF INTEREST ENTRANCE inherits from SITE ENTRANCE.
«PK»	***id***	*PointOfInterestEntranceId*	1:1	Identifier of a POINT OF INTEREST ENTRANCE.
«FK»	***ParentRef***	*PointOfInterestRef*	0:1	Parent POINT OF INTEREST of ENTRANCE.

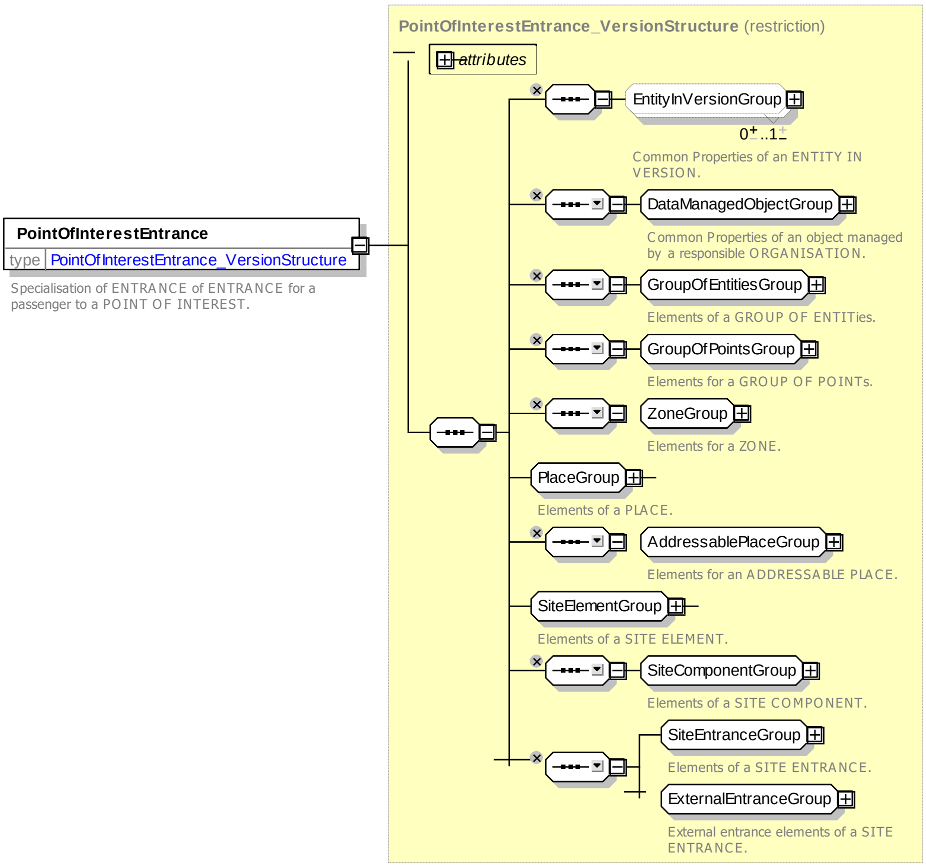


Figure 551 – *PointOfInterestEntrance – XSD*

8.4.6.4.5 PointOfInterestVehicleEntrance – Model Element

A physical entrance or exit to/from a STOP PLACE for vehicles

See 8.4.3.3.6 - VehicleEntrance – Model Element

8.4.6.4.6 PointOfInterestClassification – Model Element

A classification of a POINT OF INTEREST that may be used in a CLASSIFICATION HIERARCHY.

Table 454 – *PointOfInterestClassification* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	POINT OF INTEREST CLASSIFICATION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*PoiClassificationIdType*	1:1	Identifier of a POINT OF INTEREST CLASSIFICATION.
«AK»	***PrivateCode***	*xsd:normalizedString*	0:1	Private Code of a POINT OF INTEREST CLASSIFICATION.
«cntd»	***alternative-Descriptors***	*ClassiificationDescriptor*	0:*	Alternative descriptors of a POINT OF INTEREST CLASSIFICATION.

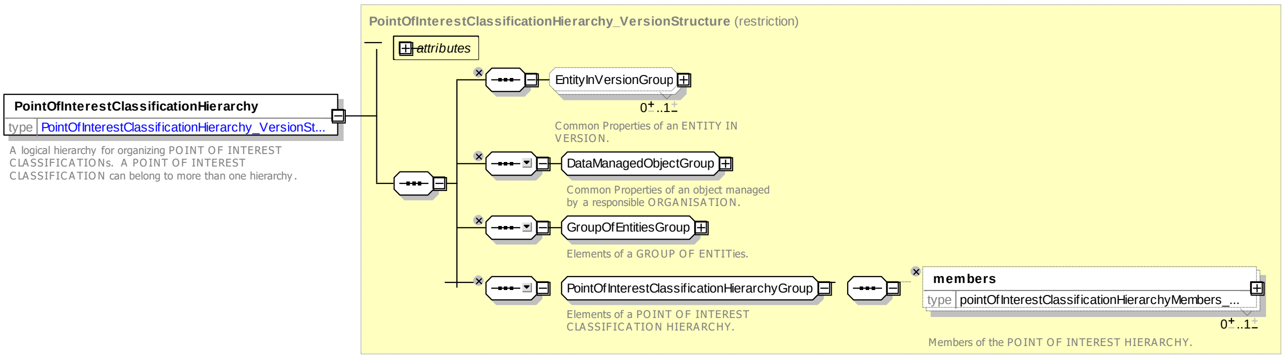


Figure 552 – *PointOfInterestClassification – XSD*

8.4.6.4.7 *PointOfInterestClassificationHierarchy*

A logical hierarchy for organizing POINT OF INTEREST CLASSIFICATIONs.

Table 455 – *PointOfInterestClassificationHierarchy* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*GroupOfEntities*	::>	POINT OF INTEREST CLASSIFICATION HIERARCHY inherits from GROUP OF ENTITies.
«PK»	***id***	*PoiHierarchyIdType*	1:1	Identifier of a POINT OF INTEREST CLASSIFICATION HIERARCHY.
«FK»	***members***	*PoiClassificationHierarchyMember*	0:*	POINT OF INTEREST CLASSIFICATION HIERARCHY MEMBERs making up a POINT OF INTEREST CLASSIFICATION HIERARCHY.

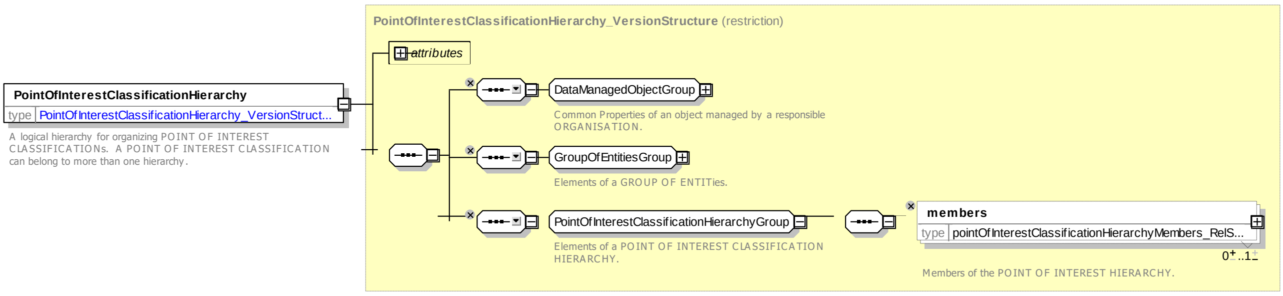


Figure 553 – *PointOfInterestClassificationHierarchy – XSD*

8.4.6.4.8 PointOfInterestClassificationHierarchyMember

A member of a POINT OF INTEREST CLASSIFICATION HIERARCHY that locates a particular POINT OF INTEREST CLASSIFICATION in a particular CLASSIFICATION HIERARCHY.

Table 456 – *PointOfInterestClassificationHierarchyMember* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*VersionedChild*	::>	POINT OF INTEREST CLASSIFICATION HIERARCHY MEMBER inherits from VERSIONED CHILD.
«PK»	***id***	*PoiClassificationHierarchyMemberIdType*	1:1	Identifier of a POINT OF INTEREST CLASSIFICATION HIERARCHY MEMBER.
«FK»	***ParentRef***	*PointOfInterestHierarchyRef*	0:1	Parent POINT OF INTEREST HIERARCHY to which a POINT OF INTEREST CLASSIFICATION MEMBER belongs.
«FK»	***MemberClassificationRef***	*PoiClassificationRef*	1:1	POINT OF INTEREST CLASSIFICATION that is super classification defined by this HIERARCHY MEMBER.
«FK»	***SubcategoryClassificationRef***	*PoiClassificationRef*	1:1	POINT OF INTEREST CLASSIFICATION that is subsidiary to Member Classification in the CLASSIFICATION HIERARCHY.

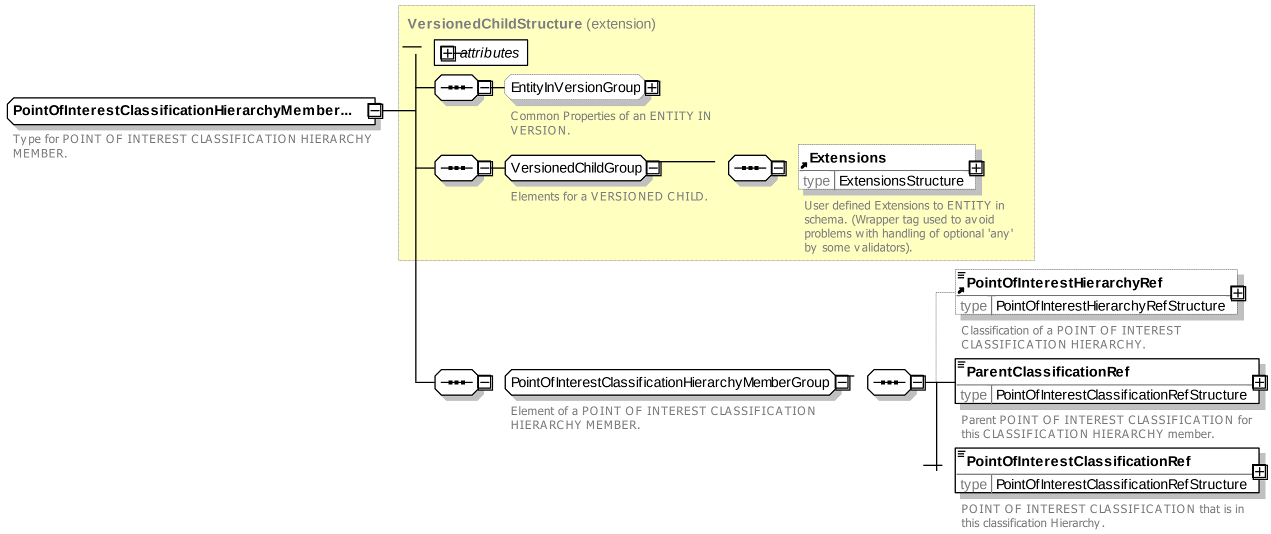


Figure 554 – *PointOfInterestClassificationHierarchyMember – XSD*

8.4.6.5 Point of Interest – XML Example

The following XML code fragment shows part of a POINT OF INTEREST for a Stadium that identifies the pixel coordinates for two ENTRANCEs the entrances**.**

EXAMPLE

```
<PointOfInterest>
    <Name>Frampton Football Stadium</Name>
    <Description>normalizedString</Description>
    <Centroid>
        <Location>
            <Longitude>-180</Longitude>
            <Latitude>-90</Latitude>
            <Altitude>-1000</Altitude>
            <Precision>0.0</Precision>
```

```xml
        </Location>
    </Centroid>
    <AccessibilityAssessment>
        <MobilityImpairedAccess>true</MobilityImpairedAccess id="ACID_21">
        <limitations>
            <AccessibilityLimitation>
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <accessModes>
        <AccessMode>foot</AccessMode>
    </accessModes>
    <ShortName>Frampton FC</ShortName>
    <Image>http://www.amy.com/stad.gif</Image>
    <CrossRoad>Foo St</CrossRoad>
    <PublicUse>true</PublicUse>
    <Covered>indoors</Covered>
    <Gated>gatedArea</Gated>
    <Lighting>wellLit</Lighting>
    <TopographicPlaceRef>678</TopographicPlaceRef>
    <PostalAddress id="normalizedString">
        <BuildingName>normalizedString</BuildingName>
        <AddressLine1>23 Foo St</AddressLine1>
        <Town>Frampton</Town>
        <Suburb>normalizedString</Suburb>
        <PostCode>FgR 457</PostCode>
    </PostalAddress>

    <levels>
        <Level id="tbd:POI_23_Lvl_01">
            <Name>Ground</Name>
            <ShortName>Ground</ShortName>
            <LevelCode>G</LevelCode>
        </Level>
        <Level id=" tbd:POI_23_Lvl_02">
            <Name>Upper Terrace</Name>
            <ShortName>Upper</ShortName>
            <LevelCode>T1</LevelCode>
        </Level>
    </levels>
    <entrances>
        <Entrance id="tbd:POI_23_A1_E1">
            <Name>Foo St Entrance A</Name>
```

8.4.7 Associating Equipment with Places

EQUIPMENT can be located within a SITE with an EQUIPMENT PLACE using both relative (e.g. 6m along a link) or absolute (e.g. WGS coordinates). In many cases it is sufficient just to associate equipment with a SITE COMPONENT. In other cases it is useful to give a precise location.

Certain types of equipment are LOCAL SERVICEs that are not placed but rather associated with the STOP PLACE as a whole.

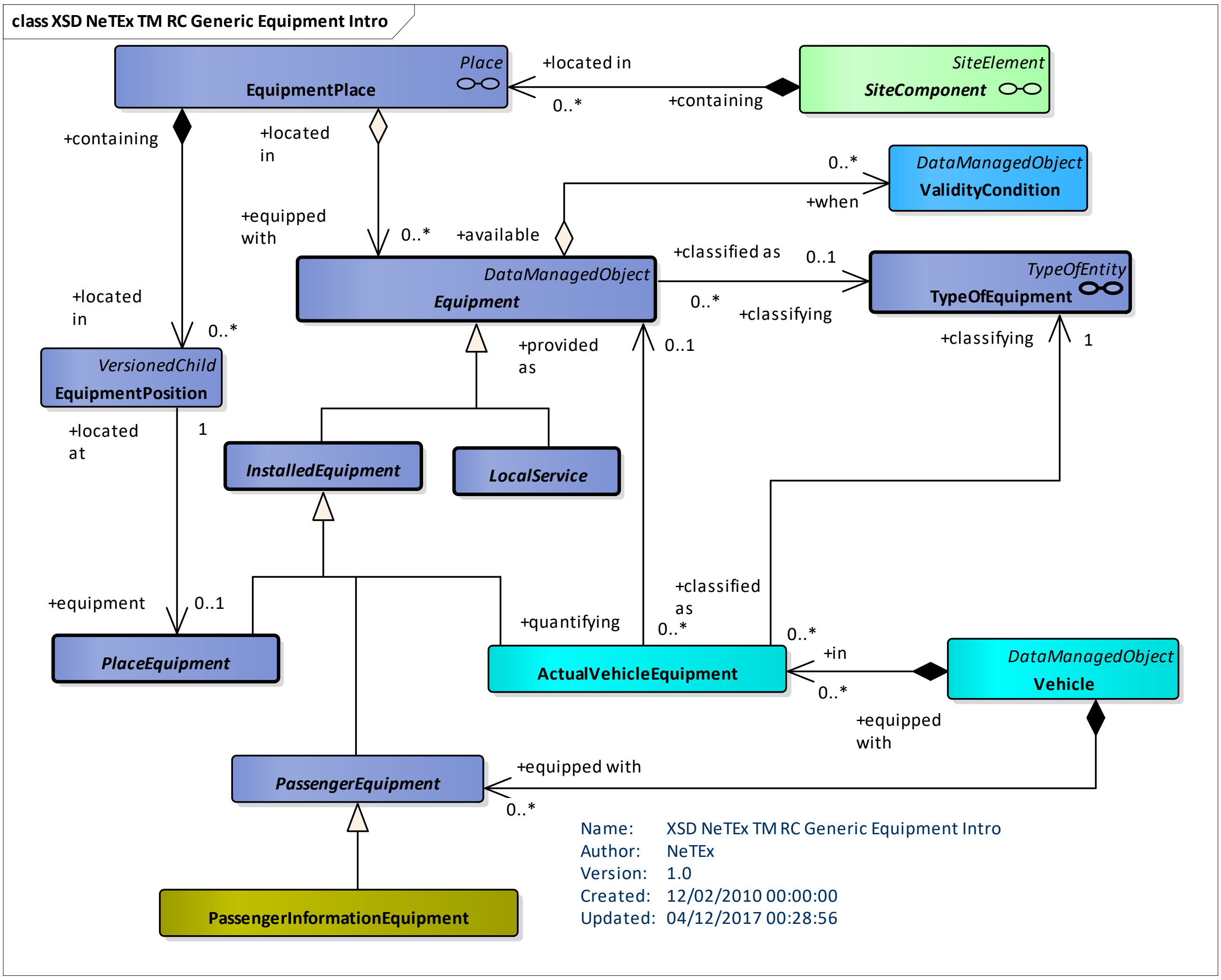


Figure 555 – Equipment & Places – Physical Model (UML)

8.4.8 Equipment Description

8.4.8.1 Site Equipment

The topological components of a SITE (that is constituent parts of STOP PLACES, POINTs of INTERET etc.), such as QUAYs, ACCESS SPACES, ENTRANCEs and PATH LINKs can be annotated with equipment, images and other attributes that describe the detailed properties of the interchange, for example, lifts, or stairs, ticket barriers, surfaces, and their accessibility properties. *NeTEx* provides a systematic set of standardised EQUIPMENT objects, describing different types of equipment including ticket machines, doors, gates, ramps, seats, phones and information displays with standardised attributes.

8.4.8.2 Equipment Description – Model Dependencies

The Equipment Description Models describe the EQUIPMENT that can be associated with STOP PLACEs.

The submodels themselves depend on a number of general NeTEx framework models (for example, GENERIC EQUIPMENT, VEHICLE TYPES, EQUIPMENT PLACE) described elsewhere – See REUSABLE COMPONENTs.

The following figure shows the dependencies between the EQUIPMENT:

— SITE EQUIPMENT MODEL: models a site specific EQUIPMENT.

— PASSENGER SERVICE: EQUIPMENT MODEL for passenger services.

— ACCESS: EQUIPMENT MODEL for access to a SITE.

— TICKETING EQUIPMENT: models ticket validation equipment.

— SIGN EQUIPMENT: models sign EQUIPMENT.

— PARKING EQUIPMENT: Parking EQUIPMENT.

— COMMERCIAL SERVICE: EQUIPMENT for commercial services.

— VEHICLE PASSENGER EQUIPMENT: on board passenger equipment for vehicles.

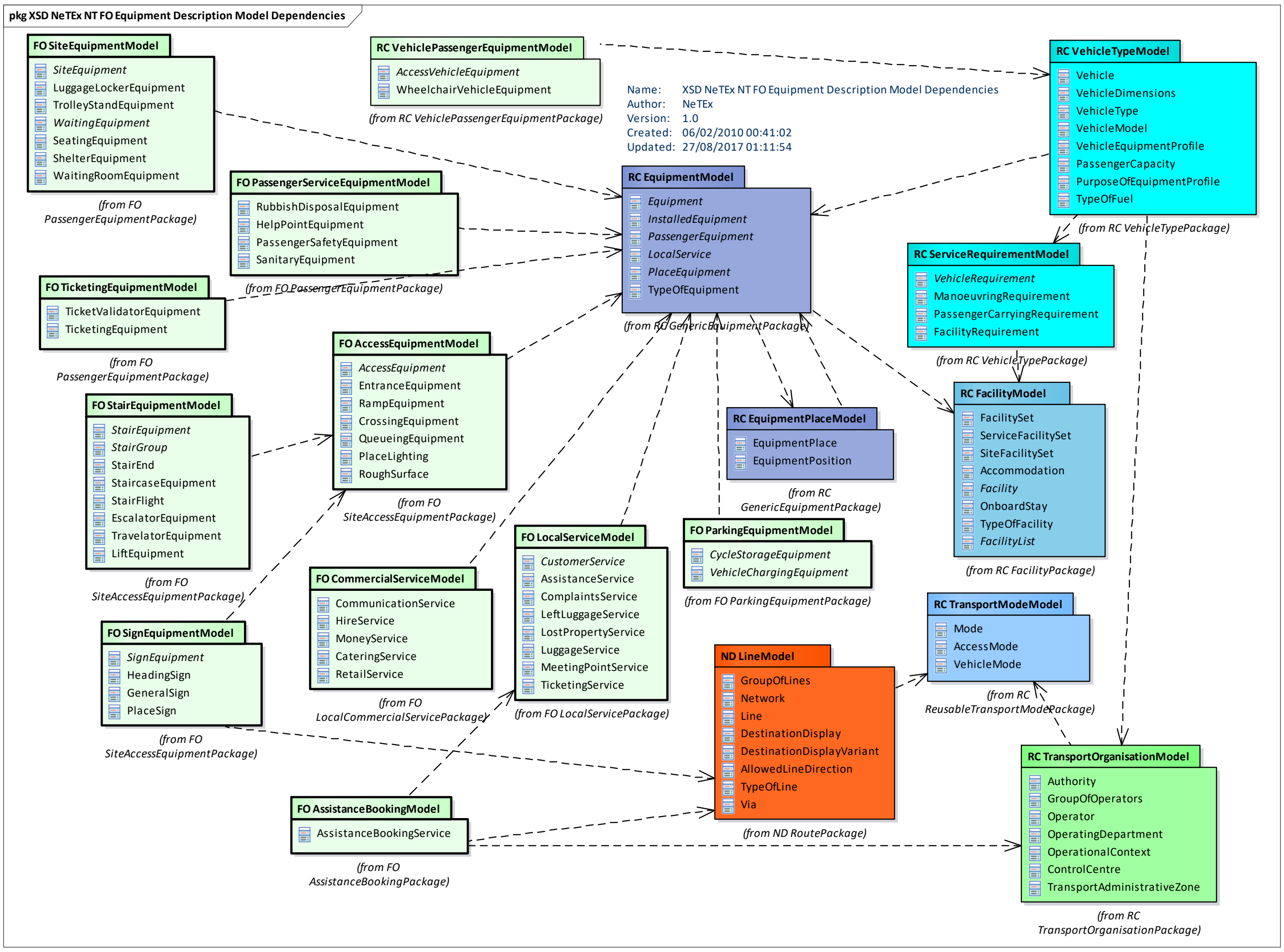


Figure 556 – Fixed objects – Equipment Model dependencies (UML)

8.4.8.3 Site Equipment – Examples

EQUIPMENT and LOCAL SERVICE items can be used to adorn a schematic map or other interactive visualisation of a SITE on a mobile or fixed device.

Some examples of using equipment data are shown in the following figure.

Figure 557 – Equipment Example Hover windows for Equipment (NRE Stations Made Easy)

Each EQUIPMENT type will typically have an icon associated with it.

Table 457 – Equipment Example – NRE Station Made Easy: Legend (From Euston Station)

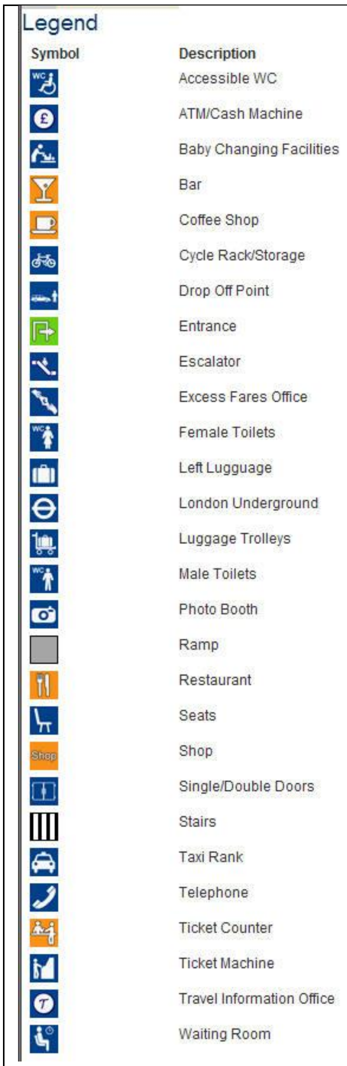


Figure 558

Facility	Element	Attribute
Accessible WC	***SanitaryEquipment***	wheelchairAccessible
Cash Machine	***CateringService***	cashMachine
Baby Changing	***SanitaryEquipment***	babyChange
Bar	***CateringService***	bar
Coffee Shop	***CateringService***	coffeeShop
Cycle Rack	***CycleStorageEquipment***	racks
Drop Off Point	QUAY + ***SignEquipment***	setDown
Entrance	ENTRANCE + ***SignEquipment***	entrance
Escalator	***EscalatorEquipment***	
Excess Fares	***TicketingEquipment***	excessFares
Female Toilets	***SanitaryEquipment***	femaleOnly
LeftLuggage	***LuggageLockerEquipment***	
Metro	ENTRANCE +***SignEquipment***	
Luggage Trolleys	***TrolleyStandEquipment***	
Male Toilets	***SanitaryEquipment***	maleOnly
Photo Booth	***RetailService***	photoBooth
Ramp	***RampEquipment***	
Restaurant	***CateringService***	restaurant
Seats	***WaitingRoomEquipment***	seats
Shop	***RetailEquipment***	
Doors	***EntranceEquipment***	

	Stairs	***StairEquipment***	
	Taxi	QUAY + S***ignEquipment***	
	Telephone	***CommunicationService***	phone
	Ticket Counter	***TicketingEquipment***	counterService
	Ticket Machine	***TicketingEquipment***	TicketMachines
	Info Office	***AssistanceService***	information
	Waiting Room	***WaitingRoomEquipment***	
	Lost Property	***LostPropertyService***	
	Internet	***CommunicationService***	internet
	Postbox	***CommunicationService***	postbox

8.4.8.4 Types of Equipment

The different types of equipment are summarised in the following diagram and listed in the following figure. Each of these may have specific attributes.

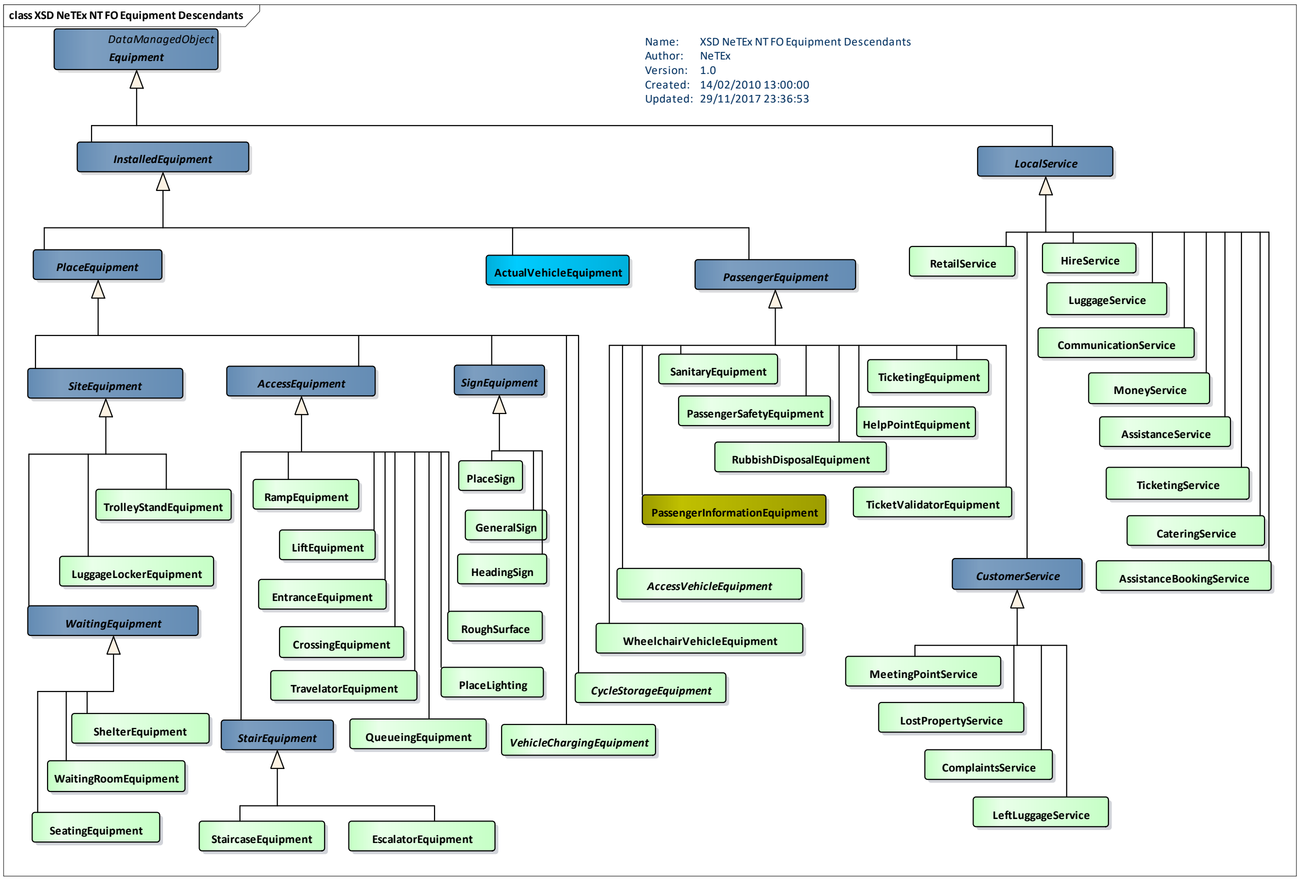


Figure 559 – Overview of Equipment types – (UML)

The following table shows the different types of EQUIPMENT. The "Accessibility attributes" column lists specific properties of ***Equipment*** that are relevant for accessibility.

Table 458 – Summary of NeTEx Equipment types relevant for Accessibility

Group	Subgroup	Equipment	Accessibility attributes
PlaceEquipment	AccessEquipment	***RoughSurface***	SurfaceType.
		EntranceEquipment	Dimensions, wheelchair passable, controls, acoustic sensor, automatic.
		StaircaseEquipment	Handrail, handrail height, step height, number of steps.
		LiftEquipment	Dimensions, wheelchair passable, wheelchair turning circle.
		EscalatorEquipment	Width.
		TravelatorEquipment	Width.
		RampEquipment	Dimensions, gradient, handrail, bands, strips.

		QueuingEquipment	
		CrossingEquipment	Strips, sounds, sensors, acoustic aids dropped curb.
	SignEquipment	***StopPlaceSign***	A sign giving the stop name.
		HeadingSign	
		GeneralSign	
	Ticketing	***TicketingEquipment***	Low counter access.
		TicketValidatorEquipment	
	StopPlace	***LuggageLockerEquipment***	
		ShelterEquipment	Number of seats, dimensions, StepFree, wheelchair area width, wheelchair area depth.
		TrolleyStandEquipment	
		WaitingRoomEquipment	Number of seats, dimensions, step free, wheelchair area width, wheelchair area depth.
	PassengerEquipment	***PassengerInfoEquipment***	Accessibility Info
		PassengerSafetyEquipment	ccTV, Panic button, SOS Phones, Height of SOS Panel, Acoustic Announcements.
		SanitaryFacility	Gender, Type Of Sanitary Facility Wheelchair turning circle.
Local Service	Customer	***AssistanceService***	Services to help customers board. etc.
		MeetingpointService	

8.4.8.5 Equipment Properties

The different types of equipment make use of common enumerations.

8.4.8.6 Passenger Service Equipment Model

8.4.8.6.1 PASSENGER SERVICE EQUIPMENT – Conceptual MODEL

There are three types of PASSENGER SERVICE EQUIPMENT, PASSENGER SAFETY EQUIPMENT, describing help points etc.; SANITARY EQUIPMENT describing toilets and washing amenities, and RUBBISH DISPOSAL, describing bins for different categories of waste.

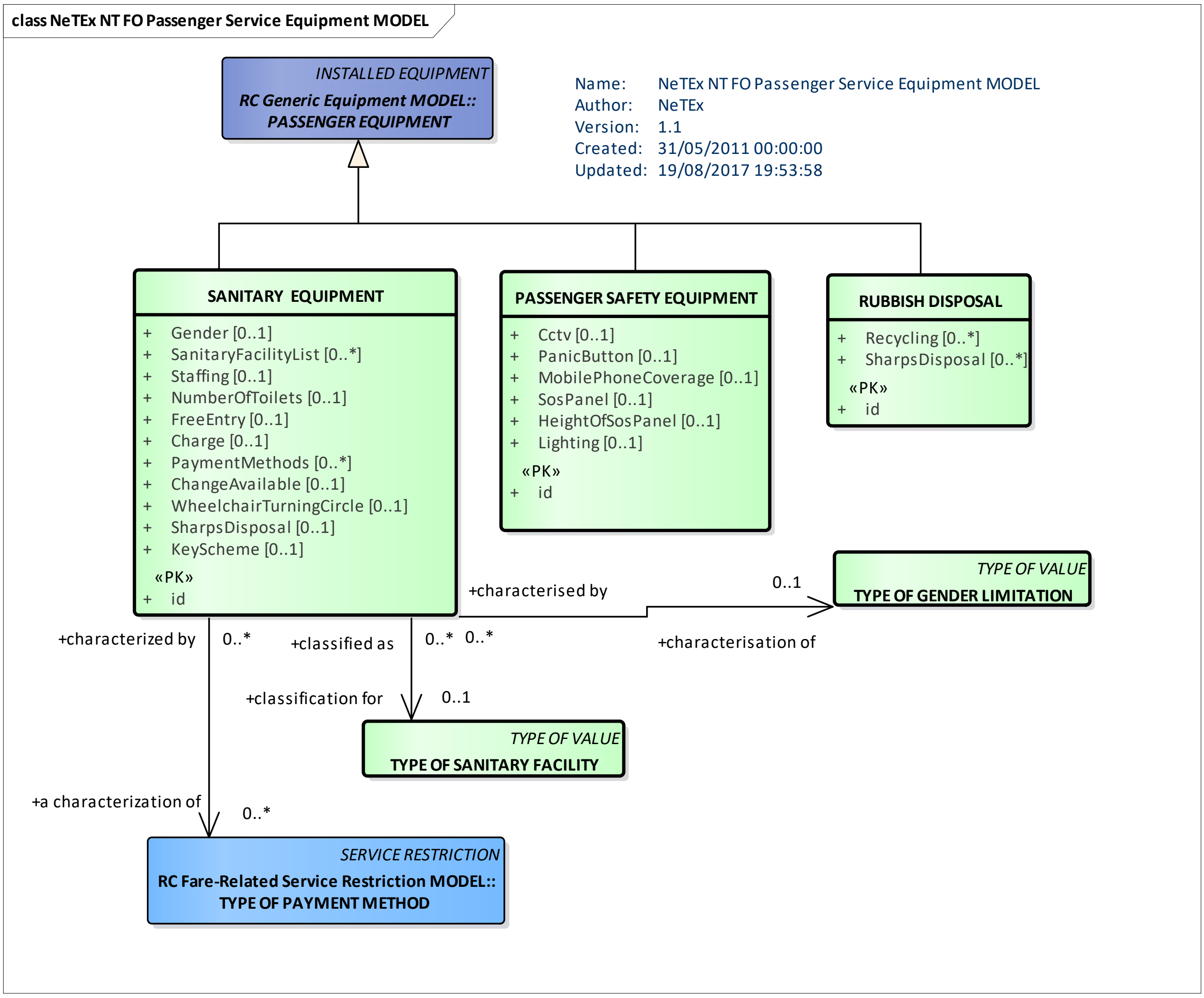


Figure 560 – Passenger Equipment – Conceptual MODEL (UML)

8.4.8.6.2 Passenger Service Equipment – Physical Model

The following figure shows the attributes of the PASSENGER SERVICE EQUIPMENT model elements.

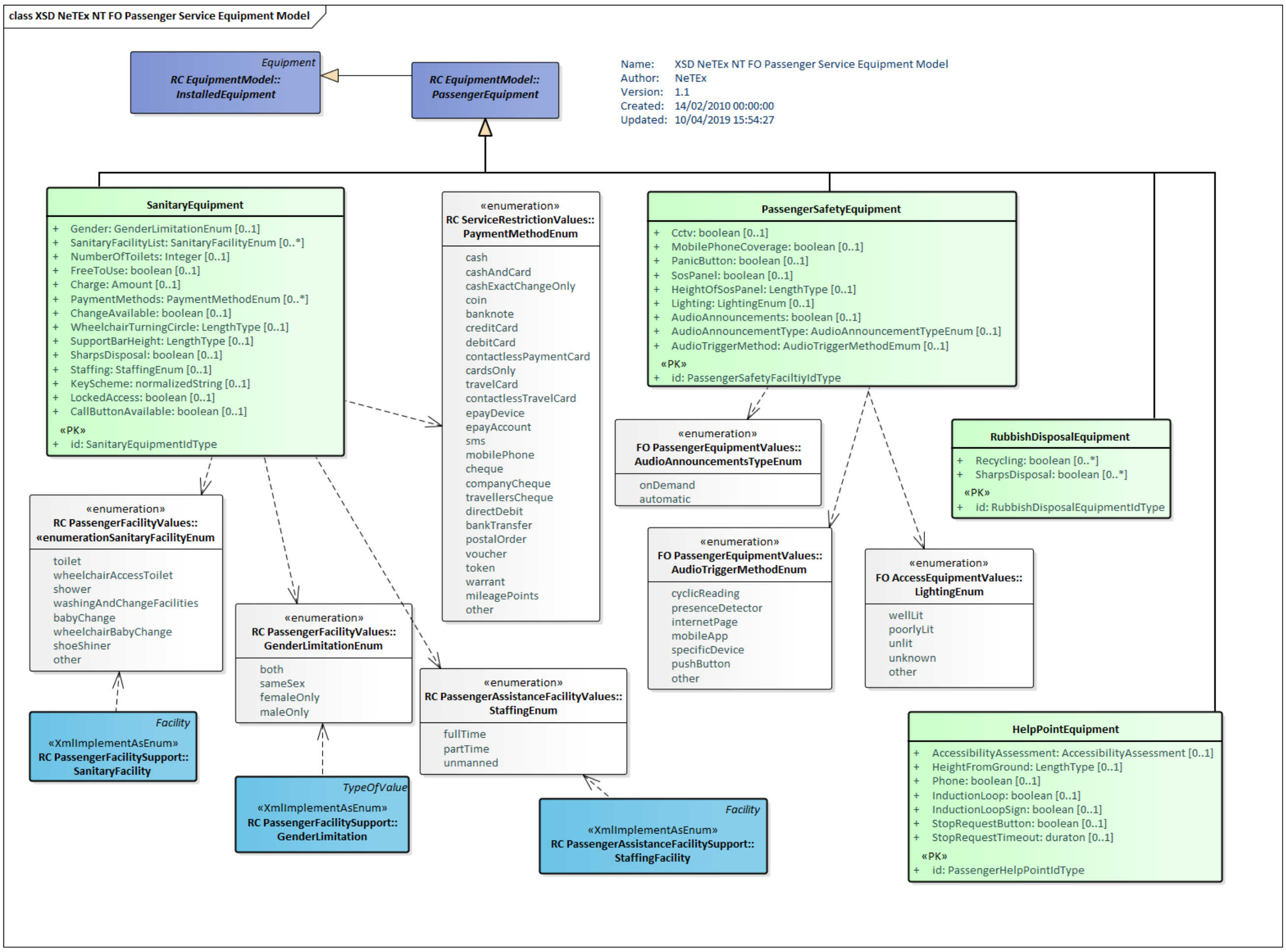


Figure 561 – Passenger Service Equipment – Physical Model (UML)

8.4.8.6.3 Passenger Service Equipment – Attributes and XSD

8.4.8.6.3.1 PassengerSafetyEquipment – Model Element

Specialisation of PASSENGER EQUIPMENT for PASSENGER SAFETY.

Table 459 – *PassengerSafetyEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PassengerEquipment*	::>	PASSENGER SAFETY EQUIPMENT inherits from PASSENGER EQUIPMENT.
«PK»	***id***	*PassengerSafetyFacility-IdType*	1:1	Identifier of PASSENGER SAFETY EQUIPMENT.
	Cctv	*xsd:boolean*	0:1	Whether there is CCTV coverage.
	MobilePhone-Coverage	*xsd:boolean*	0:1	Whether there is Mobile phone coverage.
	PanicButton	*xsd:boolean*	0:1	Whether there is a panic button.

	SosPanel	*xsd:boolean*	0:1	Whether there is a SoS Panel.
	HeightOfSosPanel	*LengthType*	0:1	Height of SoS panel above ground.
«enum»	***Lighting***	*LightingEnum*	0:1	Type of lighting. See allowed vaues under PLACE LIGHTING.
	Audio-Announcements	*xsd:boolean*	0:1	Whether there are audio announcements.
«enum»	***Audio-AnnouncementType***	*AudioAnnouncement-TypeEnum*	0:1	Type of audio announcement. See allowed values below. +v1.1.
«enum»	***AudioTriggerMethod***	*AudioTriggerMethod-Enum*	0:1	How audio announcements are triggered. See allowed values below. +v1.1.

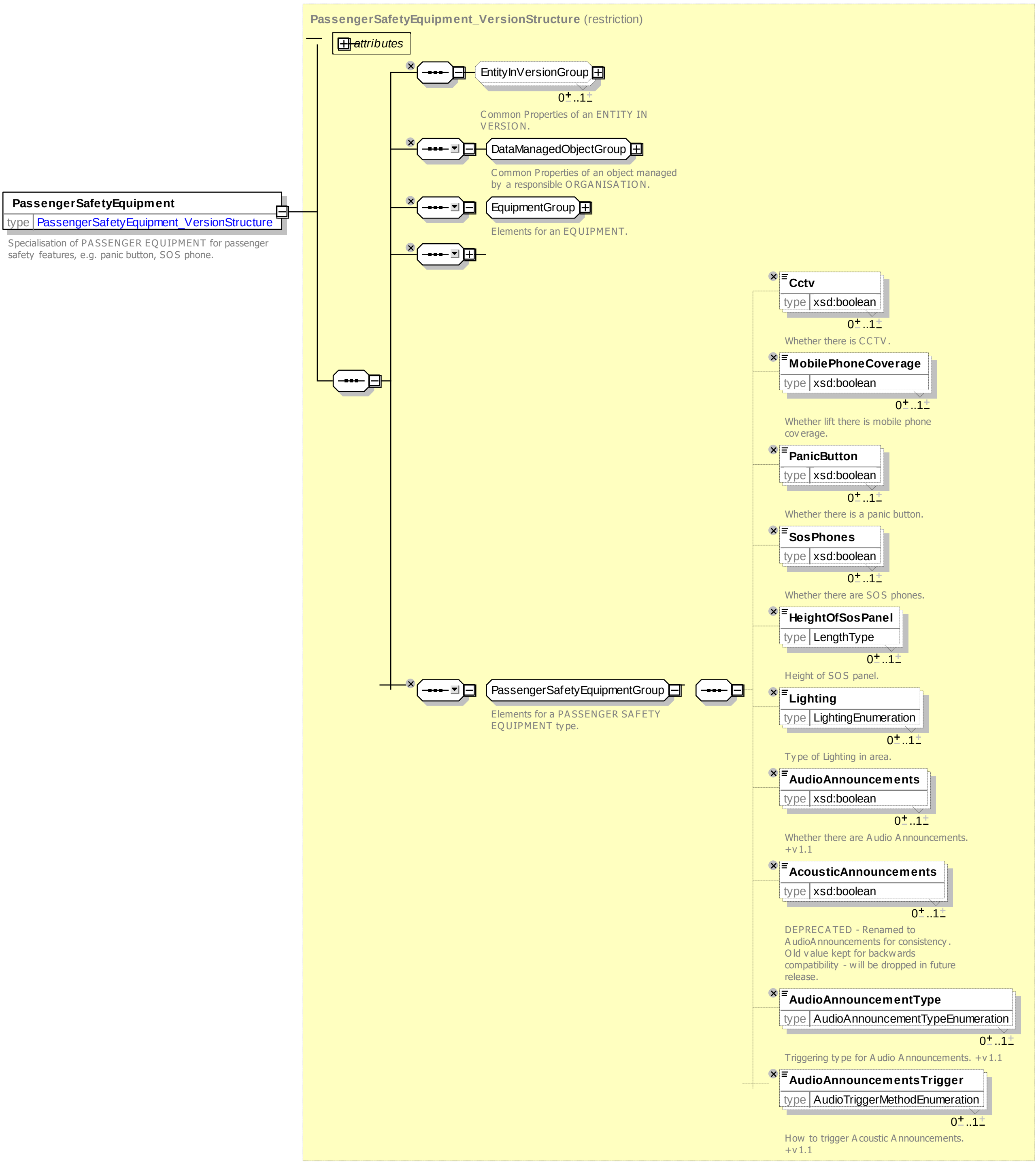


Figure 562 – *PassengerSafetyEquipment – XSD*

AudioAnnouncementType – Allowed values

Allowed values for ***AudioAnnouncementType*** (*AudiioAnnouncementTypeEnumeration*).

Table 460 – *AudioAnnouncementType* – Allowed values

Value	Description

onDemand	Annoucnements are made only on request
automatic	Announcements are made automatically.

AudioTriggerMethod – Allowed values

Allowed values for ***AudioTriggerMethod*** (*AudioTriggerMethodEnum*).

Table 461 – *AudioTriggerMethod* – Allowed values

Value	Description
presenceDetector	Automatic detection of presence
mobileApp	Request with mobile app
internetPage	Request with internet page
specificDevice	Request with a special device.
pushButton	Request by pushing a button
other	Other means

8.4.8.6.3.2 **SanitaryEquipment – Model Element**

Properties of WC and washing facilities.

Table 462 – *SanitaryEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PassengerEquipment*	*::>*	SANITARY EQUIPMENT inherits from PASSENGER EQUIPMENT.
«PK»	***id***	*SanitaryEquipment*«enum»	1:1	Identifier of SANITARY EQUIPMENT.
«cntd»	***Accessibility-Assessment***	*AccessibilityAssessment*	0:1	ACCESSIBILITY ASSESSMENT for SANITARY EQUIPMENT.
«enum»	***Gender***	*GenderLimitationEnum*	0:1	Gender limitation on use of facility. See allowed values below.
«enum»	***SanitaryFacility-List***	*SanitaryFacilityEnum*	0:*	Type of SANITARY FACILITY. See allowed values below.
	NumberOfToilets	*xsd:integer*	0:1	Number of toilets.
	FreeToUse	*xsd:boolean*	0:1	Whether entry is free.
	Charge	*Amount*	0:1	Charge for use.
	Currency	*CurrencyType*	0:1	Currency in which charge is made.
«enum»	***PaymentMethods***	*PaymentMethodEnum*	0:*	Method of payment. See allowed values in RC Service Restrictions Model.
	ChangeAvailable	*xsd:boolean*	0:1	Whether payment entry machine can give change.

	Wheelchair-TurningCircle	*LengthType*	0:1	Wheelchair turning circle in Toilet.
	SupportBarHeight	*LengthType*	0:1	Height of the support bar (when there is one). +v1.1
	CallButton-Available	*xsd:boolean*	0:1	Whether a call button is available. +v1.1
	SharpsDisposal	*xsd:boolean*	0:1	Whether there is sharps disposal.
«enum»	***Staffing***	*StaffingEnum*	0:1	Whether facility is staffed. See allowed values in Passenger Assistance Model.
	LockedAccess	*xsd:boolean*	0:1	Whether toilet may be locked end thus a key is needed (or an equivalent tool) to access.+v1.1
	KeyScheme	*xsd:normalizedString*	0:1	Key scheme under which facility is accessible.

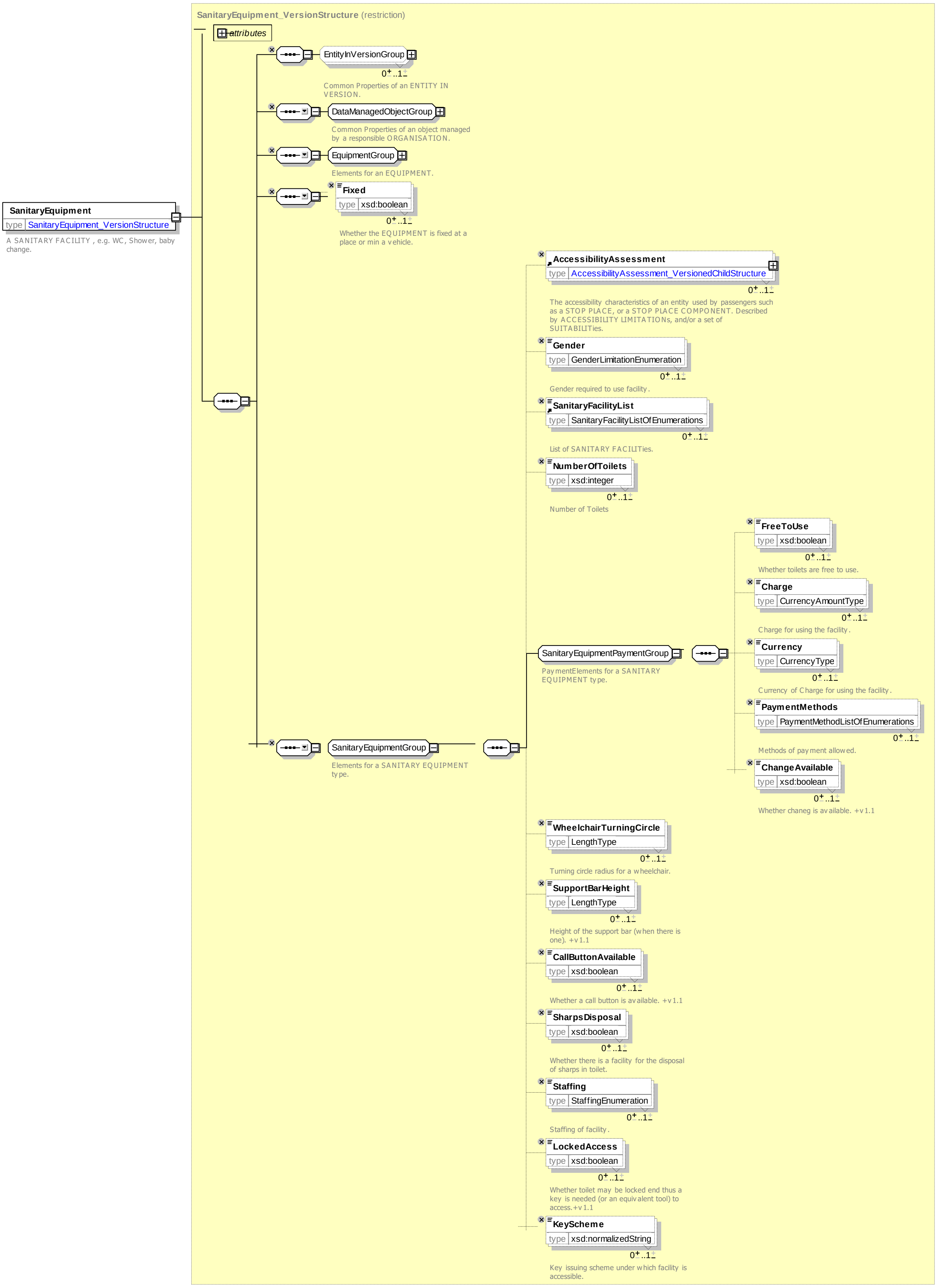


Figure 563 – *SanitaryEquipment – XSD*

GenderLimitation – Allowed values

Allowed values for ***GenderLimitation*** (*GenderLimitationEnumeration)*.

Table 463 –*GenderLimitation* – Allowed values

Value	Description
both	Both genders
maleOnly	Male only
femaleOnly	Female only

TypeOfSanitaryFacility – Allowed values

Allowed values for Type of SANITARY FACILITY (*TypeOfSanitaryFacilityEnumeration*).

Table 464 –*SanitaryFacilityType* – Allowed Values

Value	Description
toilet	Lavatories
wheelchairAccessToilet	Toilet for wheelchair access
shower	Shower
wheelchairBabyChange	Baby change facility for wheelchair users
babyChange	Baby change facility
washingAndChangeFacilities	Wash and Change facilities
other	Other sanitary facility

8.4.8.6.3.3 **RubbishDisposalEquipment – Model Element**

Specialisation of EQUIPMENT for Rubbish disposal, describing rubbish types, etc.

Table 465 – *RubbishDisposalEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PassengerEquipment*	*::>*	RUBBISH DISPOSAL EQUIPMENT inherits from PASSENGER EQUIPMENT
«PK»	***id***	*RubbishDisposalEquipmentIdType*	1:1	Identifier of RUBBISH DISPOSAL EQUIPMENT.
	Recycling	*xsd:boolean*	0:1	Whether there are rubbish separation and recycling facilities.
	SharpsDisposal	*xsd:boolean*	0:1	Whether there are disposal facilities for needles or medical sharps.

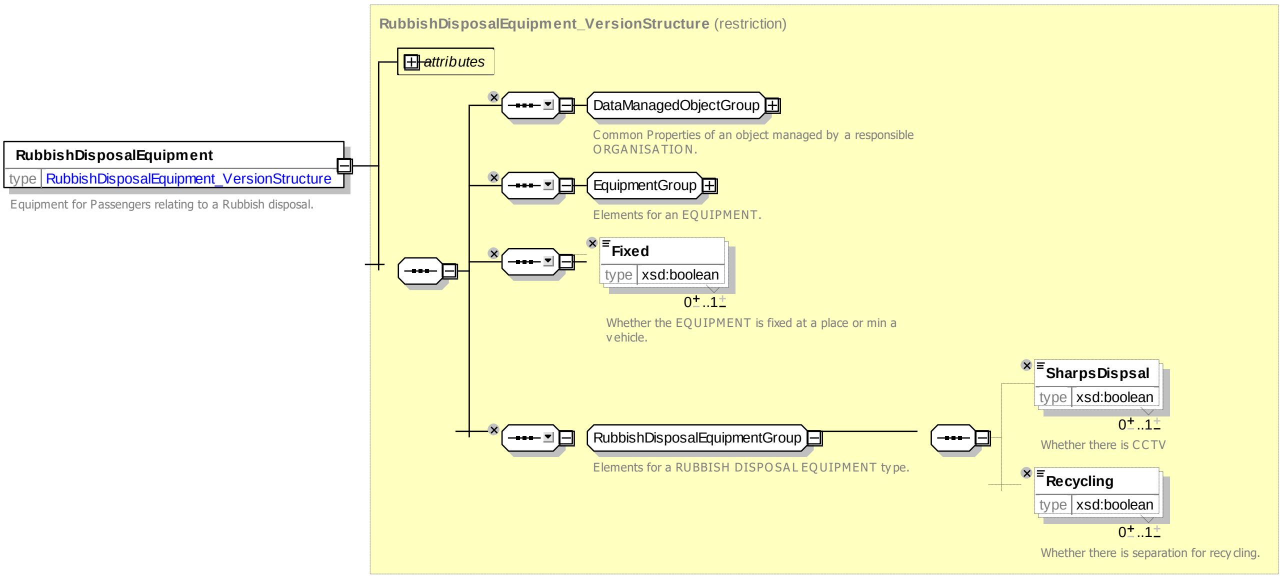


Figure 564 – *RubbishDisposalEquipment – XSD*

8.4.8.7 Waiting Equipment

8.4.8.7.1 WAITING EQUIPMENT – Conceptual MODEL

Various types of WAITING EQUIPMENT can be specified including SEATING, WAITING ROOMs.

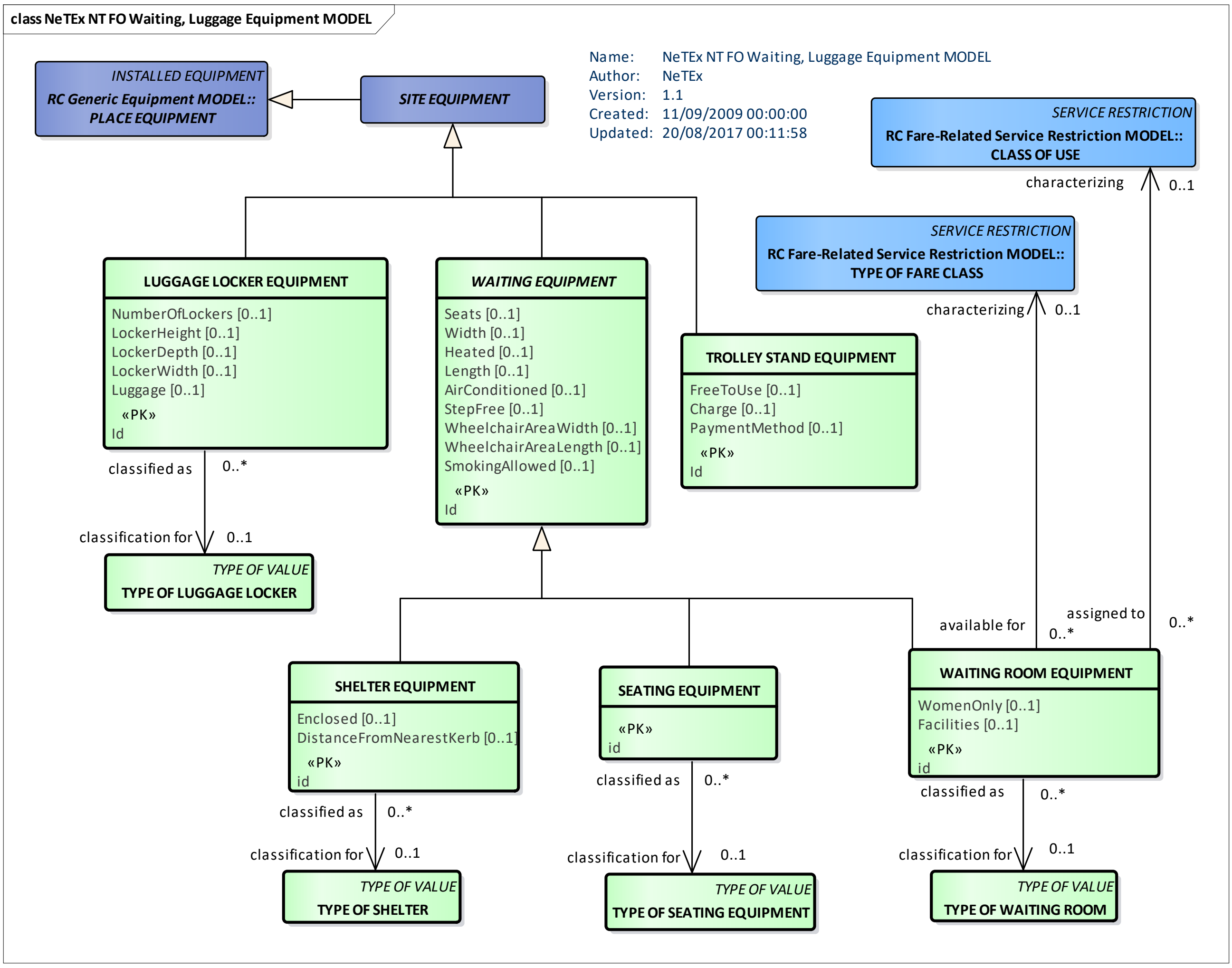


Figure 565 – WAITING EQUIPMENT – Conceptual MODEL (UML)

8.4.8.7.2 Waiting Equipment – Physical Model

The following figure shows detailed attributes of the WAITING EQUIPMENT model elements.

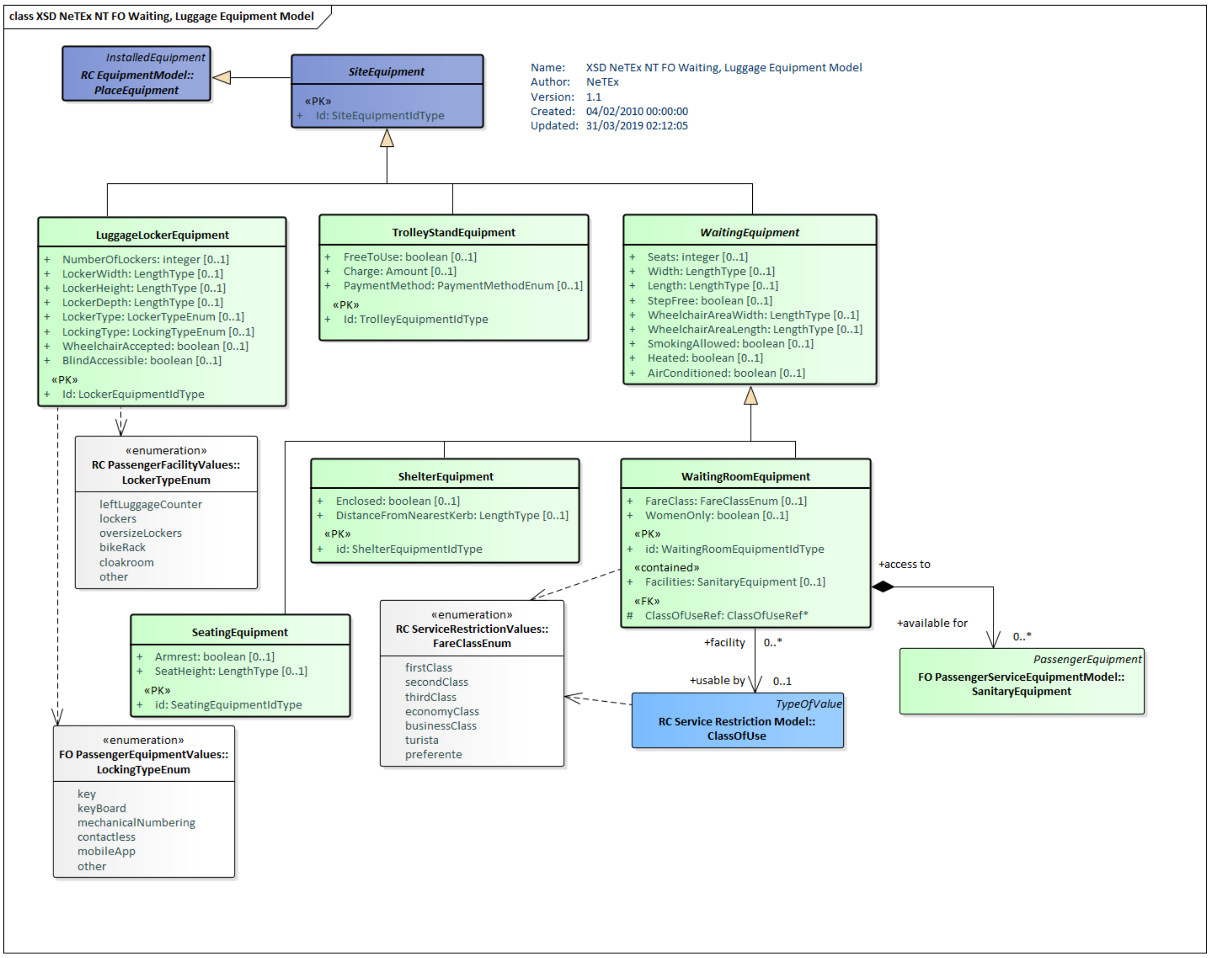


Figure 566 – Waiting Equipment – Physical Model (UML) – Detail

8.4.8.7.3 Waiting Equipment – Attributes and XSD

8.4.8.7.3.1 SiteEquipment – Model Element

Specialisation of PLACE EQUIPMENT for SITEs (e.g. LUGGAGE LOCKER, WAITING EQUIPMENT, TROLLEY STAND, etc.).

8.4.8.7.3.2 LuggageLockerEquipment – Model Element

Specialisation of STOP PLACE EQUIPMENT for LUGGAGE LOCKERs.

Table 466 – *LuggageLockerEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SiteEquipment*	*::>*	LUGGAGE LOCKER EQUIPMENT inherits from SITE EQUIPMENT.
«PK»	***id***	*LockerEquipmentIdType*	1:1	Identifier of LOCKER EQUIPMENT.
	NumberOfLockers	*xsd:integer*	0:1	Number of Lockers.
	LockerWidth	*LengthType*	0:1	Width of Lockers.

	LockerHeight	*LengthType*	0:1	Height of Lockers.
	LockerDepth	*LengthType*	0:1	Depth of Lockers.
«enum»	***LockerType***	*LockerTypeEnum*	0:1	Type of Luggage locker. See allowed values below.
«enum»	***LockingType***	*LockingTypeEnum*	0:1	Locking mechanism of LOCKER EQUIPMENT. See allowed values below. +v1.1
	Wheelchair-Accepted	*xsd:booleanr*	0:1	Whether a wheelchair may be left +v1.1
	BlindAccessible	*xsd:booleanr*	0:1	Whether a visually impaired person may use the mechanismt +v1.1

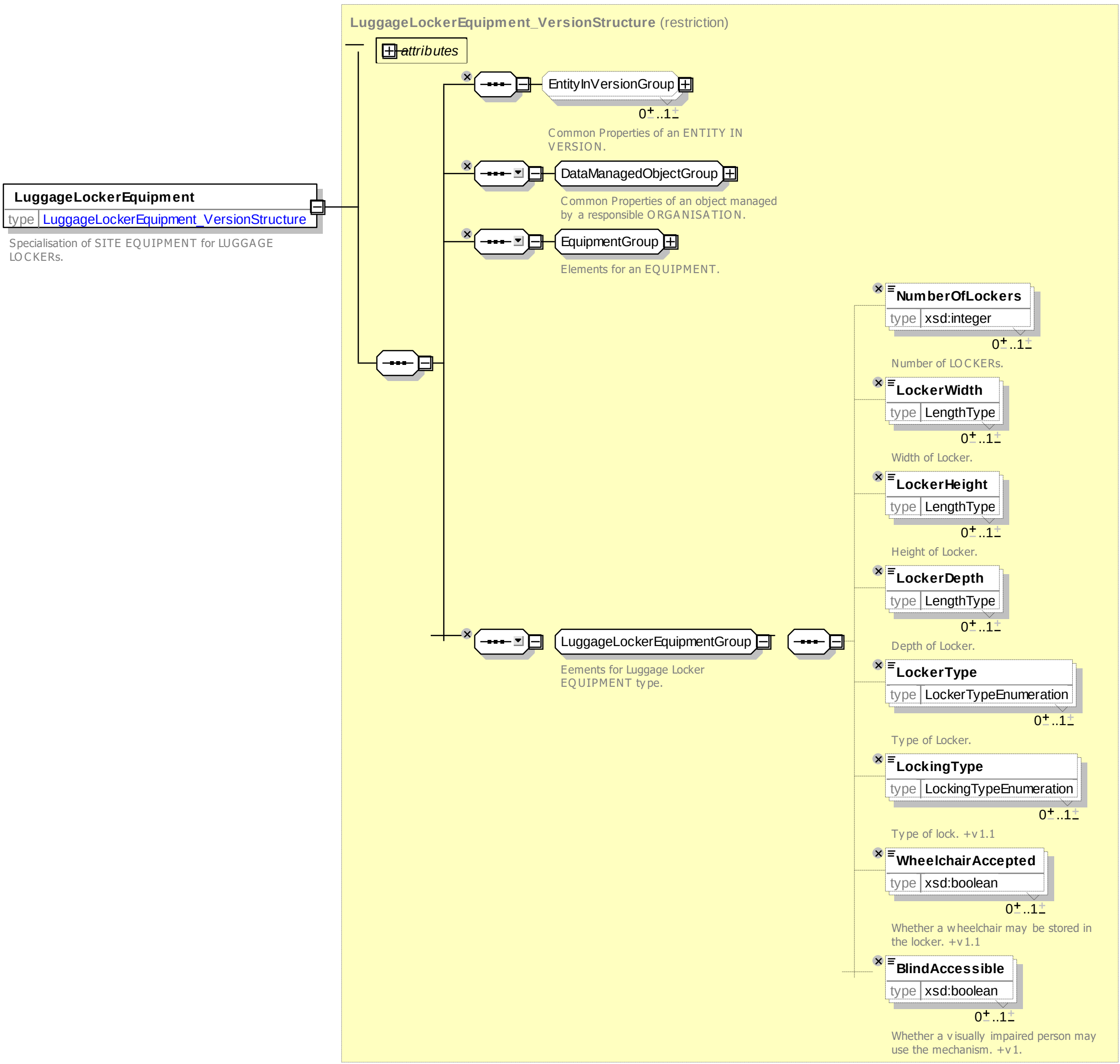


Figure 567 – *LuggageLockerEquipment* – XSD

Table 467 – *LockerType* – Allowed Values

Value	Description
leftLuggageCounter	Left luggage counter
lockers	Lockers
oversizeLockers	Oversize lockers
bikeRack	Bike rack
cloakroom	Cloakroom

8.4.8.7.3.3 TrolleyStandEquipment – Model Element

Specialisation of STOP PLACE EQUIPMENT for TROLLEY STANDs.

Table 468 – *TrolleyStandEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*SiteEquipment*	::>	TROLLEY STAND EQUIPMENT inherits from SITE EQUIPMENT.
«PK»	***id***	*TrolleyEquipmentIdType*	1:1	Identifier of TROLLEY STAND EQUIPMENT.
	FreeToUse	*xsd:boolean*	0:1	Whether EQUIPMENT is free to use.
	Charge	*Amount*	0:1	Charge for using TROLLEY STAND EQUIPMENT.
	Currency	*CurrencyType*	0:1	Currency in which charge is in.
«enum»	***PaymentMethod***	*PaymentMethodEnum*	0:1	Method of paying for use of equipment. See Service Restriction Model for allowed values.

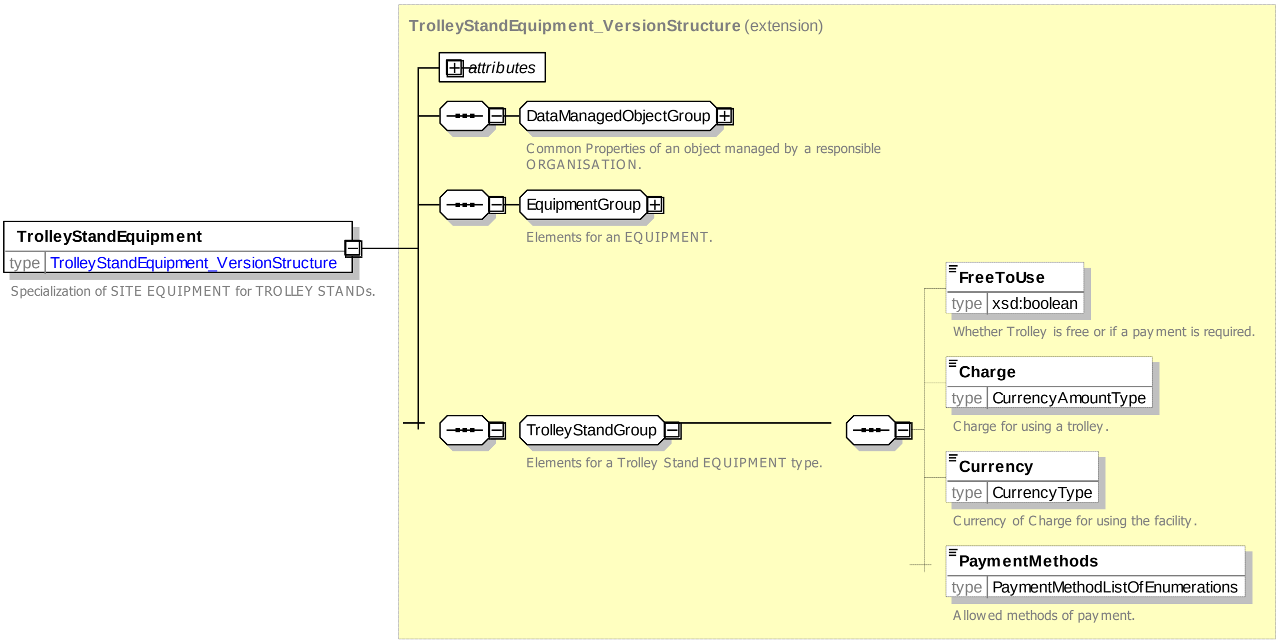


Figure 568 – *TrolleyStandEquipment* – XSD

8.4.8.7.3.4 WaitingEquipment – Model Element

Specialisation of STOP PLACE EQUIPMENT for WAITING EQUIPMENT (shelter, waiting room, etc.).

Table 469 – *WaitingEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SiteEquipment*	*::>*	WAITING EQUIPMENT inherits from SITE EQUIPMENT.
«PK»	***id***	*WaitingEquipment-IdType*	1:1	Identifier of WAITING EQUIPMENT.
	Seats	*xsd:integer*	0:1	Number of seats in WAITING area.
	Width	*LengthType*	0:1	Width of WAITING area.
	Length	*LengthType*	0:1	Length of WAITING area.
	StepFree	*xsd:boolean*	0:1	Whether waiting area is step free.
	WheelchairAreaWidth	*LengthType*	0:1	Width of Wheelchair WAITING area.
	WheelchairArea-Length	*LengthType*	0:1	Length of Wheelchair WAITING area.
	SmokingAllowed	*xsd:boolean*	0:1	Whether smoking is allowed in waiting area.
	Heated	*xsd:boolean*	0:1	Whether shelter has heating.
	AirConditioned	*xsd:boolean*	0:1	Whether shelter has air conditioning.

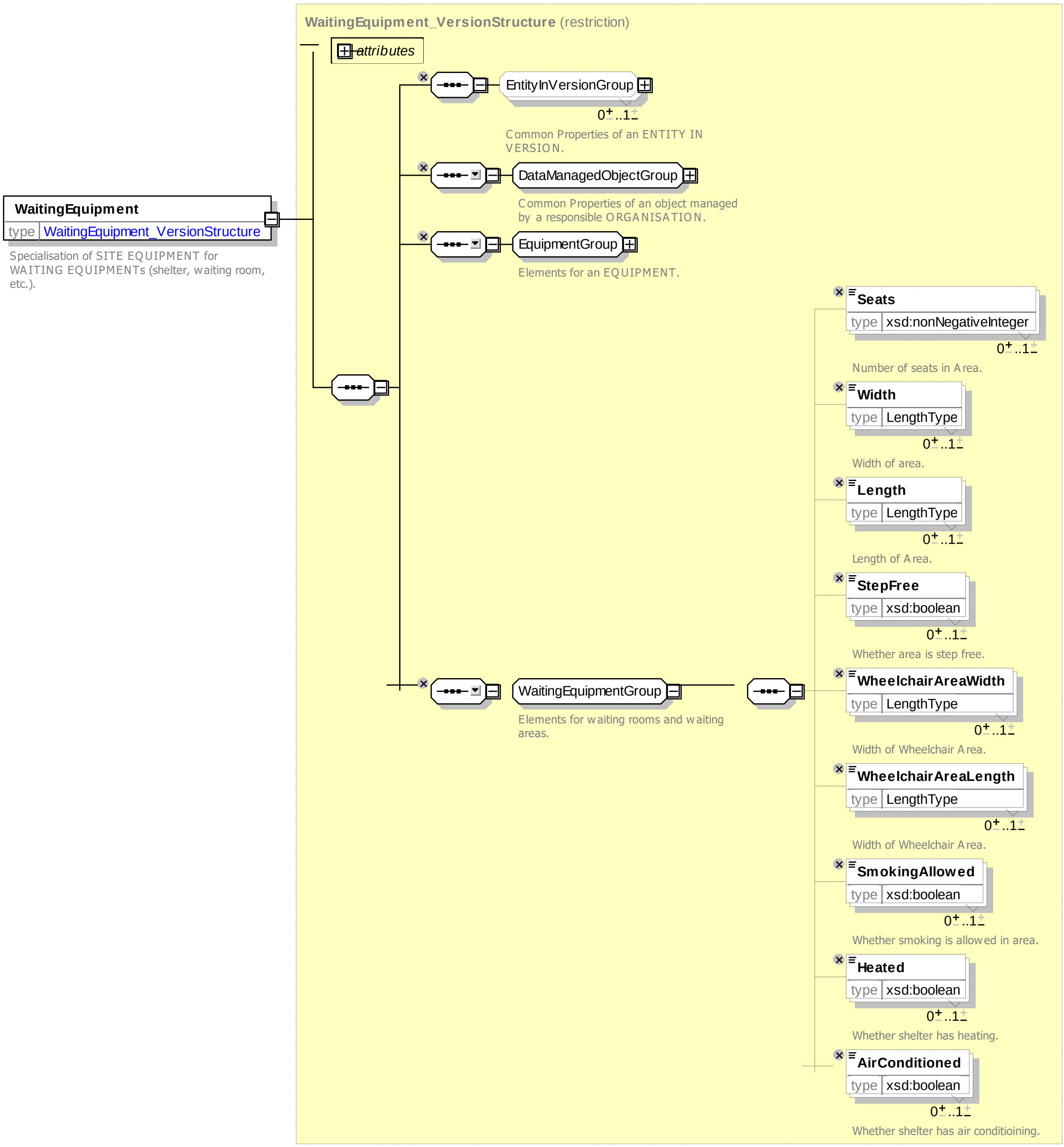


Figure 569 – *WaitingEquipment – XSD*

8.4.8.7.3.5 SeatingEquipment – Model Element

Properties of a seating equipment located on a QUAY or other SPACE.

Table 470 – *SeatingEquipment* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*WaitingEquipment*	::>	SEATING EQUIPMENT inherits from WAITING EQUIPMENT.
«PK»	***id***	*SeatingEquipmentIdType*	1:1	Identifier of SEATING EQUIPMENT.
	Armrest	*xsd:boolean*	0:1	Whether the seating hs an armrest. +v1.1.
	SeatHeight	*LengthType*	0:1	Height of the seating. +v1.1

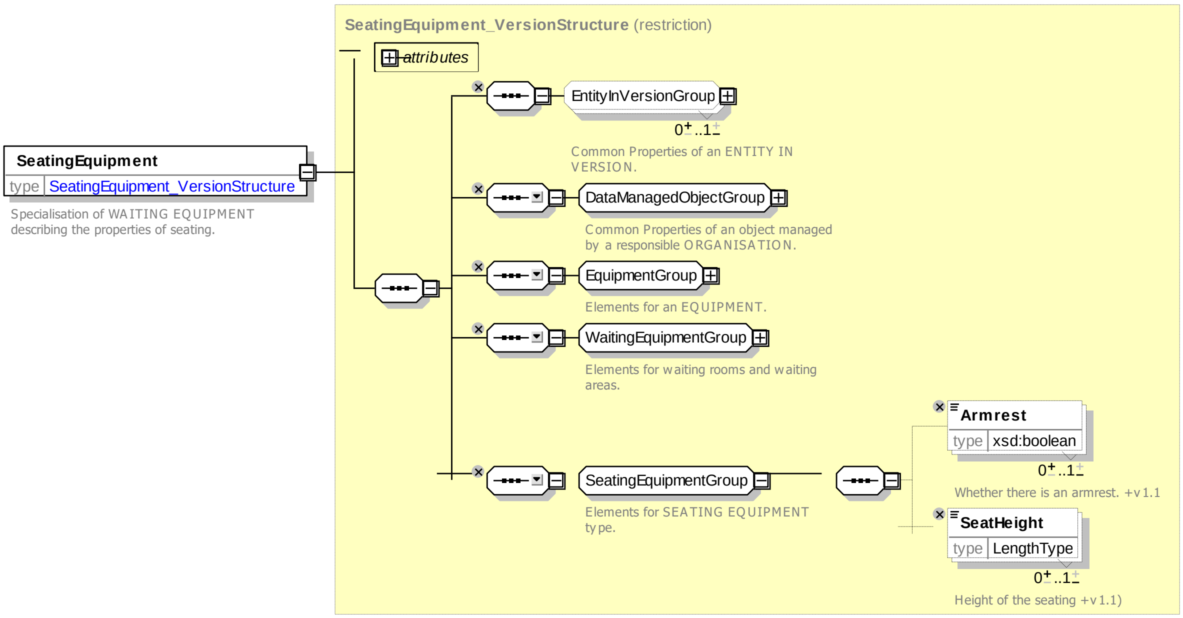


Figure 570 – *SeatingEquipment – XSD*

8.4.8.7.3.6 ShelterEquipment – Model Element

Specialisation of WAITING EQUIPMENT for a SHELTER.

Table 471 – *ShelterEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*WaitingEquipment*	::>	SHELTER EQUIPMENT inherits from WAITING EQUIPMENT.
«PK»	***id***	*ShelterEquipmentIdType*	1:1	Identifier of SHELTER EQUIPMENT.
	Enclosed	*xsd:boolean*	0:1	Whether shelter is enclosed for protection from weather etc.
	DistanceFrom-NearestKerb	*LengthType*	0:1	Distance of shelter from kerb.

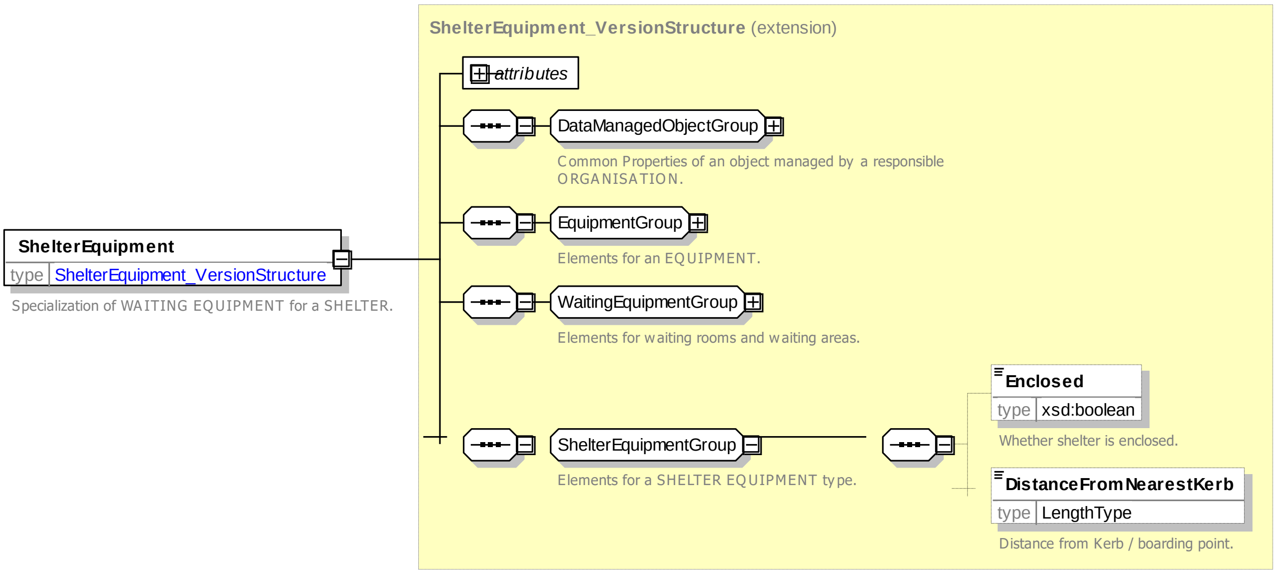


Figure 571 – *ShelterEquipment – XSD*

8.4.8.7.3.7 WaitingRoomEquipment – Model Element

Specialisation of WAITING EQUIPMENT for WAITING ROOM, classified by TYPE OF WAITING ROOM.

Table 472 – *WaitingRoomEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*WaitingEquipment*	::>	WAITING ROOM EQUIPMENT inherits from WAITING EQUIPMENT.
«PK»	***id***	*WaitingRoomEquipmentIdType*	1:1	Identifier of WAITING ROOM EQUIPMENT.
«enum»	***FareClass***	*FareClassEnum*	1:1	FARE CLASS required to use room. See Servcie Restriction Model for allowed values.
	WomenOnly	*xsd:boolean*	1:1	Whether Waiting room is women only.
«enum»	***Sanitary***	*SanitaryEquipmentEnum*	0:1	Sanitary Facilities in WAITING room. See SANITARY EQUIPMENT for allowed values.
«FK»	***ClassOfUseRef***	*ClassOfUseRef*	1:1	CLASS OF USE for facility.

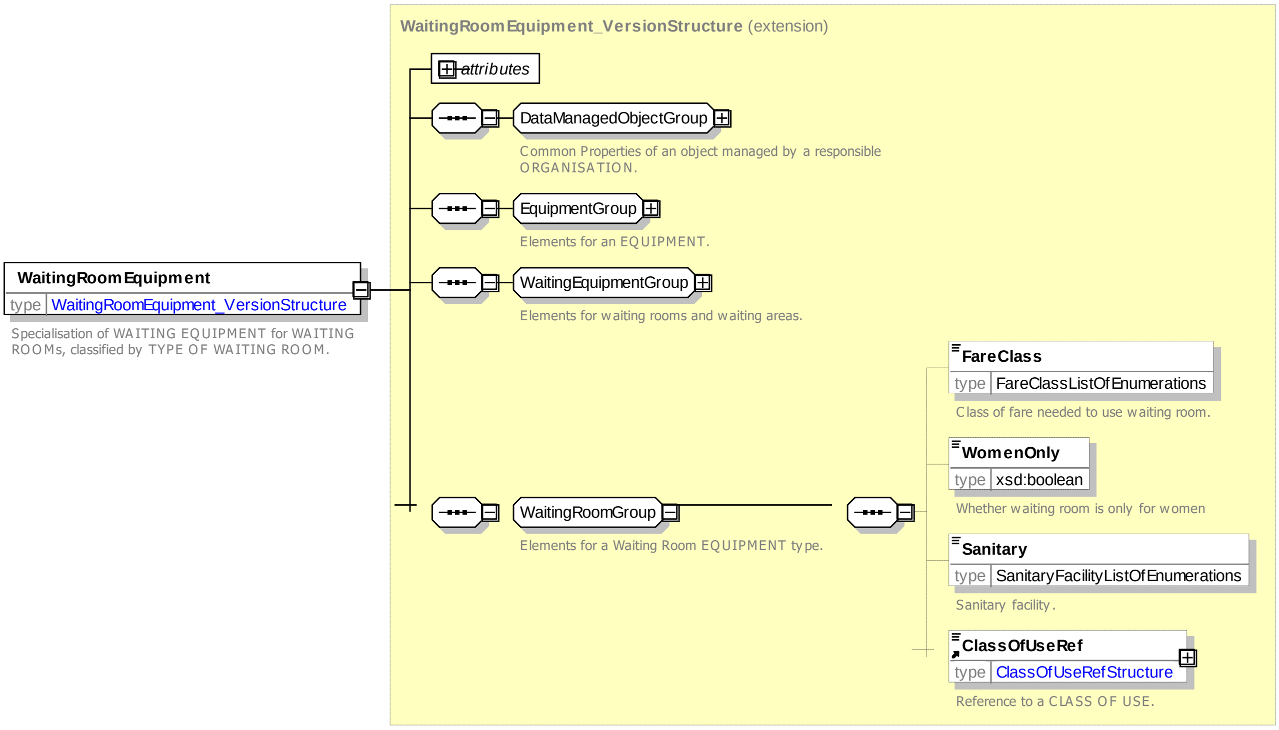


Figure 572 – *WaitingRoomEquipment – XSD*

8.4.8.8 Access Equipment

8.4.8.8.1 ACCESS EQUIPMENT – Conceptual MODEL

ACCESS EQUIPMENT describes equipment and properties relating to access to a SITE by a passenger, such as STAIRs, RAMPs, ESCALATORs, QUEUING EQUIPMENT, SURFACEs, PLACE LIGHTING, etc.

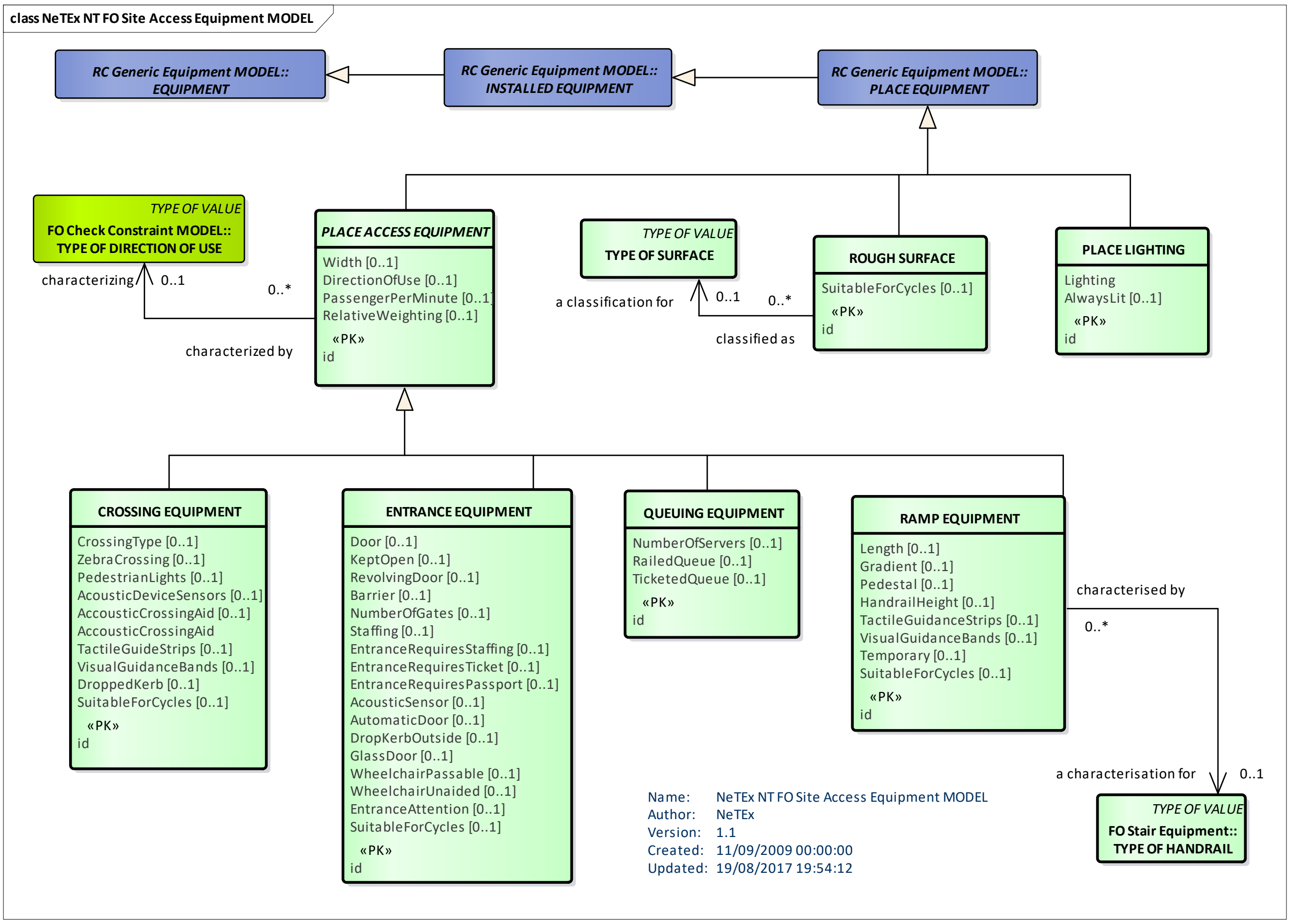


Figure 573 – SITE ACCESS EQUIPMENT – Conceptual MODEL (UML)

8.4.8.8.2 Access Equipment – Physical Model

The following figure shows detailed attributes of the ACCESS EQUIPMENT model elements.

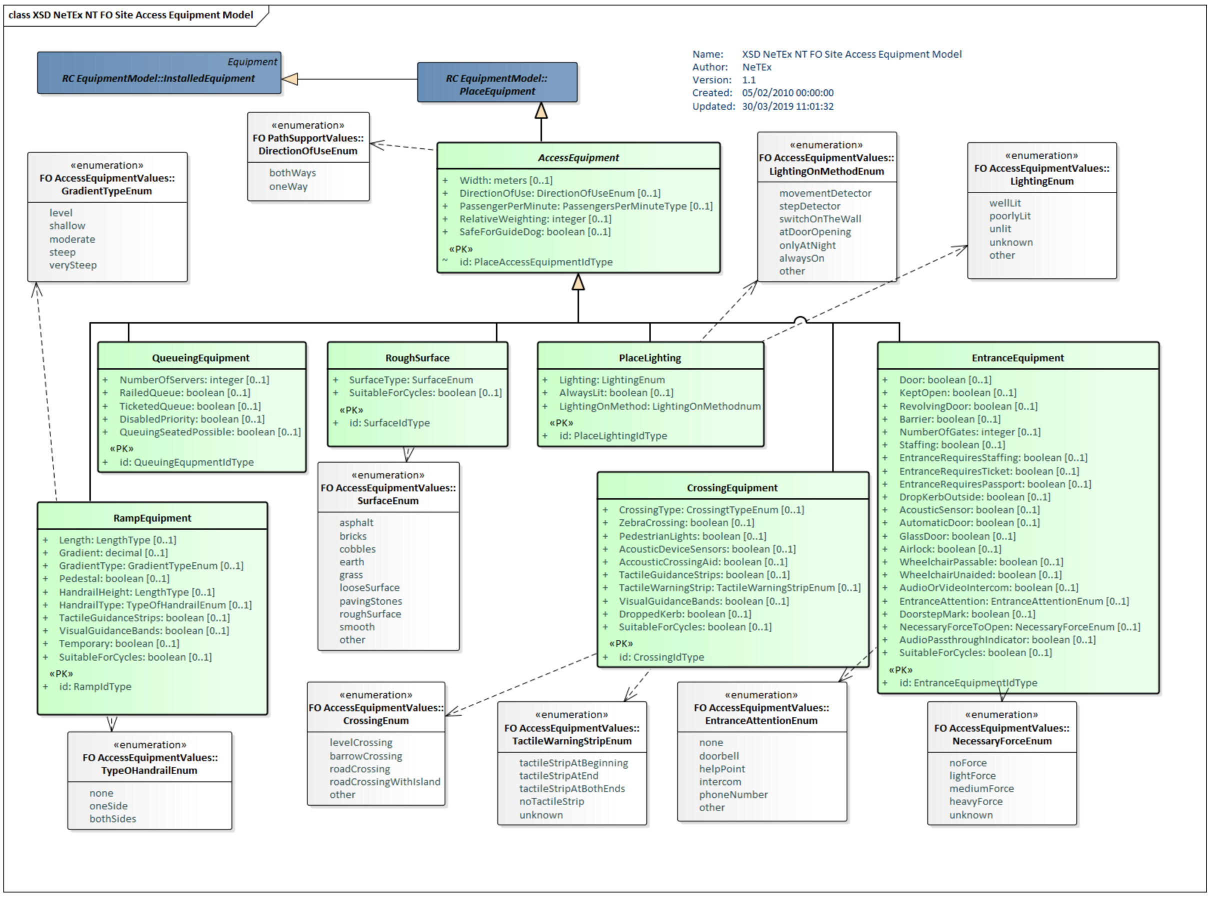


Figure 574 – Access Equipment – Physical Model (UML)

8.4.8.8.3 Access Equipment – Attributes and XSD

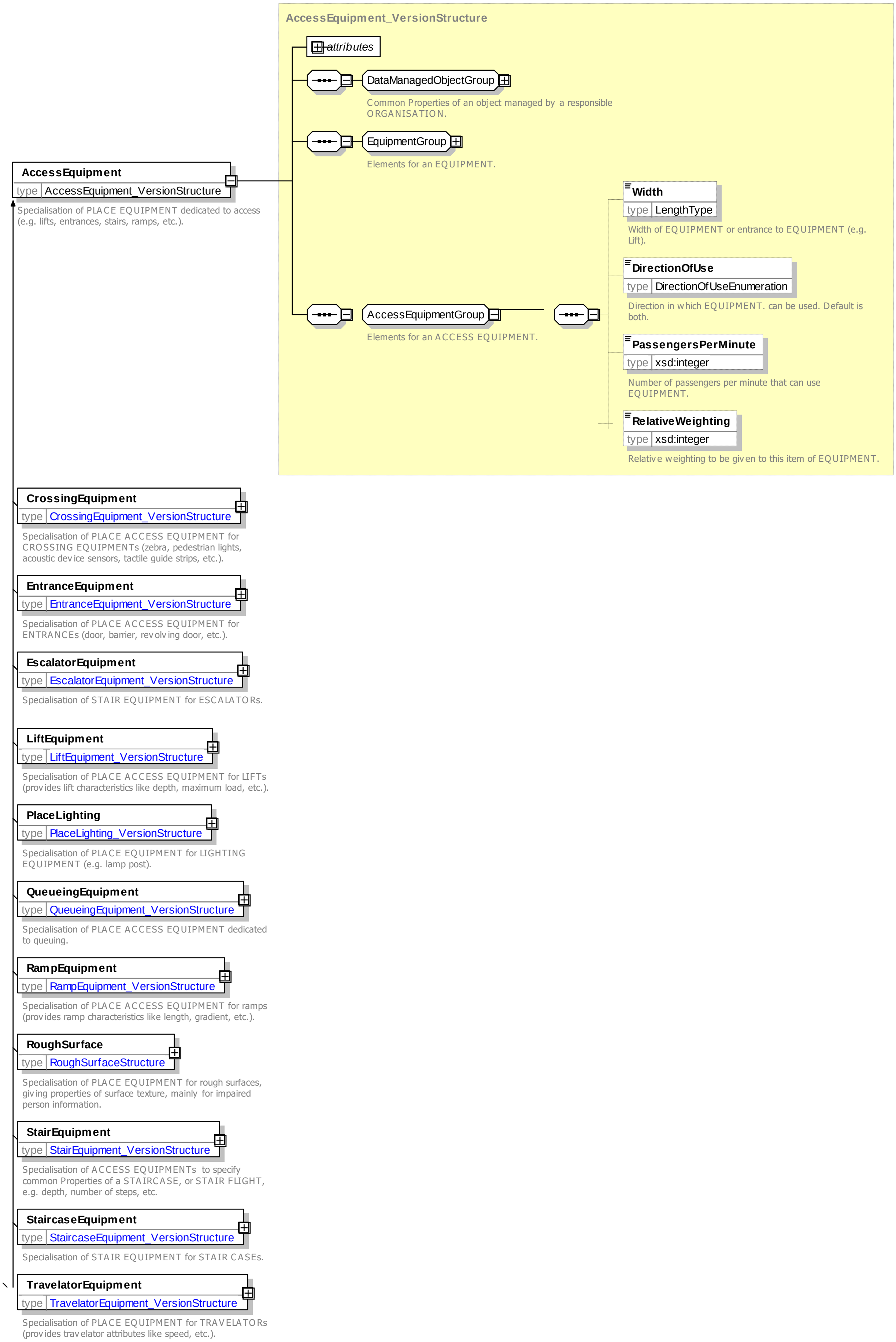
8.4.8.8.3.1 AccessEquipment – Model Element

Specialisation of PLACE EQUIPMENT for PLACE ACCESS EQUIPMENTs (e.g. lifts, entrances, stairs, ramps, etc.).

Table 473 – *AccessEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceEquipment*	*::>*	ACCESS EQUIPMENT inherits from PLACE EQUIPMENT.
«PK»	***id***	*AccessEquipmentIdType*	1:1	Identifier of ACCESS EQUIPMENT.
	Width	*meters*	0:1	Width of EQUIPMENT or entrance to equipment (Lift).
«enum»	***DirectionOfUse***	*DirectionOfUseEnum*	0:1	Direction in which EQUIPMENT can be used. The default is *'both'*.
	PassengerPer-Minute	*PassengersPerMinuteType*	0:1	Number of passengers per minute that can use EQUIPMENT.

	RelativeWeighting	*xsd:integer*	0:1	Relative weighting to be given to this item EQUIPMENT.



AccessEquipment_VersionStructure
attributes
DataManagedObjectGroup
Common Properties of an object managed by a responsible ORGANISATION.
EquipmentGroup
Elements for an EQUIPMENT.
AccessEquipment
type AccessEquipment_VersionStructure
Specialisation of PLACE EQUIPMENT dedicated to access (e.g. lifts, entrances, stairs, ramps, etc.).
Width
type LengthType
Width of EQUIPMENT or entrance to EQUIPMENT (e.g. Lift).
DirectionOfUse
type DirectionOfUseEnumeration
Direction in which EQUIPMENT. can be used. Default is both.
AccessEquipmentGroup
Elements for an ACCESS EQUIPMENT.
PassengersPerMinute
type xsd:integer
Number of passengers per minute that can use EQUIPMENT.
RelativeWeighting
type xsd:integer
Relative weighting to be given to this item of EQUIPMENT.
CrossingEquipment
type CrossingEquipment_VersionStructure
Specialisation of PLACE ACCESS EQUIPMENT for CROSSING EQUIPMENTs (zebra, pedestrian lights, acoustic device sensors, tactile guide strips, etc.).
EntranceEquipment
type EntranceEquipment_VersionStructure
Specialisation of PLACE ACCESS EQUIPMENT for ENTRANCEs (door, barrier, revolving door, etc.).
EscalatorEquipment
type EscalatorEquipment_VersionStructure
Specialisation of STAIR EQUIPMENT for ESCALATORs.
LiftEquipment
type LiftEquipment_VersionStructure
Specialisation of PLACE ACCESS EQUIPMENT for LIFTs (provides lift characteristics like depth, maximum load, etc.).
PlaceLighting
type PlaceLighting_VersionStructure
Specialisation of PLACE EQUIPMENT for LIGHTING EQUIPMENT (e.g. lamp post).
QueueingEquipment
type QueueingEquipment_VersionStructure
Specialisation of PLACE ACCESS EQUIPMENT dedicated to queuing.
RampEquipment
type RampEquipment_VersionStructure
Specialisation of PLACE ACCESS EQUIPMENT for ramps (provides ramp characteristics like length, gradient, etc.).
RoughSurface
type RoughSurfaceStructure
Specialisation of PLACE EQUIPMENT for rough surfaces, giving properties of surface texture, mainly for impaired person information.
StairEquipment
type StairEquipment_VersionStructure
Specialisation of ACCESS EQUIPMENTs to specify common Properties of a STAIRCASE, or STAIR FLIGHT, e.g. depth, number of steps, etc.
StaircaseEquipment
type StaircaseEquipment_VersionStructure
Specialisation of STAIR EQUIPMENT for STAIRCASEs.
TravelatorEquipment
type TravelatorEquipment_VersionStructure
Specialisation of PLACE EQUIPMENT for TRAVELATORs (provides travelator attributes like speed, etc.).

Figure 575 – *AccessEquipment – XSD*

8.4.8.8.3.2 CrossingEquipment – Model Element

Specialisation of PLACE ACCESS EQUIPMENT for CROSSING EQUIPMENTs (zebra, pedestrian lights, acoustic device sensors, tactile guide strips, etc.).

Table 474 – *CrossingEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceAccessEquipment*	*::>*	CROSSING inherits from PLACE ACCESS EQUIPMENT.
«PK»	***id***	*CrossingIdType*	1:1	Identifier of CROSSING.
«enum»	***CrossingType***	*CrossingtTypeEnum*	0:1	Type of CROSSING. See allowed values below.
	ZebraCrossing	*xsd:boolean*	0:1	Whether CROSSING is marked as Zebra.
	PedestrianLights	*xsd:boolean*	0:1	Whether there are lights for pedestrians to cross.
	AcousticDevice-Sensors	*xsd:boolean*	0:1	Whether CROSSING has Acoustic Device sensors.
	AccousticCrossing Aid	*xsd:boolean*	0:1	Whether CROSSING has Accustic Crossing aids.
	TactileGuidance-Strips	*xsd:boolean*	0:1	Whether CROSSING has tactile guidance strips.
«enum»	***TactileWarningStrip***	*TactileWarningstripEnum*	0:1	Nature of tactile warning strip. See allowed values below. +v1.1
	VisualGuidance-Bands	*xsd:boolean*	0:1	Whether CROSSING has visual guidance strips.
	DroppedKerb	*xsd:boolean*	0:1	Whether CROSSING has dropped Kerb (both sides).
	SuitableForCycles	*xsd:boolean*	0:1	Whether CROSSING is suitable for a cycle.

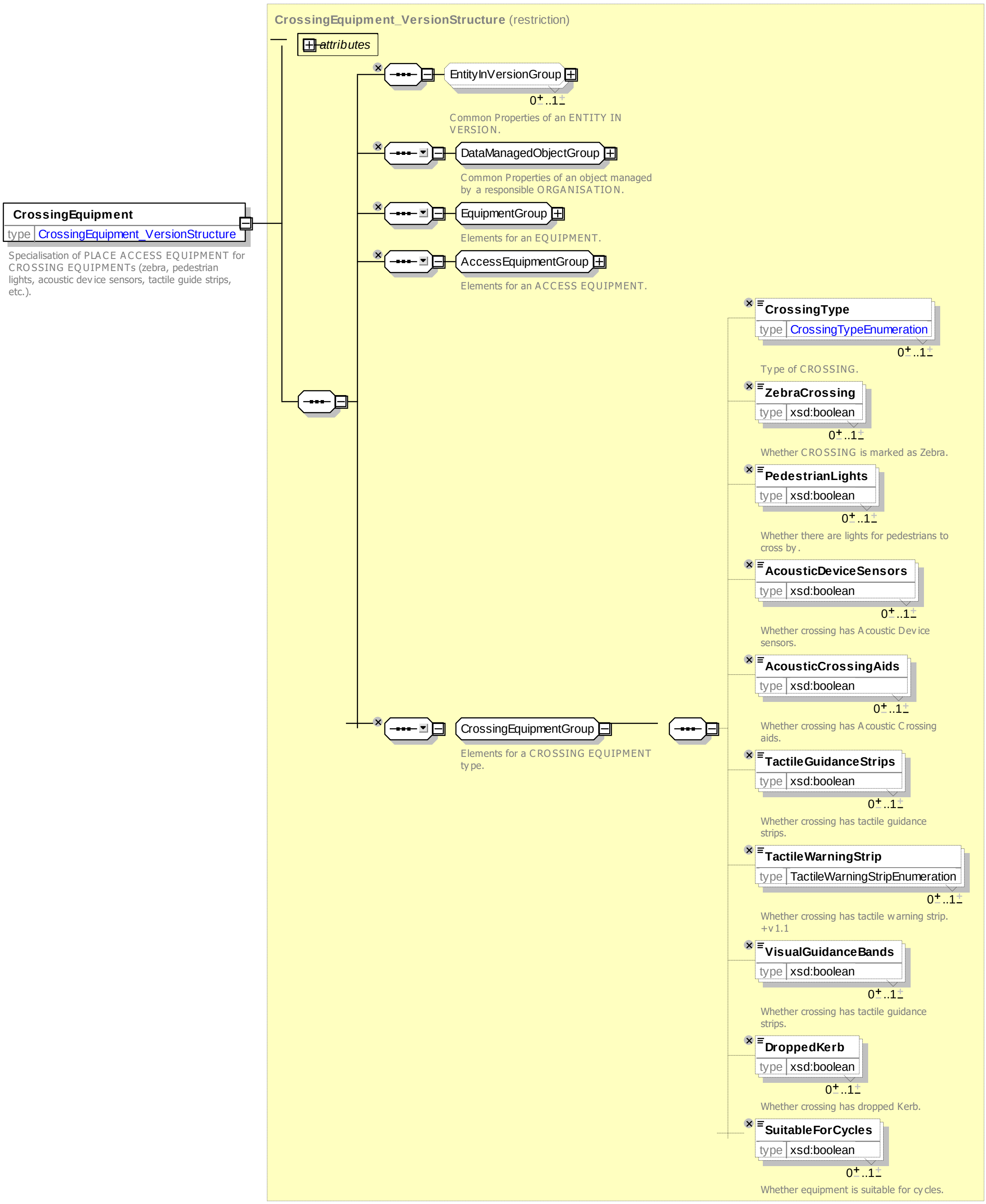


Figure 576 – *CrossingEquipment – XSD*

CrossingType – Allowed values

Allowed values for CROSSING type (*CrossingEnumeration*).

Table 475 – *CrossingType* – Allowed Values

Value	Description
levelCrossing	Level Crossing

barrowCrossing	Barrow Crossing across tracks
roadCrossing	Road crossing
roadCrossingWithIsland	Road crossing with island
other	Other crossing type

TactileWarningStrip – Allowed values

Allowed values for ***TactileWarningStrip*** (*TactileWarningStripEnumeration)*.

Table 476 – *TactileWarningStrip* – Allowed Values

Value	Description
tactileStripAtBeginning	Tactile strip at beginning
tactileStripAtEnd	Tactile strip at end
tactileStripAtBothEnds	Tactile strip at both ends
noTactileStrip	No tactile strip
unknown	Unknown

8.4.8.8.3.3 EntranceEquipment – Model Element

Specialisation of PLACE ACCESS EQUIPMENT for ENTRANCEs (door, barrier, revolving door, etc.).

Table 477 – *EntranceEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	<u>*PlaceAccess-Equipment*</u>	*::>*	ENTRANCE EQUIPMENT inherits from PLACE ACCESS EQUIPMENT.
«PK»	***id***	*EntranceEquipment-IdType*	1:1	Identifier of ENTRANCE EQUIPMENT.
XGRP	***EntranceProperties-Group***	<u>***xmlGroup***</u>	0:1	General properties of an ENTRANCE.
	EntranceRequires-Staffing	*xsd:boolean*	0:1	Whether passage requires that barrier be staffed.
	EntranceRequiresTicket	*xsd:boolean*	0:1	Whether passage requires ticket.
	EntranceRequires-Passport	*xsd:boolean*	0:1	Whether passage requires passport.
XGRP	***EntranceAccessibility-Group***	<u>***xmlGroup***</u>	0:1	Accessibility related properties of ENTRANCE.

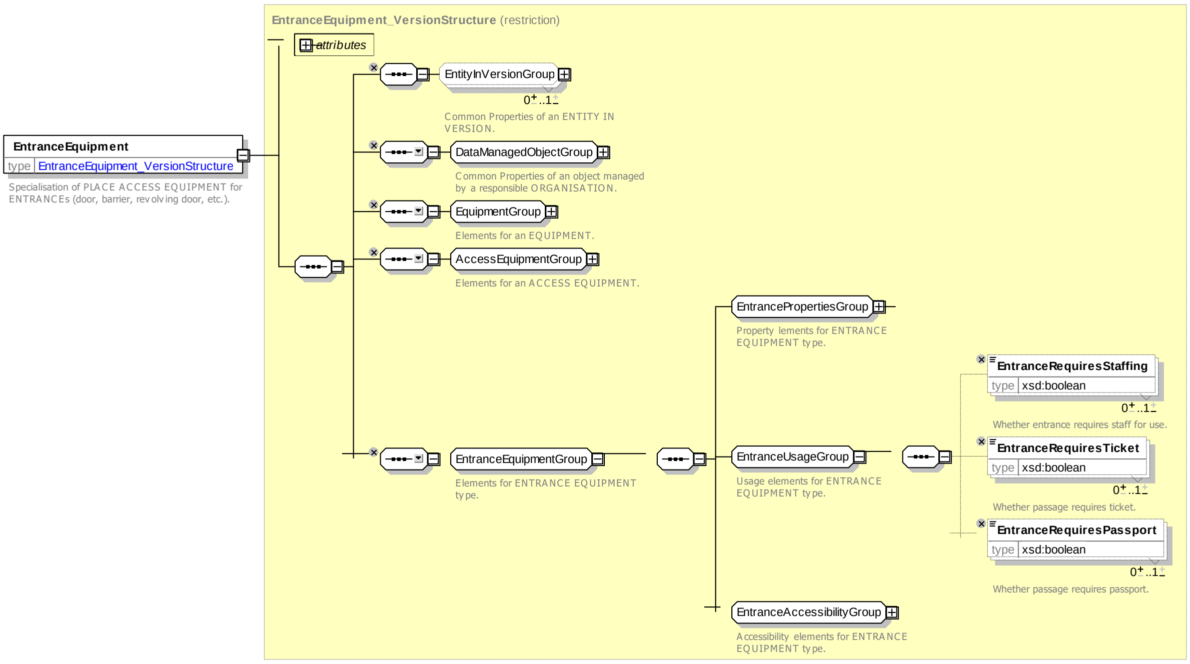


Figure 577 – *EntranceEquipment – XSD*

8.4.8.8.3.4 EntrancePropertiesGroup – XML Group

The following figure shows properties characterising ENTRANCE EQUIPMENT.

Table 478 – *EntrancePropertiesGroup* – Group

Classification	Name	Type	Cardinality	Description
	Door	*xsd:boolean*	0:1	Whether there is a door in the entrance. If false opening does not have door.
	KeptOpen	*xsd:boolean*	0:1	Whether the door is kept open.
	RevolvingDoor	*xsd:boolean*	0:1	Whether door is revolving. Only applies if door is specified.
	Barrier	*xsd:boolean*	0:1	Whether there is a physical barrier across the doorway.
	NumberOfGates	*xsd:integer*	0:1	Number of gates in barrier or entrance.
	Staffing	*xsd:boolean*	0:1	Staffing of entrance or barrier.

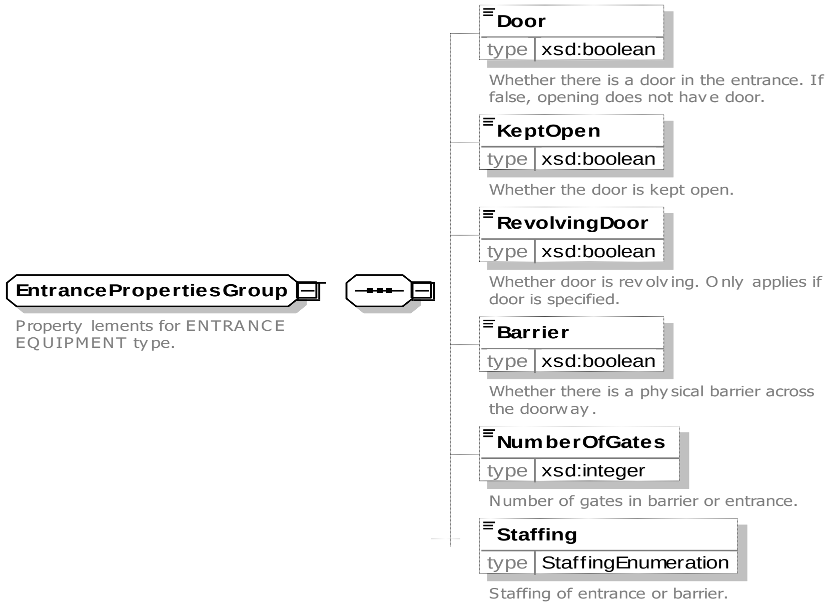


Figure 578 – *EntrancePropertiesGroup* – *XSD*

8.4.8.8.3.5 EntranceAccessibilityGroup – XML Group

The following figure shows accessibility related elements of ENTRANCE EQUIPMENT.

Table 479 – *EntranceAccessibilityGroup* – Group

Classification	Name	Type	Cardinality	Description
	DropKerbOutside	*xsd:boolean*	0:1	Whether there is a drop Kerb outside ENTRANCE.
	AcousticSensor	*xsd:boolean*	0:1	Whether door has acoustic sensors.
	AutomaticDoor	*xsd:boolean*	0:1	Whether doors are automatic.
	GlassDoor	*xsd:boolean*	0:1	Whether door is made of glass.
	Airlock	*xsd:boolean*	0:1	Whether door has an airlock +v1.1 glass.
	WheelchairPassable	*xsd:boolean*	0:1	Door can be passed in a wheel chair.
	WheelchairUnaided	*xsd:boolean*	0:1	Can be passed in a wheel chair unaided.
«enum»	***EntranceAttention***	*EntranceAttention-Enum*	0:1	Nature of doorbell, help point, etc. See allowed values below. See allowed values below.
	DoorstepMark	*xsd:boolean*	0:1	Whether there is a tactile doorstep mark. +v1.1
«enum»	***NecessaryForceTo-Open***	*NecessaryForceTo-OpenEnum*	0:1	Necessary force to open the door. See allowed values below. +v1.1.
	AudioPassthrough-Indicator	*xsd:boolean*	0:1	Whether there is a an audio signal indicating passing through the door. +v1.1
	SuitableForCycles	*xsd:boolean*	0:1	Whether ENTRANCE is suitable for a cycle.

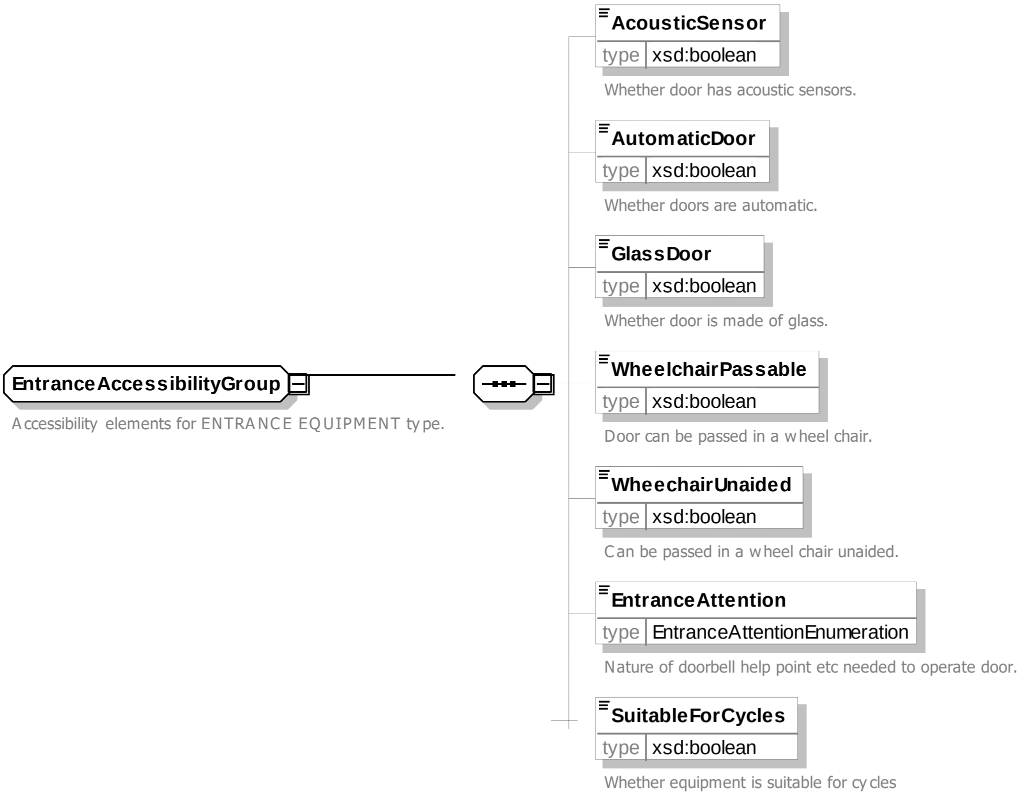


Figure 579 – *EntranceAccessibilityGroup – XSD*

EntranceAttention – Allowed values

Allowed values for Entrance Attention (*EntranceAttentionEnumeration*).

Table 480 – *EntranceAttention* – Allowed Values

Value	Description
none	No attention device
doorbell	Doorbell
helpPoint	Help point
intercom	Intercom
phoneNumber	Phone number to call
other	Other device

8.4.8.8.3.6 QueueingEquipment – Model Element

Specialisation of PLACE ACCESS EQUIPMENT for QUEUING.

Table 481 – *QueueingEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceAccessEquipment*	*::>*	QUEUING EQUIPMENT inherits from PLACE ACCESS EQUIPMENT.

«PK»	***id***	*QueuingEqupmentIdType*	1:1	Identifier of QUEUING EQUIPMENT. See allowed values below.
	NumberOfServers	*xsd:integer*	0:1	Number of tills or servers serving QUEUE.
	RailedQueue	*xsd:boolean*	01:1	Whether QUEUE is bounded by rails.
	TicketedQueue	*xsd:boolean*	0:1	Whether QUEUE is controlled by issuing priority tickets.
	DisabledPriority	*xsd:boolean*	0:1	Whether there is priority acces for the disabled. +v1.1.
	QueuingSeated-Possible	*xsd:boolean*	0:1	Whether queuing may be done seated. + v1.1

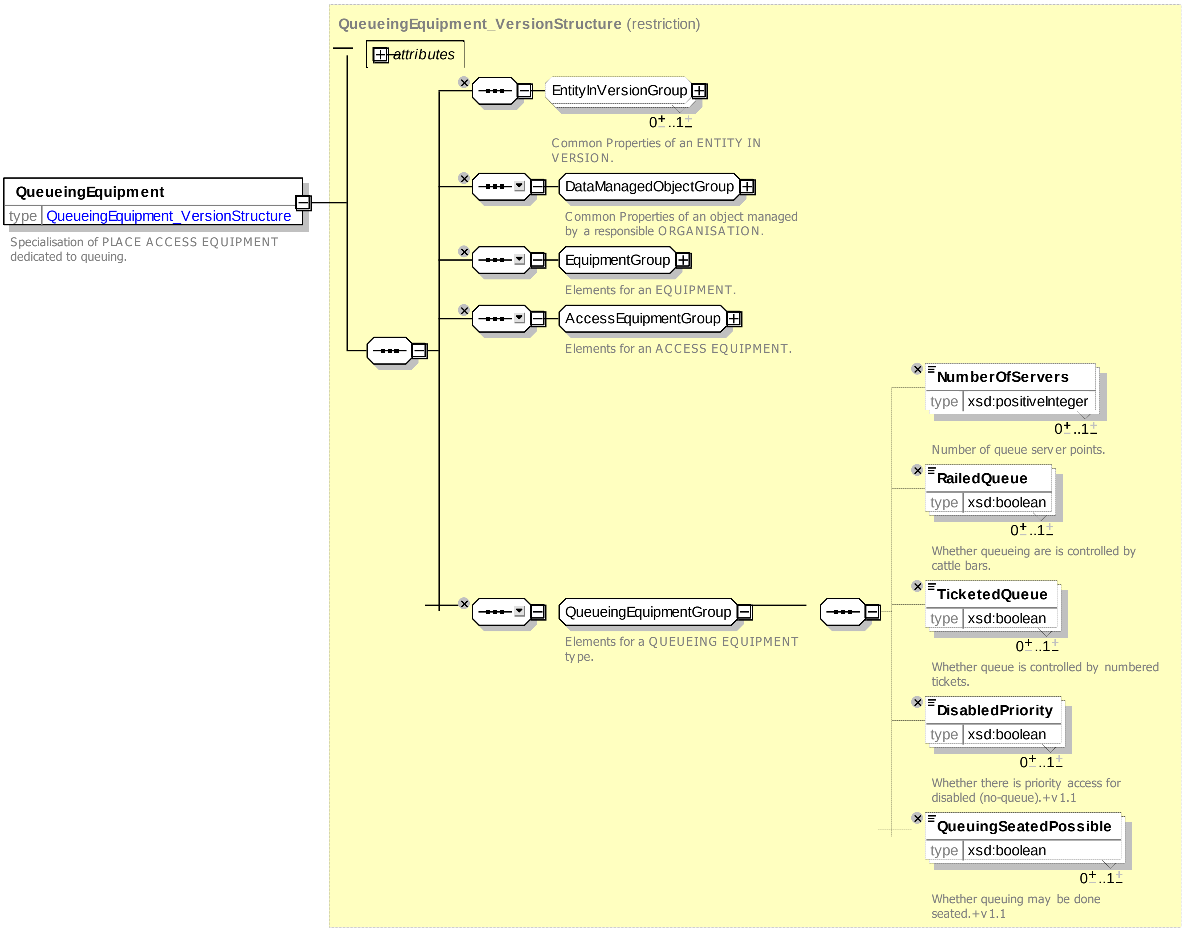


Figure 580 – *QueueingEquipment – XSD*

8.4.8.8.3.7 RampEquipment – Model Element

Specialisation of PLACE ACCESS EQUIPMENT for RAMPs (provides ramp attributes like length, gradient, etc.).

Table 482 – *RampEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceAccessEquipment*	*::>*	RAMP inherits from PLACE ACCESS EQUIPMENT
«PK»	***id***	*RampIdType*	1:1	Identifier of RAMP.
	Length	*LengthType*	0:1	Length of RAMP.
	Gradient	*xsd:positiveInteger*	0:1	Gradient of RAMP.
«enum»	***GradientType***	*RampGradientEnum*	0:1	Gradient of RAMP fixed values. See allowed values below.
	Pedestal	*xsd:boolean*	0:1	Whether RAMP has pedestal.
«enum»	***TypeOfHandrail***	*TypeOfHandrailEnum*	0:1	Type of Handrail on RAMP. See allowed values below.
	HandrailHeight	*LengthType*	0:1	Height of Handrail on RAMP.
	TactileGuidance-Strips	*xsd:boolean*	0:1	Whether RAMP has tactile guidance strips.
	VisualGuidance-Bands	*xsd:boolean*	0:1	Whether RAMP has visual guidance strips.
	Temporary	*xsd:boolean*	0:1	Whether RAMP is temporary.

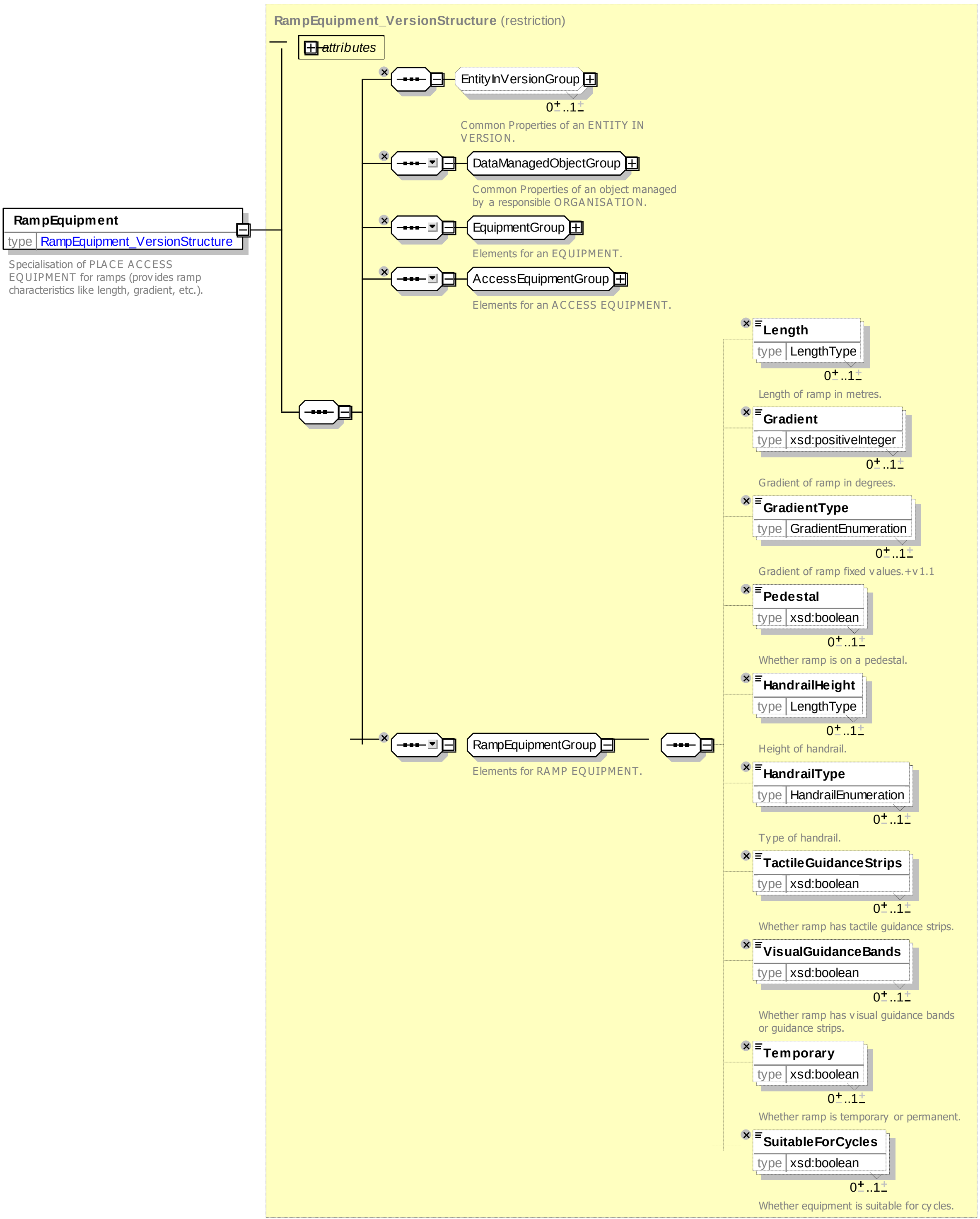


Figure 581 – *RampEquipment – XSD*

Gradient– Allowed Values

Allowed values for RAMP gradient (*RampGradientEnumeration*).

Table 483 – *Gradient* – Allowed Values

Value	Description
verySteep	Very steep gradient
steep	Steep gradient

medium	Moderate gradient
gentle	Shallow gradient
level	No gradient

8.4.8.8.3.8 PlaceLighting – Model Element

Specialisation of PLACE EQUIPMENT for LIGHTING EQUIPMENT (e.g. lamp post).

Table 484 – *PlaceLighting* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PlaceEquipment*	::>	PLACE LIGHTING inherits from PLACE EQUIPMENT.
«PK»	***id***	*PlaceLightingIdType*	1:1	Identifier of PLACE LIGHTING.
«enum»	***Lighting***	*LightingEnum*	0:1	Nature of Lighting. See allowed values below.
	AlwaysLit	*xsd:boolean*	0:1	Whether Place is always lit.
«enum»	***LightingOn-Method***	*LightingOnMethodEnum*		Method by which lighting is swicthed on. See allowed values below.

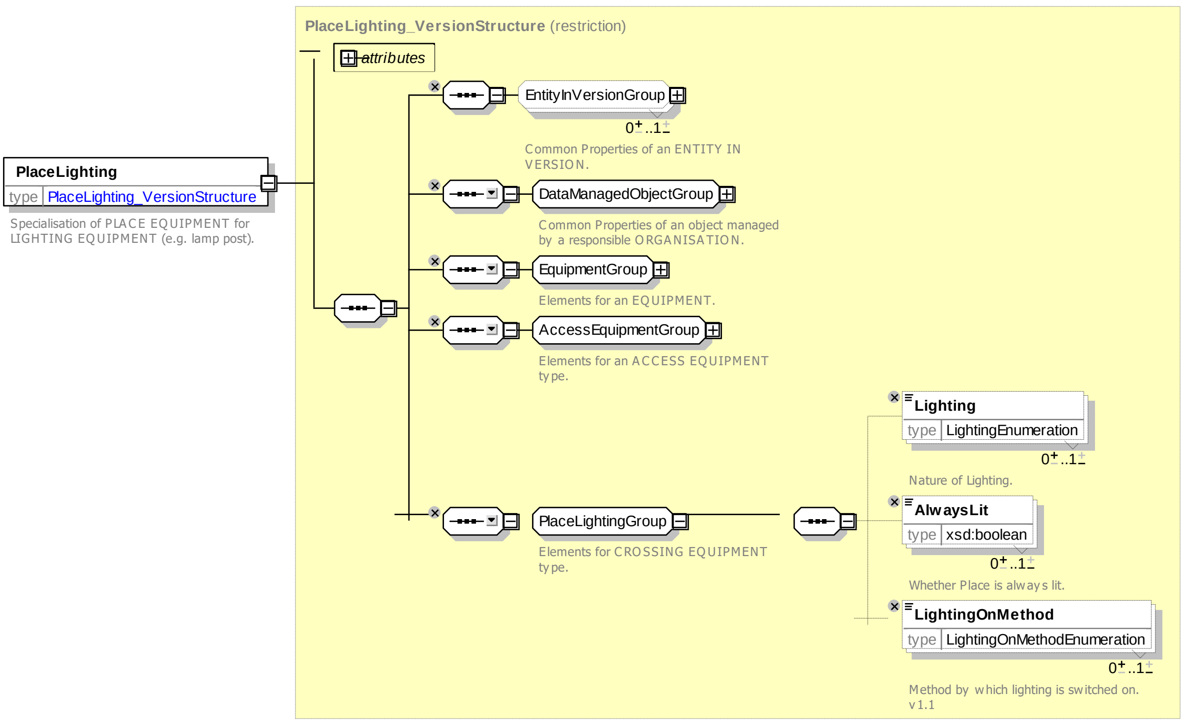


Figure 582 – *PlaceLighting – XSD*

8.4.8.8.3.9 Lighting – Allowed values

Allowed values for ***Lighting*** type *(LightingEnumeration).*

Table 485 – *Lighting* – Allowed Values

Value	Description
wellLit	Well lit
poorlyLit	Poorly lit
unlit	Unlit
unknown	Lighting not known
other	Other lighting type

LightingOnMethod– Allowed Values

Allowed values for ***LightingOnMethod*** (*LightingOnMethodEnuerationm*).

Table 486 – *LightingOnMethod* – Allowed Values

Value	Description
movementDetector	Movement detector turns on lighting automatically.
stepDetector	Steep gradient
switchOnTheWall	Moderate gradient
atDoorOpening	Shallow gradient
onlyAtNight	Lights are on, but only at night
alwaysOn	Lights are always on
other	Other

8.4.8.8.3.10 *RoughSurface*– Model Element

Specialisation of PLACE EQUIPMENT for ROUGH SURFACEs. Properties of surface texture, mainly for impaired person information.

Table 487 – *RoughSurface* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceEquipment*	*::>*	ROUGH SURFACE inherits from PLACE EQUIPMENT.
«PK»	***id***	*SurfaceIdType*	1:1	Identifier of ROUGH SURFACE.
«enum»	***SurfaceType***	*SurfaceEnum*	1:1	Type of surface. See allowed values below.
	SuitableForCycles	*xsd:boolean*	0:1	Whether ROUGH SURFACE is suitable for a cycle.

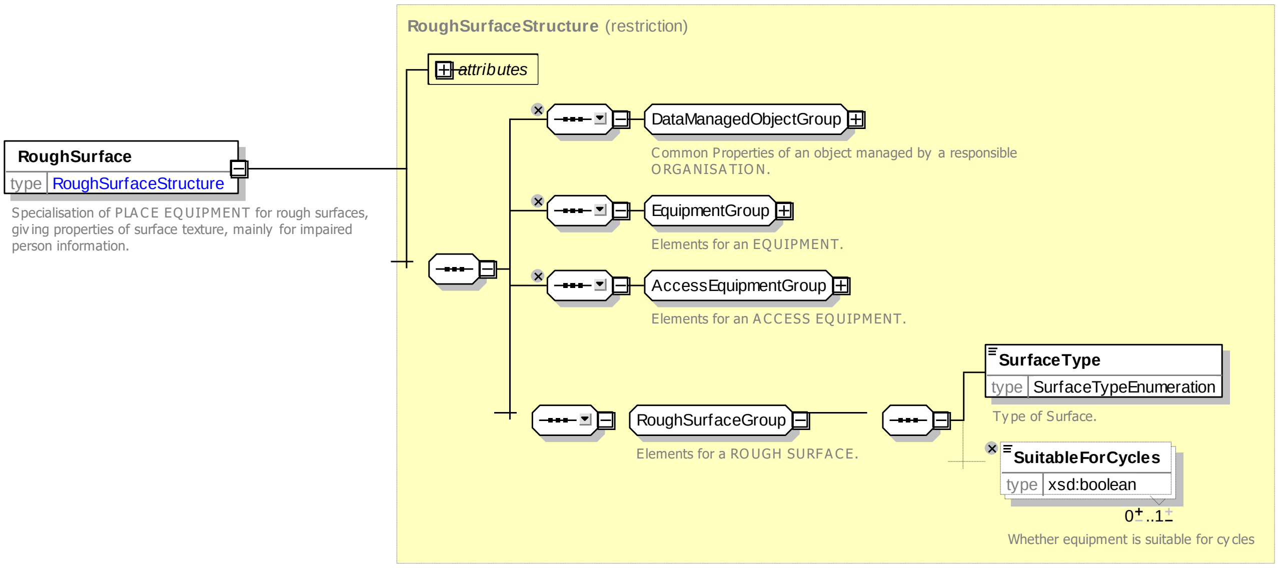


Figure 583 – *RoughSurface – XSD*

8.4.8.8.3.11 SurfaceType – Allowed values

Allowed values for ***SurfaceType*** (*SurfaceEnumeration*).

Table 488 – *SurfaceType* – Allowed Values

Value	Description
asphalt	Surface is asphalt.
bricks	Surface is bricks.
cobbles	Surface is cobbles.
earth	Surface is bare earth.
grass	Surface is grass.
looseSurface	Loose surface.
pavingStones	Paving stones.
roughSurface	Rough surface.
smooth	Smooth surface.
other	Other.

8.4.8.8.4 STAIR EQUIPMENT Equipment – Conceptual MODEL

The Stair CONCEPTUAL model describes stairs and escalators.

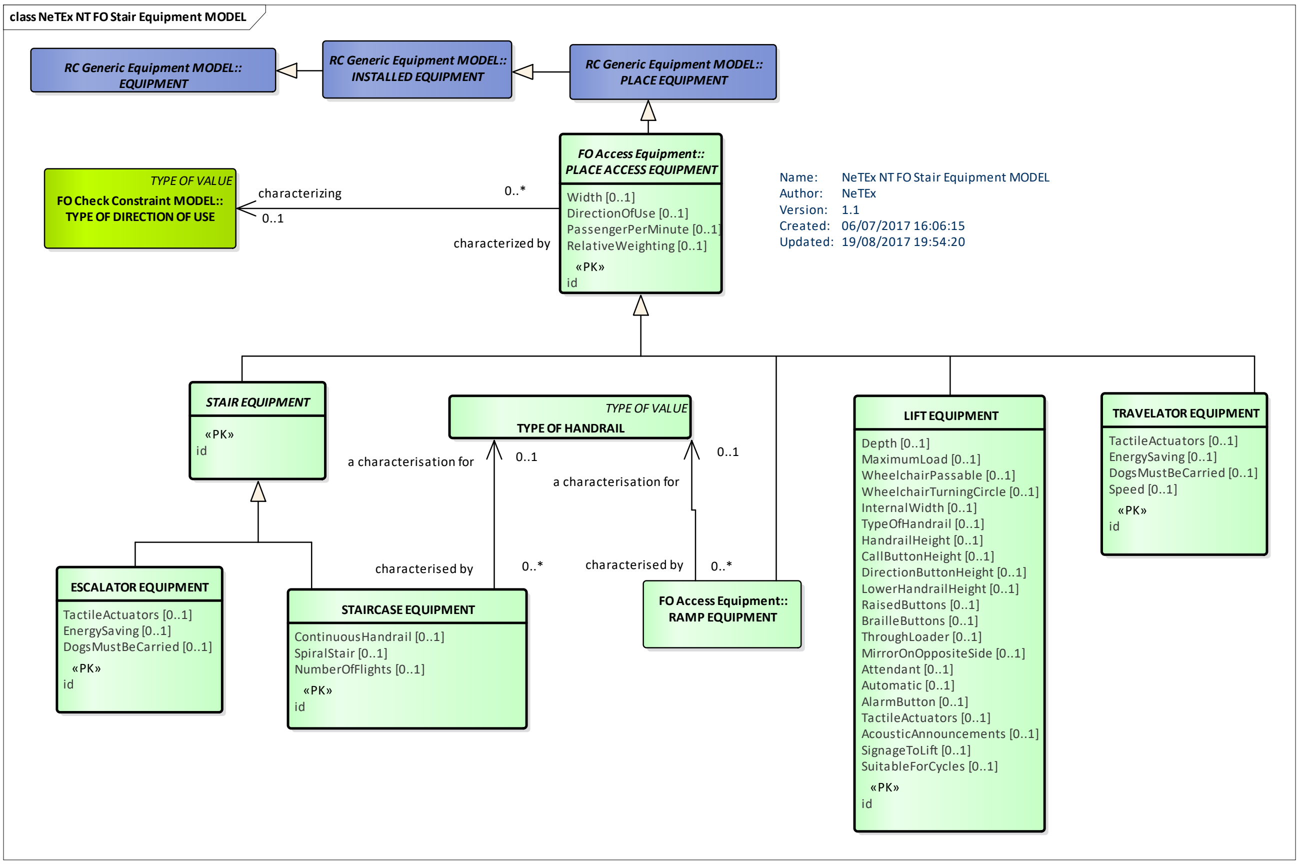


Figure 584 – Stair Equipment – Conceptual MODEL (UML)

8.4.8.8.5 Stair Equipment – Physical Model

The following figure shows detailed attributes of the STAIR EQUIPMENT model elements

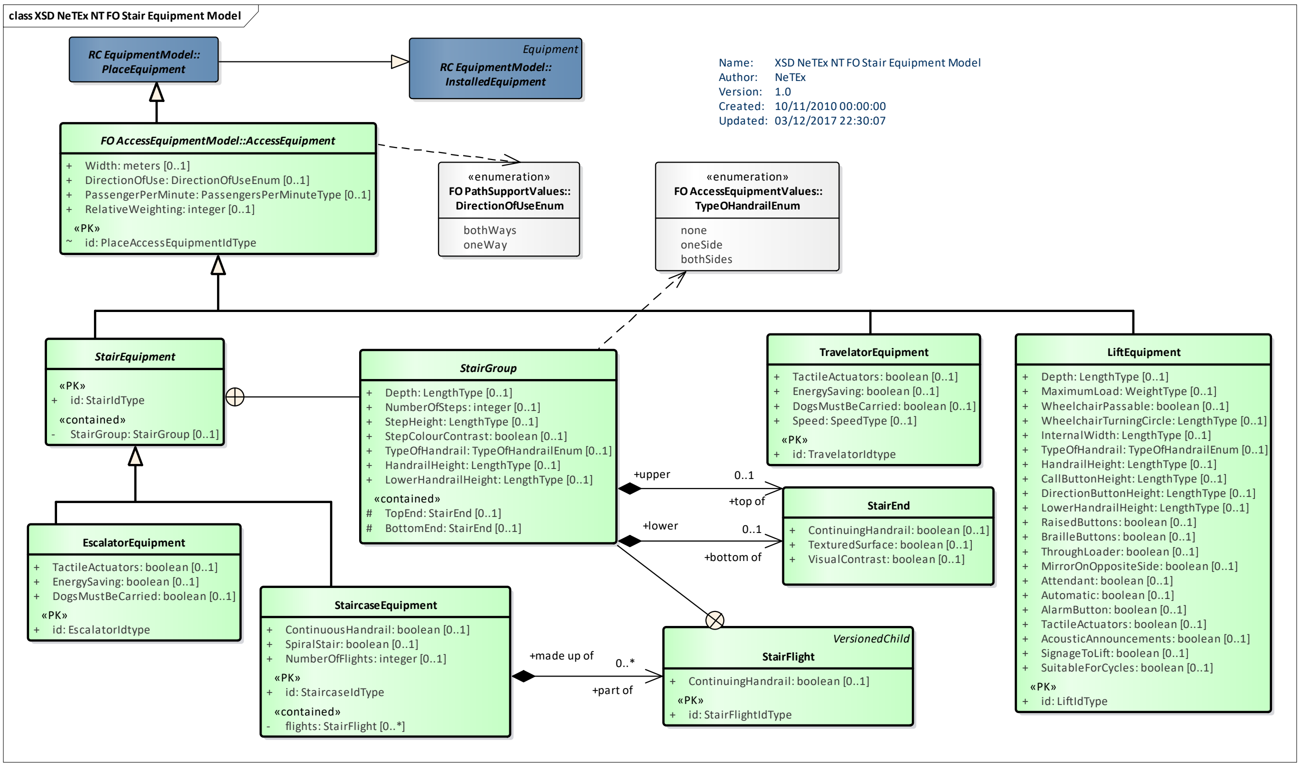


Figure 585 – Stair Equipment – Physical Model (UML)

8.4.8.8.6 Stair Equipment – Attributes and XSD

8.4.8.8.6.1 StairEquipment – Model Element

Properties of a stair.

Table 489 – *StairEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*PlaceAccessEquipment*</u>	::>	STAIR inherits from PLACE ACCESS EQUIPMENT.
«PK»	***id***	*StairIdType*	1:1	Identifier of STAIR.
XGRP	***StairGroup***	<u>***xmlGroup***</u>	0:1	Common Stair Properties.

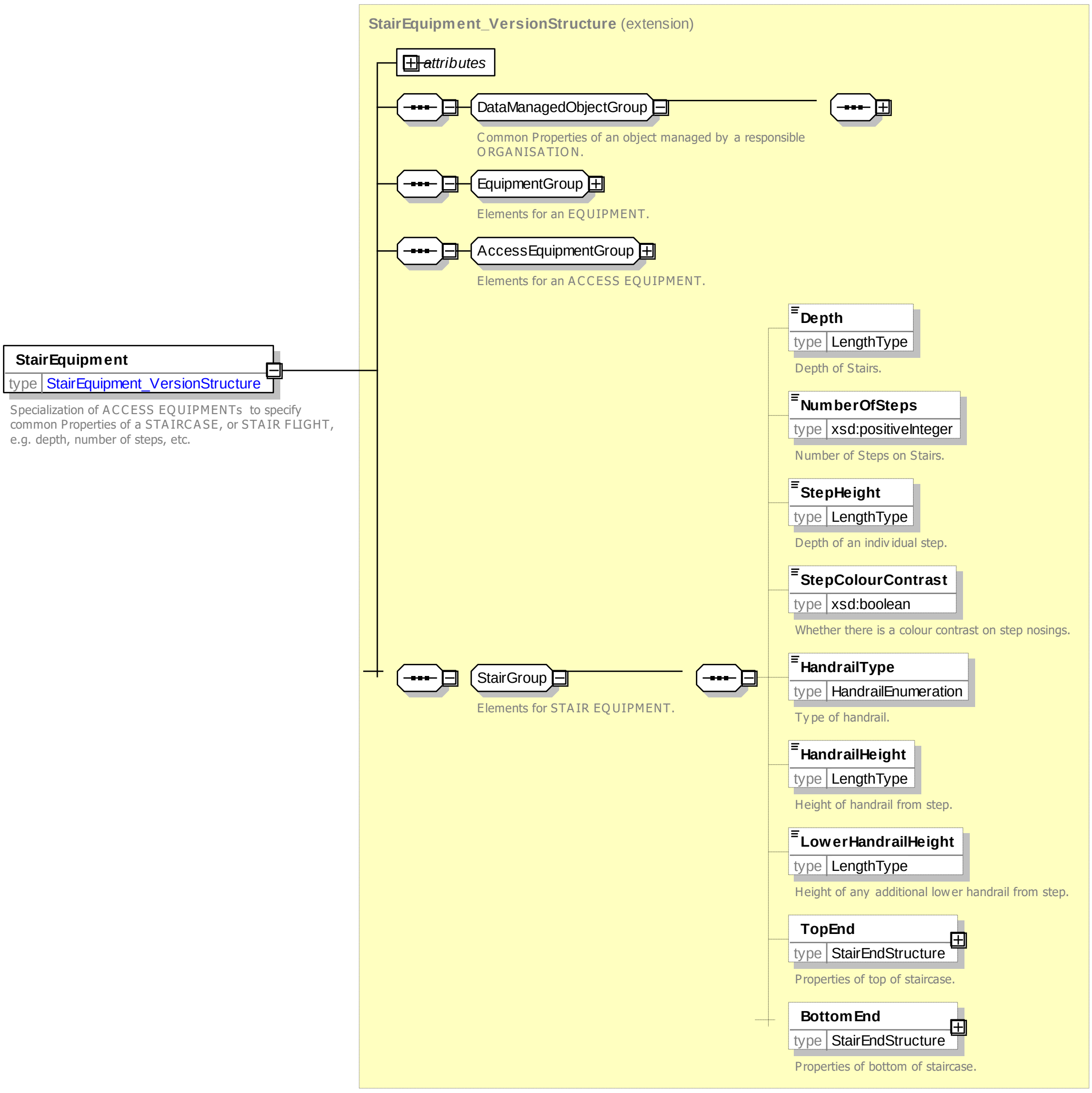


Figure 586 – *StairEquipment – XSD*

8.4.8.8.6.2 StairGroup – Model Element

Properties of a group of stairs.

Table 490 – *StairGroup – XML Group*

Classification	Name	Type	Cardinality	Description
	Depth	*LengthType*	0:1	Depth of stairs.
	NumberOfSteps	*xsd:integer*	0:1	Number of steps on stairs.
	StepHeight	*LengthType*	0:1	Depth of an individual step.
	StepColourContrast	*xsd:boolean*	0:1	Whether there is a colour contrast on step nosings.
«enum»	***TypeOfHandrail***	*TypeOfHandrailEnum*	0:1	Type of handrail. See allowed values below.

	HandrailHeight	*LengthType*	0:1	Height of handrail from step.
	LowerHandrailHeight	*LengthType*	0:1	Height of any additional lower handrail from step.
«cntd»	***TopEnd***	*StairEnd*	0:1	Properties of top of staircase.
«cntd»	***BottomEnd***	*StairEnd*	0:1	Properties of bottom of staircase.

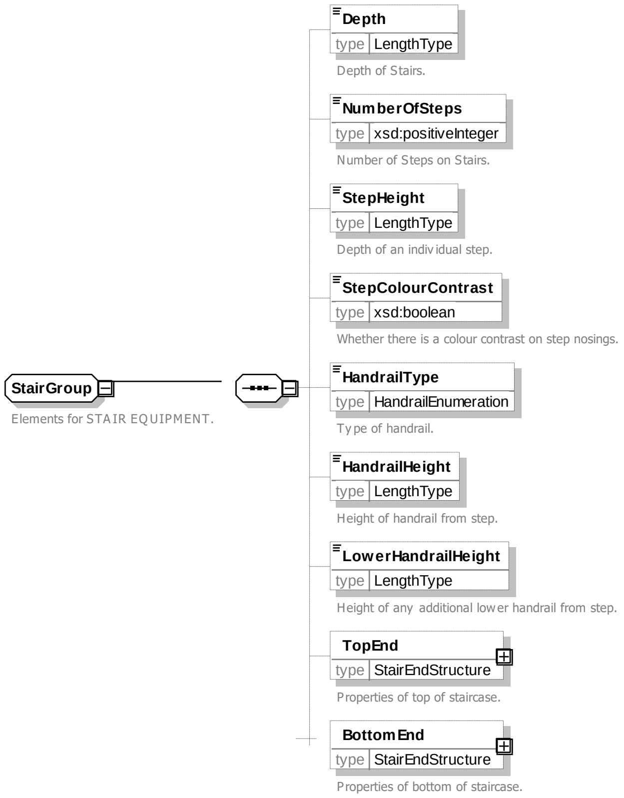


Figure 587 – *StairGroup* – *XSD*

TypeOfHandrail – Allowed values

Allowed values for HandrailType (*TypeOHandrailEnumeration*).

Table 491 – *TypeOfHandrail* – Allowed Values

Value	Description
none	No handrail
oneSide	Handrail one side
bothSides	Handrail both sides

8.4.8.8.6.3 StairEnd – Model Element

Properties of one end of a stair.

Table 492 – *StairEnd* – Structure

Classification	Name	Type	Cardinality	Description
	Continuing-Handrail	*xsd:boolean*	0:1	Whether there is a handrail that continues from previous section.
	TexturedSurface	*xsd:boolean*	0:1	Whether there is a textured ground surface.
	VisualContrast	*xsd:boolean*	0:1	Whether there is a colour contrast.

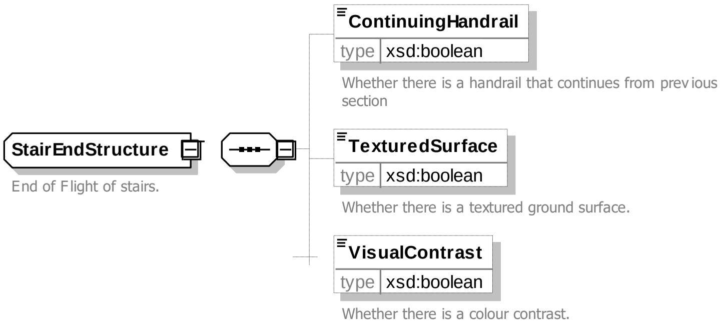


Figure 588 – *StairEnd – XSD*

8.4.8.8.6.4 ***StaircaseEquipment* – Model Element**

Specialisation of STAIR EQUIPMENT for STAIR CASEs.

Table 493 – *StaircaseEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*StairEquipment*	*::>*	STAIRCASE inherits from STAIR EQUIPMENT.
«PK»	***id***	*StaircaseIdType*	1:1	Identifier of STAIRCASE.
	Continuous-Handrail	*xsd:boolean*	0:1	Whether handrail is continuous across staircase.
	WithoutRiser	*xsd:boolean*	0:1	Whether openwork stairs (no riser) +v1.1
	SpiralStair	*xsd:boolean*	0:1	Whether stairs are spiral.
	NumberOfFlights	*xsd:integer*	0:1	Number of flights of stairs.
«cntd»	***flights***	*StairFlight*	0:*	Flight of stairs.

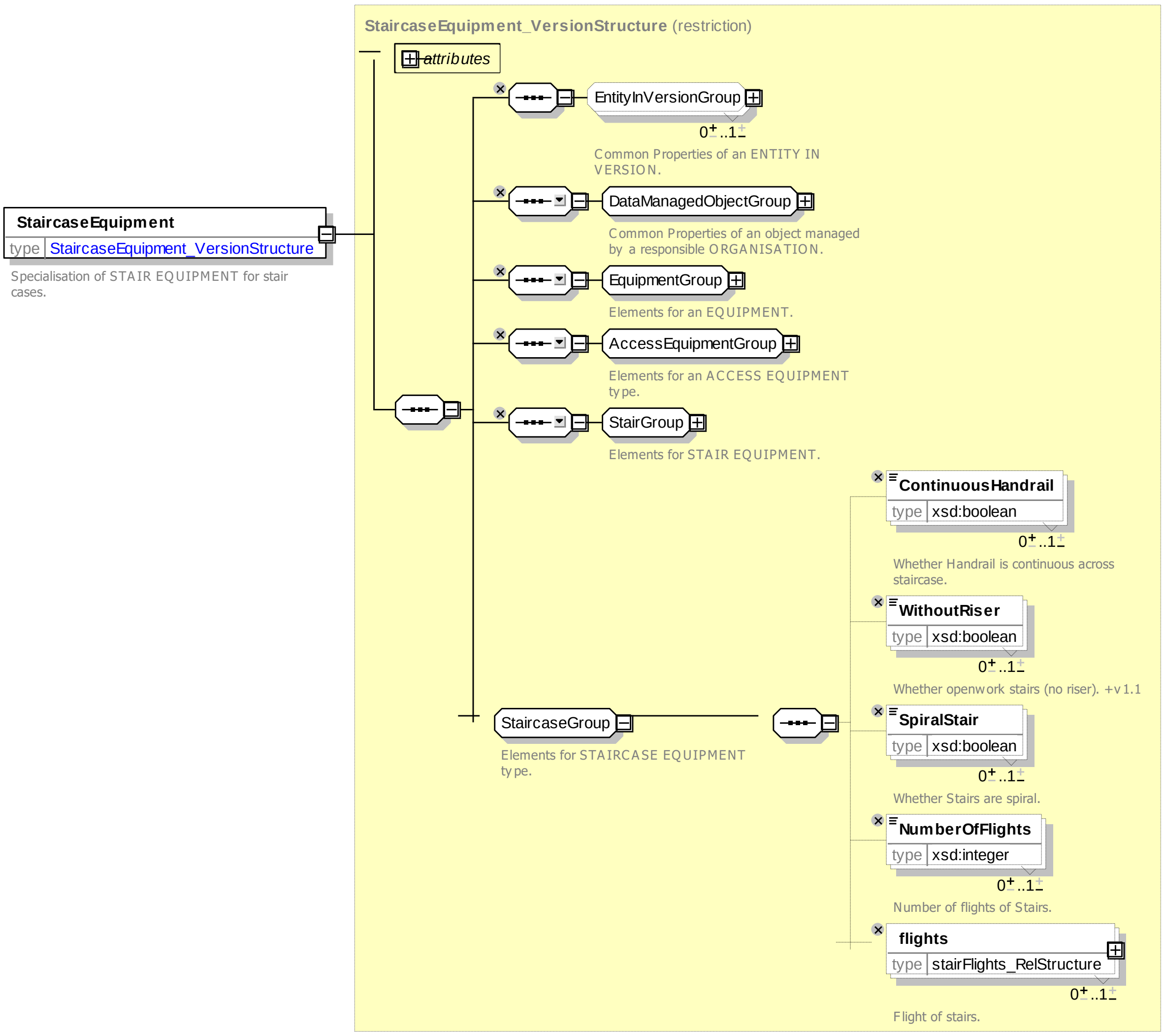


Figure 589 – *StairCaseEquipment – XSD*

8.4.8.8.6.5 StairFlight – Model Element

An individual flight of stairs.

Table 494 – *StairFlight* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VesrionedCHild*</u>	::>	STAIR FLIGHT inherits from VERSIONED CHILD.
«PK»	***id***	*StairFlightIdType*	1:1	Identifier of STAIR FLIGHT.
XGRP	***StairGroup***	<u>***xmlGroup***</u>	0:1	Common Stair Properties. See earlier.
	Continuous-Handrail	*xsd:boolean*	0:1	Whether Handrail is continuous from previous stage.

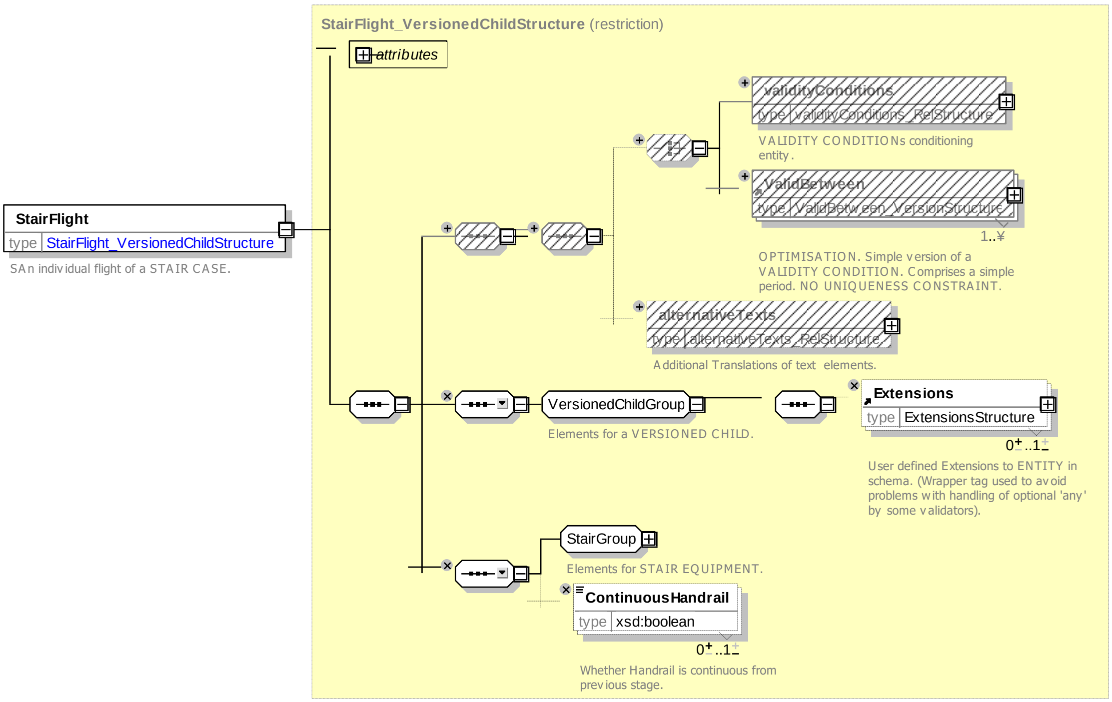


Figure 590 – *StairFlight – XSD*

8.4.8.8.7 EscalatorEquipment – Model Element

Specialisation of STAIR EQUIPMENT for ESCALATORs.

Table 495 – *EscalatorEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*StairEquipment*	::>	ESCALATOR inherits from STAIR EQUIPMENT.
«PK»	***id***	*EscalatorIdtype*	1:1	Identifier of ESCALATOR.
	TactileActuators	*xsd:boolean*	0:1	Whether there are tactile actuators for ESCALATOR.
	EnergySaving	*xsd:boolean*	0:1	Whether ESCALATOR is energy saving.
	DogsMustBe-Carried	*xsd:boolean*	0:1	Whether dogs must be carried on ESCALATOR. +v1.1
	EscalatorWith-Landing	*xsd:boolean*	0:1	Whether ESCALATOR has a landing. +v1.1

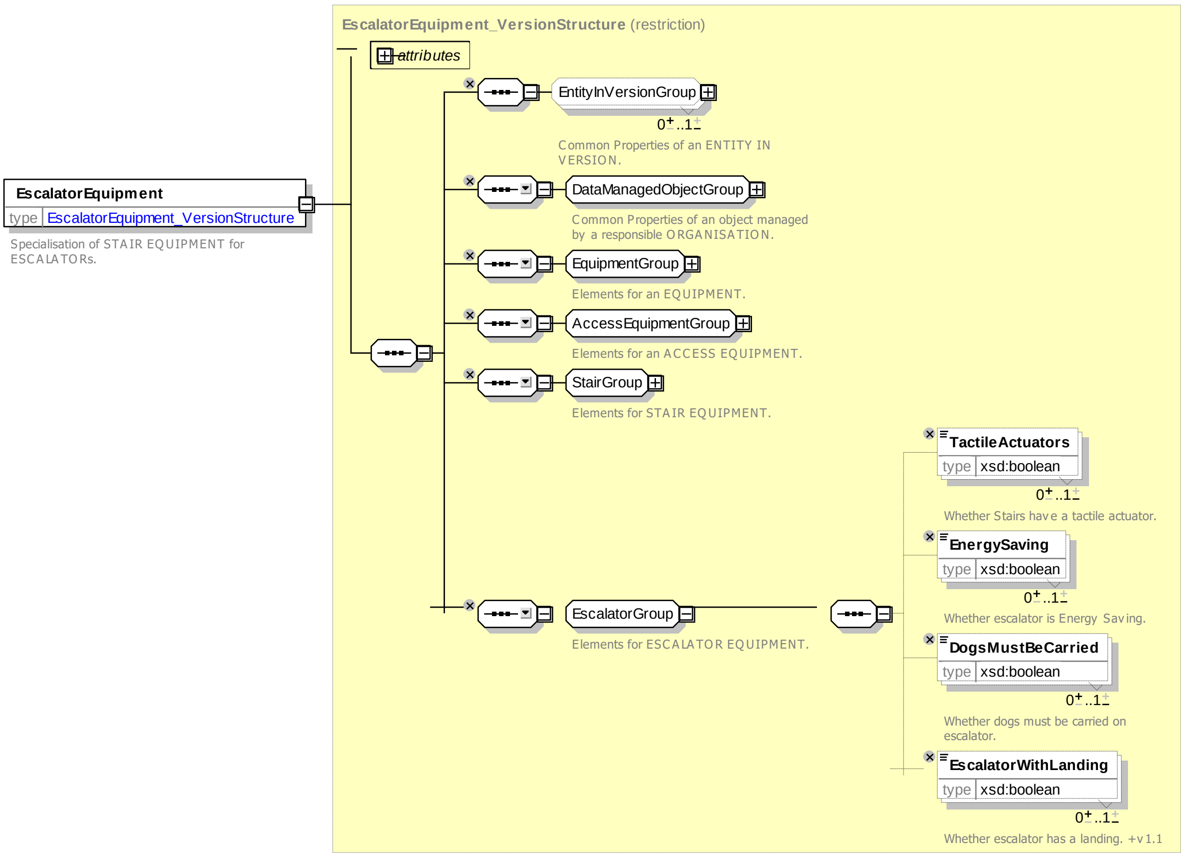


Figure 591 – *EscalatorEquipment – XSD*

8.4.8.8.7.1 *TravelatorEquipment* – Model Element

Specialisation of PLACE EQUIPMENT for TRAVELATORs (provides travelator attributes like speed, etc.).

Table 496 – *TravelatorEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceAccessEquipment*	*::>*	TRAVELATOR inherits from PLACE ACCESS EQUIPMENT.
«PK»	***id***	*TravelatorIdtype*	1:1	Identifier of TRAVELATOR.
	TactileActuators	*xsd:boolean*	0:1	Whether TRAVELATOR has tactile actuators.
	EnergySaving	*xsd:boolean*	0:1	Whether TRAVELATOR is energy saving.
	Speed	*SpeedType*	0:1	Speed of TRAVELATOR.
	Length	*LengthType*	0:1	Length of TRAVELATOR. +v1.1.
	Gradient	*xsd:decimal*	0:1	Slope for inclined moving walk - degrees. +v1.1.
	EnergySaving	*xsd:boolean*	0:1	Whether TRAVELATOR integrates an escalaor part. +v1.1.

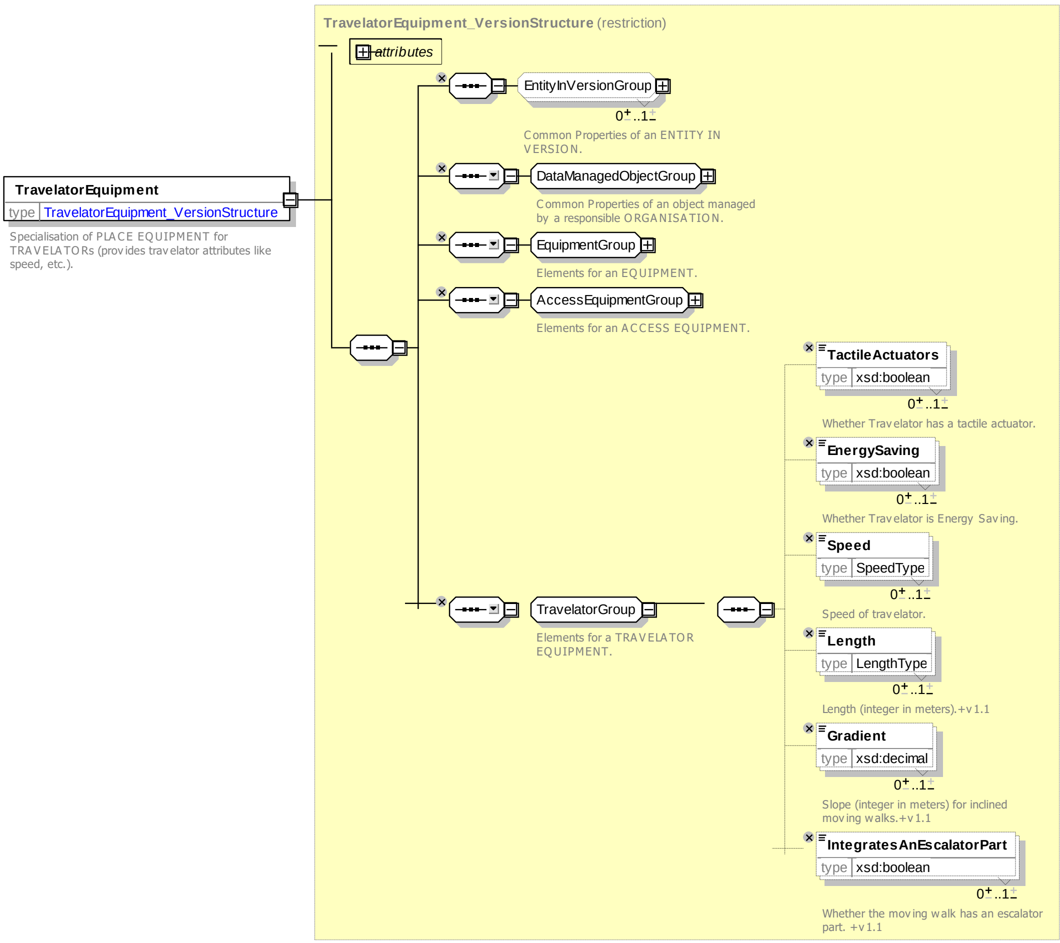


Figure 592 – *TravelatorEquipment* – *XSD*

8.4.8.8.8 *LiftEquipment* – Model Element

Specialisation of PLACE ACCESS EQUIPMENT for LIFTs (provides lift attributes like depth, maximum load, etc.).

Table 497 – *LiftEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PlaceAccess-Equipment*	*::>*	LIFT inherits from PLACE ACCESS EQUIPMENT.
«PK»	***id***	*Lift*	1:1	Identifier of LIFT.
XGRP	***LiftDimensionsGroup***	***xmlGroup***	0:1	Elements relating to LIFT dimensions.
XGRP	***LiftHandrailGroup***	***xmlGroup***	0:1	Elements relating to LIFT handrail.
XGRP	***LiftButtonGroup***	***xmlGroup***	0:1	Elements relating to LIFT buttons.
XGRP	***LiftPropertiesGroup***	***xmlGroup***	0:1	Elements relating to LIFT properties.

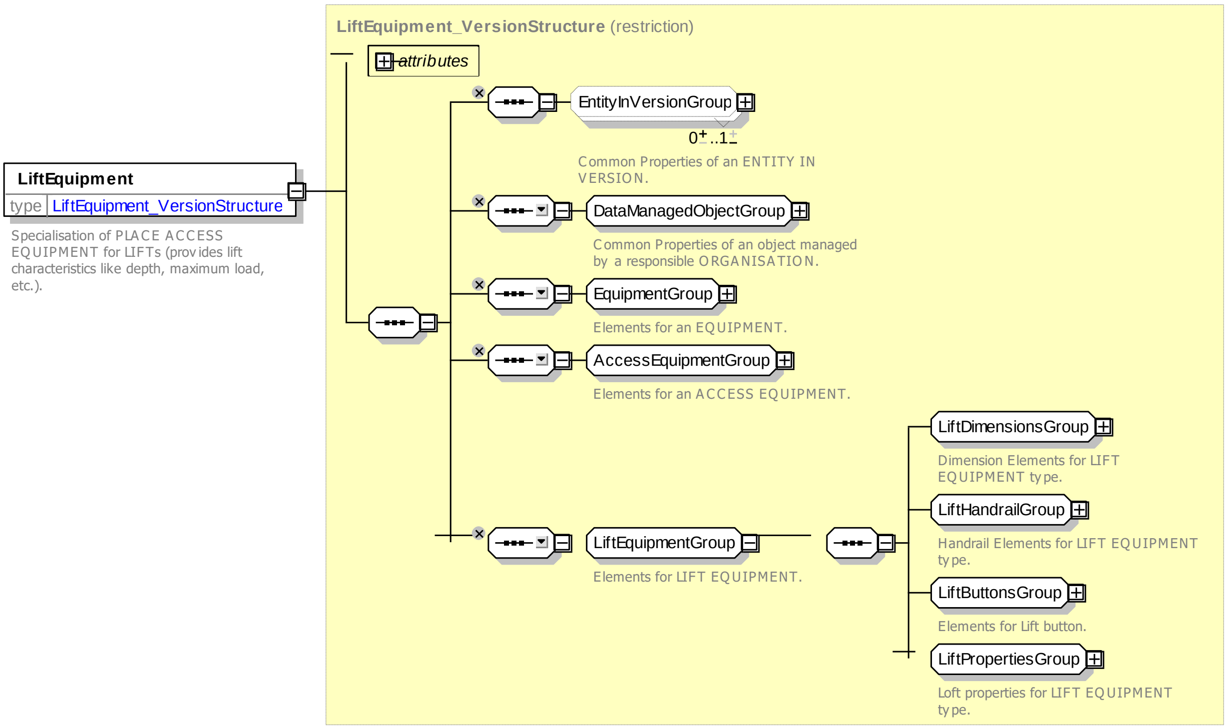


Figure 593 – *LiftEquipment – XSD*

8.4.8.8.8.1 LiftDimensionGroup – XML Group

Table 498 – *LiftDimensionGroup* – Group

Classification	Name	Type	Cardinality	Description
	Depth	*LengthType*	0:1	Depth of LIFT.
	MaximumLoad	*WeightType*	0:1	Maximum load of LIFT.
	WheelchairPassable	*xsd:boolean*	0:1	Whether LIFT is wheelchair passable.
	WheelchairTurningCircle	*LengthType*	0:1	Wheelchair turning circle of LIFT.
	InternalWidth	*LengthType*	0:1	Internal width of LIFT

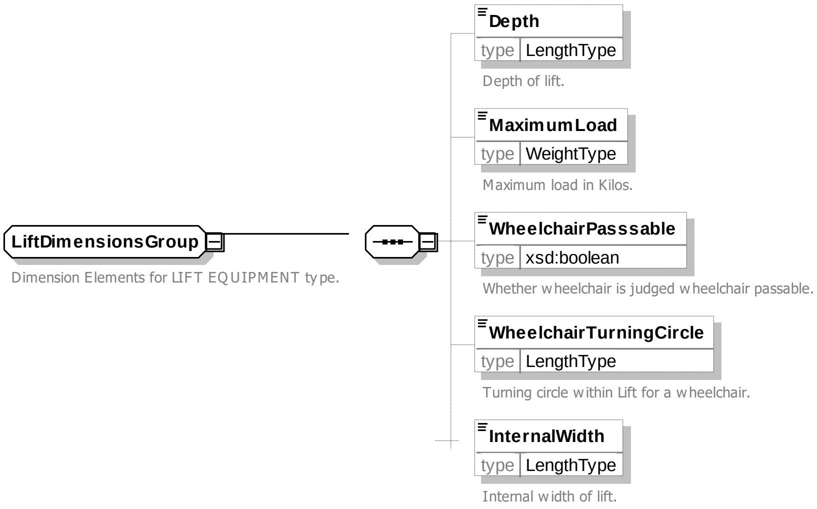


Figure 594 – *LiftDimensionGroup – XSD*

8.4.8.8.8.2 LiftHandrailGroup – XML Group

Table 499 – *LiftHandrailGroup* – Group

Classification	Name	Type	Cardinality	Description
«enum»	***HandrailType***	*TypeOfHandrail-Enum*	0:1	Type of handrail. See allowed values below.
	HandrailHeight	*LengthType*	0:1	Height of handrail from step.
	LowerHandrailHeight	*LengthType*	0:1	Height of any handrail from floor.

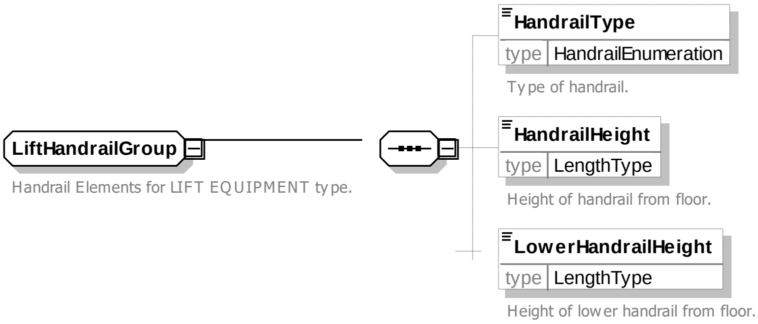


Figure 595 – *LiftHandrailGroup – XSD*

8.4.8.8.8.3 LiftButtonsGroup – XML Group

Table 500 – LiftButtonsGroup – Group

Classification	Name	Type	Cardinality	Description
	CallButtonHeight	*LengthType*	0:1	Height of call buttons of LIFT.

	DirectionButtonHeight	*LengthType*	0:1	Height of any direction button from floor.
	RaisedButtons	*xsd:boolean*	0:1	Whether LIFT has raised buttons.
	BrailleButtons	*xsd:boolean*	0:1	Whether LIFT has braille buttons.
	TactileGroundFloordButton	*xsd:boolean*	0:1	ndicates whether there is a tactile marker on the go to ground floor button. +v1.1
	GroundMArlAlignedWith-Buttons	*xsd:boolean*	0:1	Indicates a tactile marker on floor under the buttons (or aligned with) +v1.1.

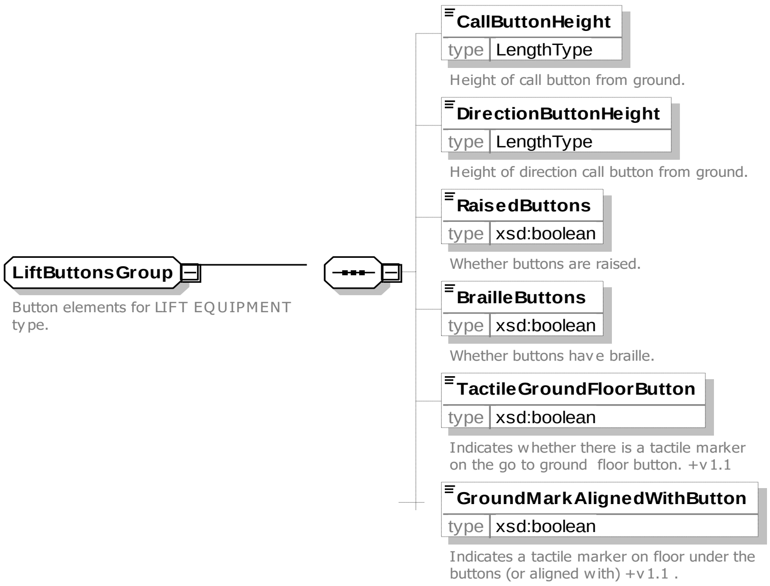


Figure 596 – *LiftButtonsGroup – XSD*

8.4.8.8.8.4 LiftPropertiesGroup – XML Group

Table 501 – *LiftPropertiesGroup* – Group

Classification	Name	Type	Cardinality	Description
	ThroughLoader	*xsd:boolean*	0:1	Whether LIFT is through loader.
	MirrorOnOppositeSide	*xsd:boolean*	0:1	Whether LIFT has a mirror on opposite side.
	Attendant	*xsd:boolean*	0:1	Whether LIFT has attendant.
	Automatic	*xsd:boolean*	0:1	Whether LIFT is automatic.
	ExternalFloorSelection	*xsd:boolean*	0:1	Whether the floor selection is made outside the lift.+v1.1
	AlarmButton	*xsd:boolean*	0:1	Whether LIFT has alarm Button.
	TactileActuators	*xsd:boolean*	0:1	Whether LIFT has tactile actuators.

	AudioAnnouncements	*xsd:boolean*	0:1	Whether LIFT has audio announcements
	SignageToLift	*xsd:boolean*	0:1	Whether LIFT has signage.
	SuitableForCycles	*xsd:boolean*	0:1	Whether LIFT is suitable for a cycle.

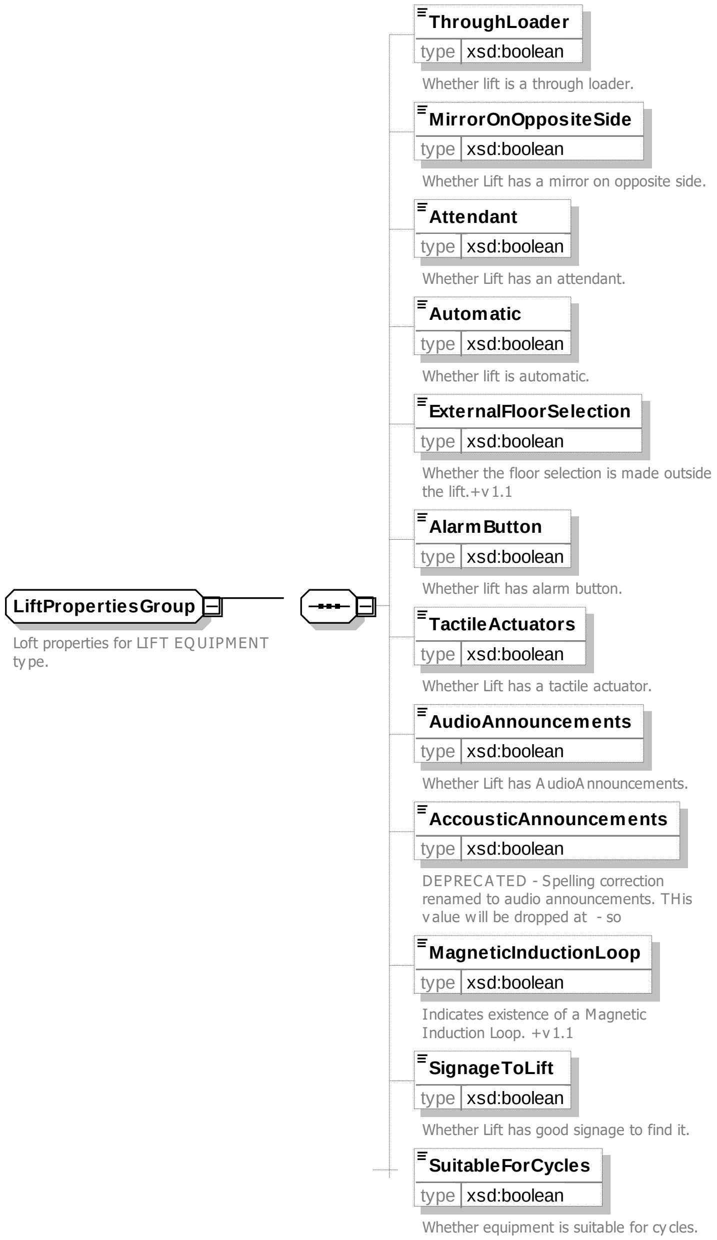


Figure 597 – *LiftPropertiesGroup – XSD*

8.4.8.9 Sign Equipment

Stations typically have carefully designed signage located at critical decision points within the interchange – for example '*District Line Northbound'*, *'London Trains platform 3 & 4', 'Exit to Centre Court Shopping Centre*', etc. The *NeTEx* model treats signs as a kind of EQUIPMENT whose contents and locations can be exactly specified. This makes it possible to create detailed journey plan instructions, augmented reality guides and other applications for the interchange that refer to what the user can see if they are positioned within it.

8.4.8.9.1 SIGN EQUIPMENT – Conceptual MODEL

SIGN EQUIPMENT can define signs visible to passengers at places in a SITE, such as PLACE SIGNS, DIRECTION SIGNs, etc. these can be used to provide guidance. Place names and LINE names may be derived from PLACEs and LINEs.

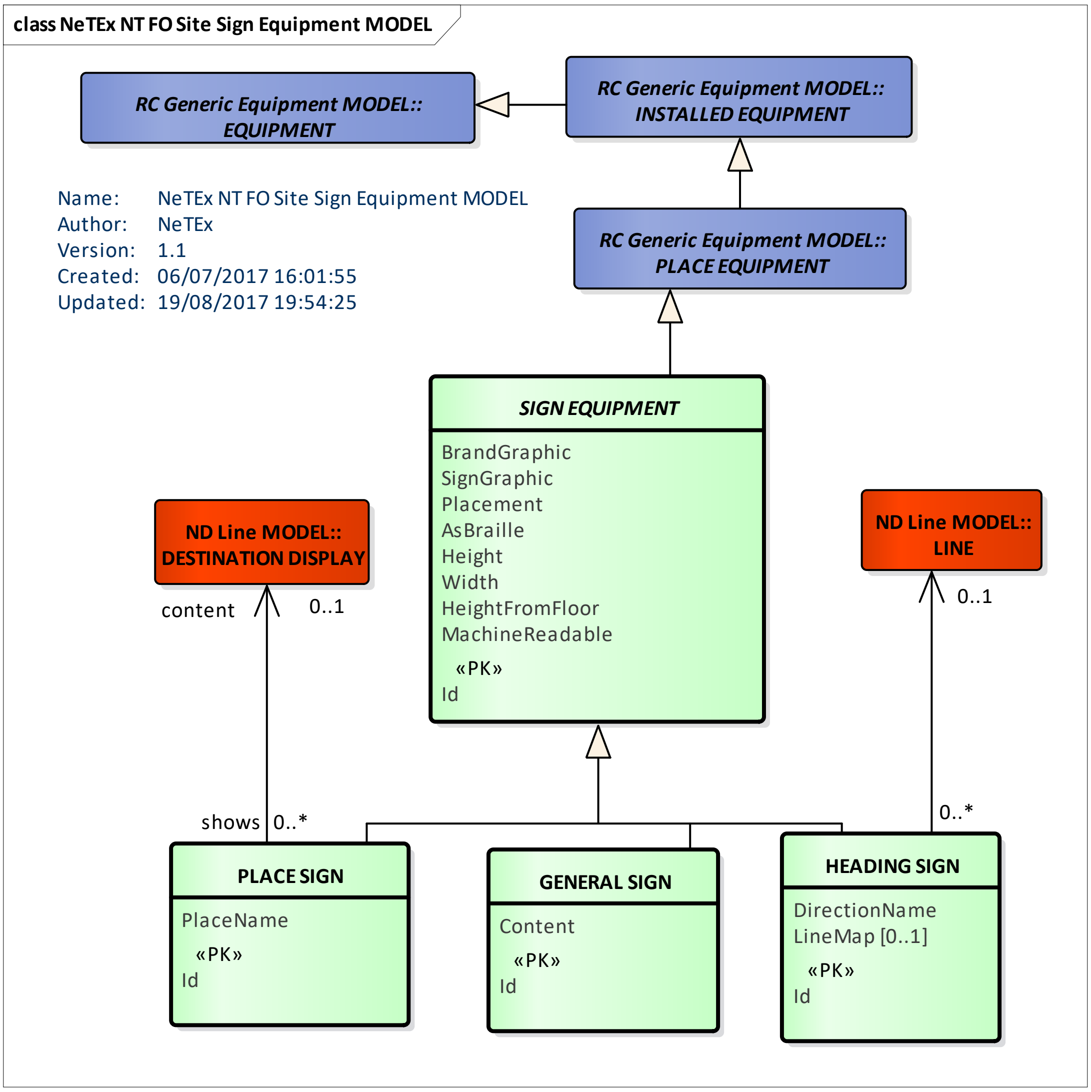


Figure 598 – SIGN EQUIPMENT – Conceptual MODEL (UML)

8.4.8.9.2 Sign Equipment – Physical Model

The following figure shows detailed attributes of the SIGN EQUIPMENT model elements.

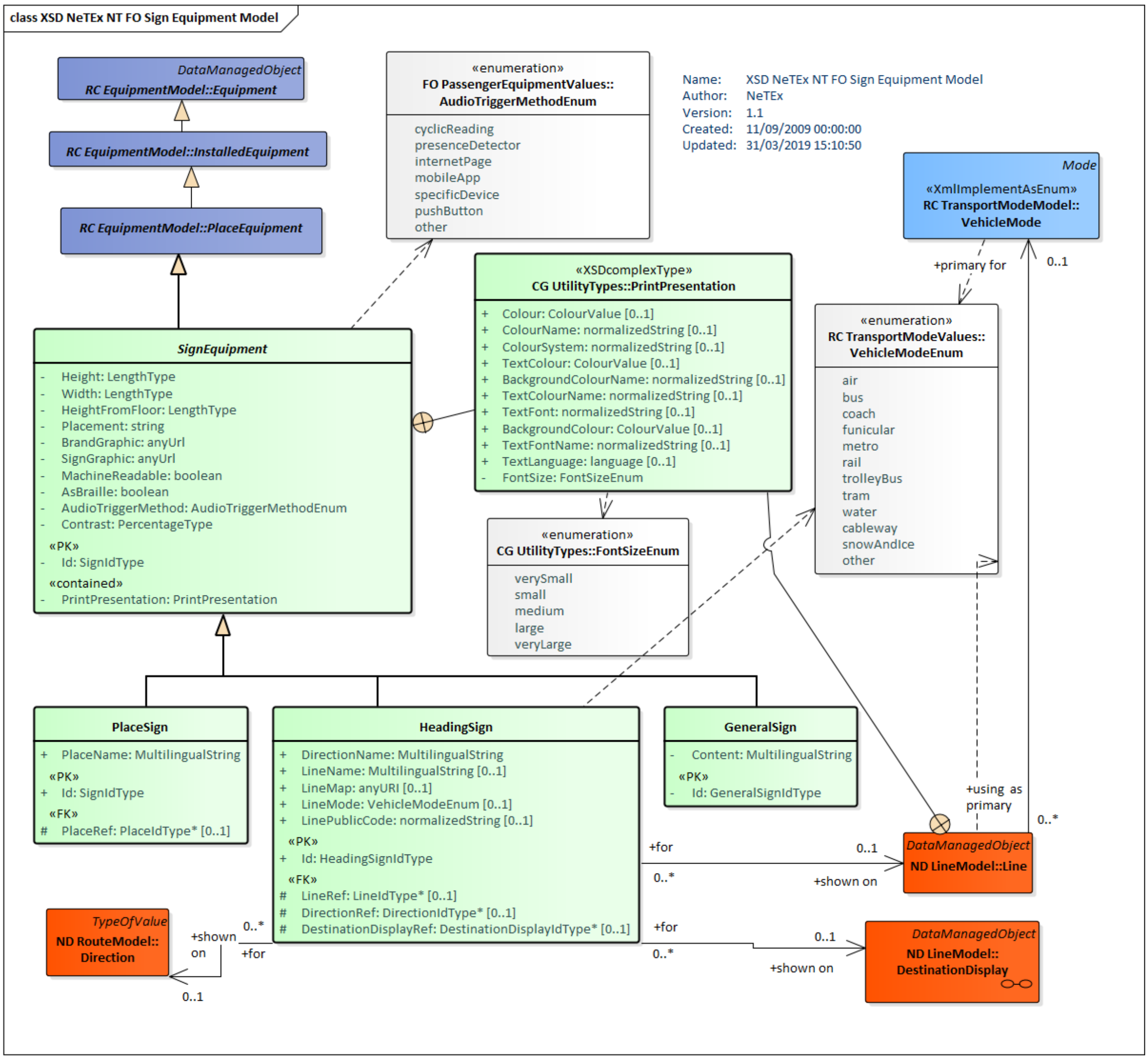


Figure 599 – Sign Equipment – Physical Model (UML) – Detail

8.4.8.9.3 Sign Equipment – Attributes and XSD

Generic specialization of PLACE EQUIPMENT for SIGNs (heading signs, etc.).

Table 502 – *SignEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PlaceEquipment*	::>	SIGN inherits from PLACE EQUIPMENT.
«PK»	***id***	*SignIdType*	1:1	Identifier of SIGN.
	Height	*LengthType*	0:1	Height of SIGN from bottom of sign.
	Width	*LengthType*	0:1	Width of SIGN.
	HeightFromFloor	*LengthType*	0:1	Height of SIGN above ground.

	Placement	*xsd:string*	0:1	Description of placement of SIGN.
	BrandGraphic	*xsd:anyUrl*	0:1	URL for Brand graphic shown on SIGN.
	SignGraphic	*xsd:anyUrl*	0:1	URL for Sign graphic shown on SIGN.
	MachineReadable	*xsd:boolean*	0:1	Whether sign is machine readable.
	AsBraille	*xsd:boolean*	0:1	Whether SIGN has braille section.
«enum»	***AudioTrigger-Method***	*AudioTriggerMethodEnum*	0:1	How the display may be read by the audio tool. See allowed values above under PASSENGER SAFETY EQUIPMENT. See allowed values below. +v1.1 z
«enum»	***Printed-Presentation***	*PrintPresentation*	0:1	Printed Presentation characteristics including font size. See Resuable Components utilty Model. +v1.1.
	Contrast	*PercentageType*	0:1	Luminance gap between text and background. A ratio of at least 3 is expected. +v1.1

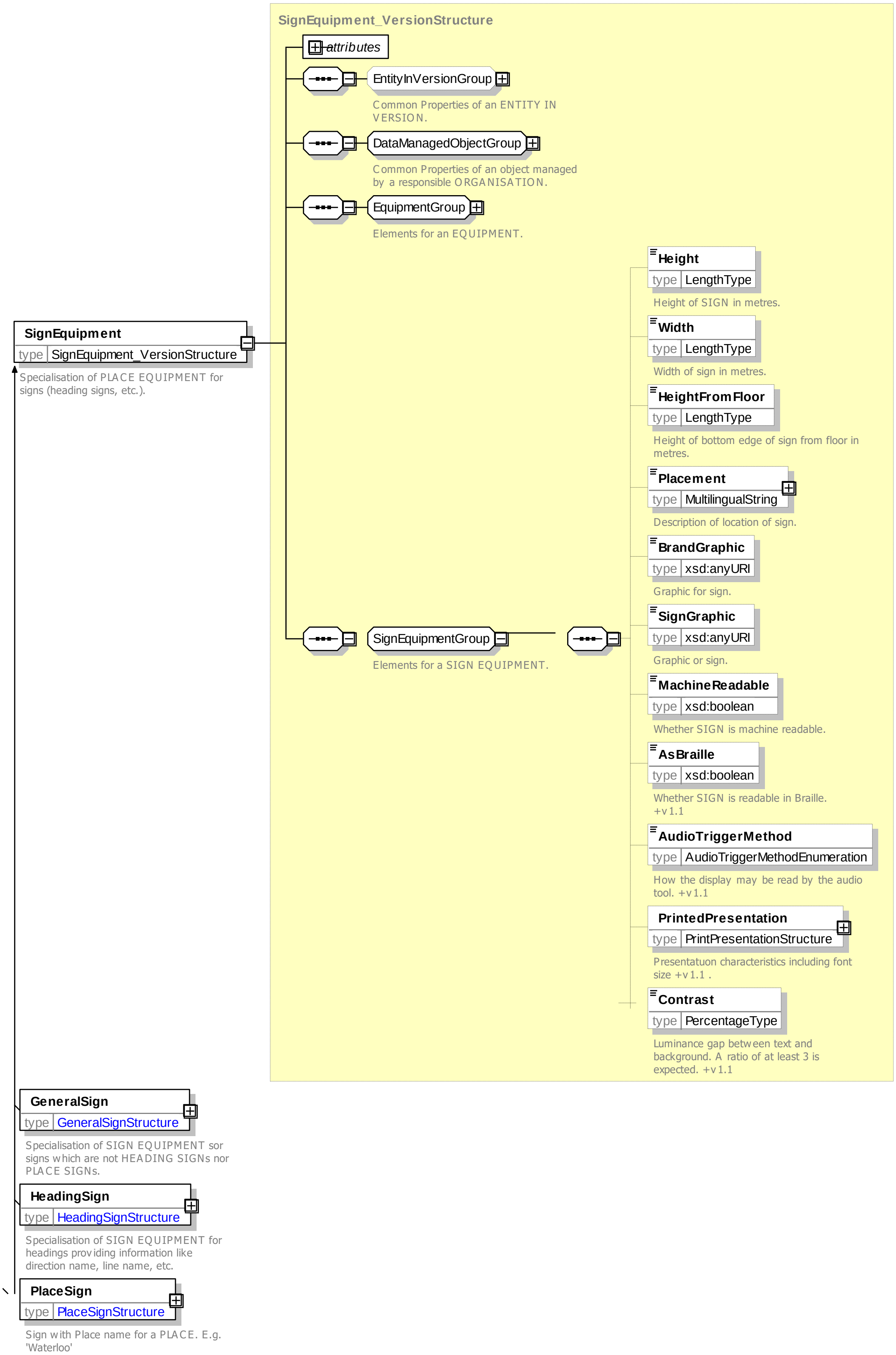

SignEquipment_VersionStructure
attributes
EntityInVersionGroup
Common Properties of an ENTITY IN VERSION.
DataManagedObjectGroup
Common Properties of an object managed by a responsible ORGANISATION.
EquipmentGroup
Elements for an EQUIPMENT.
SignEquipment
type SignEquipment_VersionStructure
Specialisation of PLACE EQUIPMENT for signs (heading signs, etc.).
Height
type LengthType
Height of SIGN in metres.
Width
type LengthType
Width of sign in metres.
HeightFromFloor
type LengthType
Height of bottom edge of sign from floor in metres.
Placement
type MultilingualString
Description of location of sign.
BrandGraphic
type xsd:anyURI
Graphic for sign.
SignEquipmentGroup
Elements for a SIGN EQUIPMENT.
SignGraphic
type xsd:anyURI
Graphic or sign.
MachineReadable
type xsd:boolean
Whether SIGN is machine readable.
AsBraille
type xsd:boolean
Whether SIGN is readable in Braille. +v1.1
AudioTriggerMethod
type AudioTriggerMethodEnumeration
How the display may be read by the audio tool. +v1.1
PrintedPresentation
type PrintPresentationStructure
Presentatuon characteristics including font size +v1.1 .
Contrast
type PercentageType
Luminance gap between text and background. A ratio of at least 3 is expected. +v1.1
GeneralSign
type GeneralSignStructure
Specialisation of SIGN EQUIPMENT sor signs which are not HEADING SIGNs nor PLACE SIGNs.
HeadingSign
type HeadingSignStructure
Specialisation of SIGN EQUIPMENT for headings providing information like direction name, line name, etc.
PlaceSign
type PlaceSignStructure
Sign with Place name for a PLACE. E.g. 'Waterloo'

Figure 600 – *SignEquipment* – *XSD*

8.4.8.9.3.1 HeadingSign – Model Element

Specialisation of SIGN for HEADING SIGNs providing information like direction name, line name, etc.

Table 503 – *HeadingSign* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SignEquipment*	*::>*	HEADING SIGN inherits from SIGN EQUIPMENT.
«PK»	***id***	*HeadingSignIdType*	1:1	Identifier of HEADING SIGN.
	PlaceName	*MultilingualString*	0:1	PLACE name on HEADING SIGN.
«FK»	***LineRef***	*LineRef*	0:1	LINE referenced by HEADING SIGN.
	LineName	*MultilingualString*	0:1	LINE name on HEADING SIGN.
	PublicCode	*xsd:normalizedString*	0:1	LINE code on HEADING SIGN.
«enum»	***TransportMode***	*VehicleModeEnum*	0:1	MODE of LINE referenced by sign. See TRANSPORT MODE for allowed values.
«enum»	***TransportSubMode***	*TransportSubMode*	0:1	SUBMODE of LINE referenced by sign.
	LineMap	*xsd:anyURI*	0:1	URL of Map associated with sign.
	DirectionRef	*DirectionRef*	1:1	DIRECTION that sign shows.
	DirectionName	*MultilingualString*	1:1	DIRECTION name that sign shows.
«FK»	***DestinationDisplay-Ref***	*LineRef*	0:1	Destination DISPLAY referenced by HEADING SIGN.

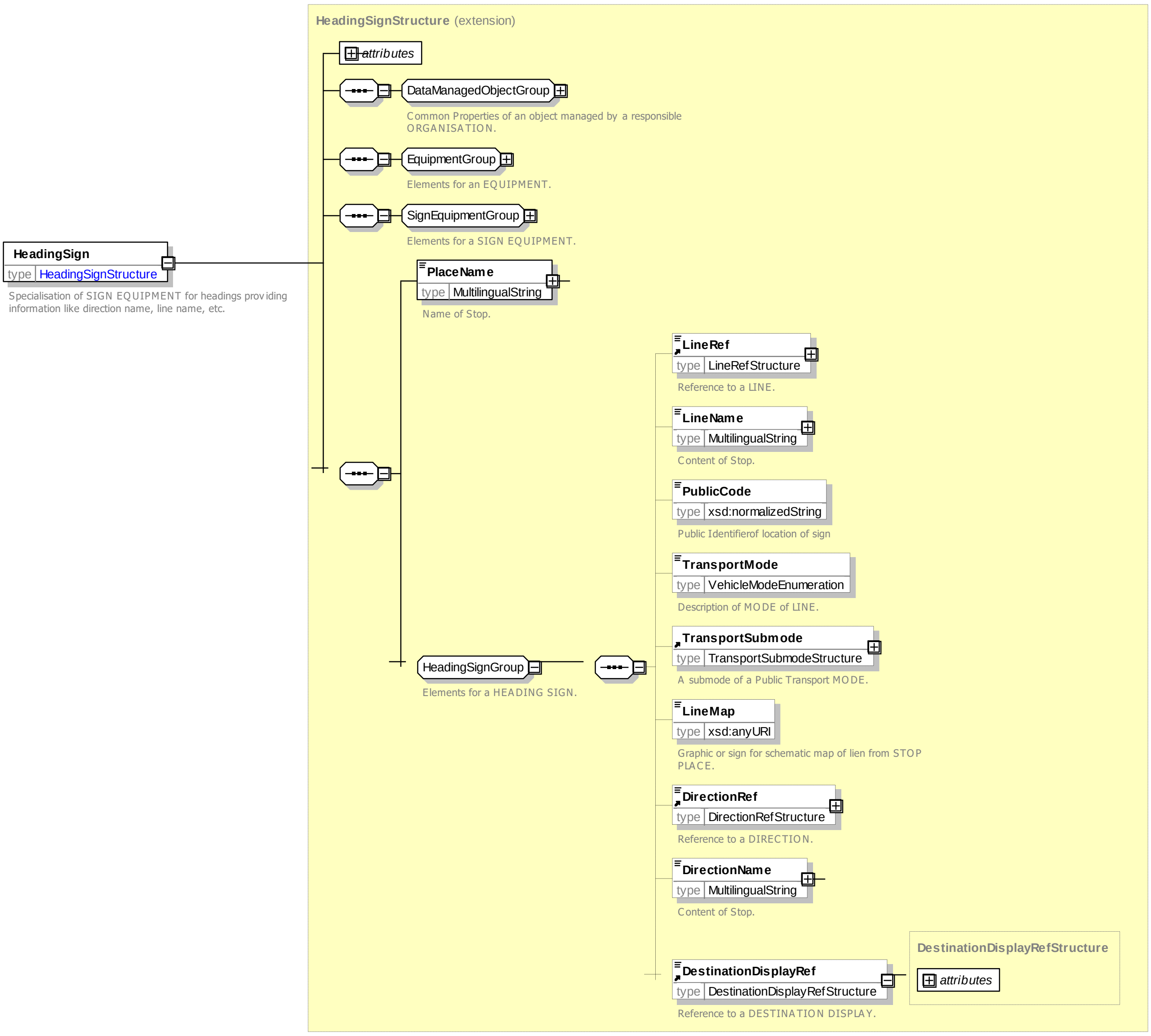


Figure 601 – *HeadingSign* – *XSD*

8.4.8.9.3.2 GeneralSign – Model Element

Specialisation of SIGN which are not HEADING SIGNs or regular SIGNs.

Table 504 – *GeneralSign* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*SignEquipment*	::>	HEADING SIGN inherits from SIGN EQUIPMENT.
«PK»	***id***	*GeneralSignIdType*	1:1	Identifier of GENERAL SIGN.
	Content	*MultilingualString*	0:1	Content of GENERAL SIGN.
«enum»	***SignContentType***	*SignContentEnum*	0:1	Type of content. See allowed values below.

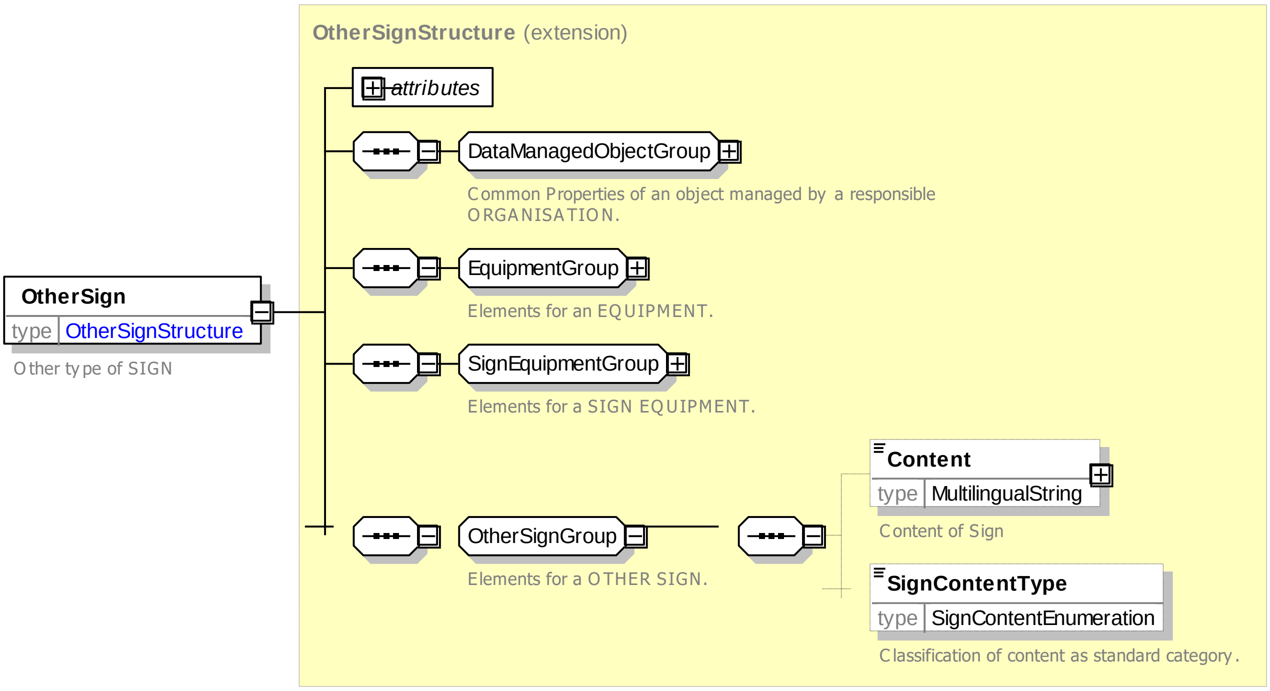


Figure 602 – *GeneralSign – XSD*

SignContentType – Allowed values

Allowed values for ***SignContentType*** (*SignContentTypeEnumeration*).

Table 505 – *SignContentType* – Allowed Values

Value	Description
entrance	Entrance
exit	Exit
emergencyExit	Emergency exit
transportMode	Transport mode
noSmoking	No smoking
tickets	Tickets
assistance	Assistance
sosPhone	SOS phone
touchPoint	Touch point
meetingPoint	Meeting point
transportModePoint	Transpor tmode point
other	Other

8.4.8.9.3.3 PlaceSign – Model Element

Sign with the name of a STOP PLACE.

Table 506 – *PlaceSign* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*SignEquipment*</u>	::>	PLACE SIGN inherits from SIGN EQUIPMENT.

«PK»	***id***	*SignIdType*	1:1	Identifier of PLACE SIGN.
	PlaceName	*MultilingualString*	1:1	Place name shown on SIGNs.
«FK»	***PlaceRef***	(*PlaceRef*)	0:1	Reference to Place shown on SIGN.

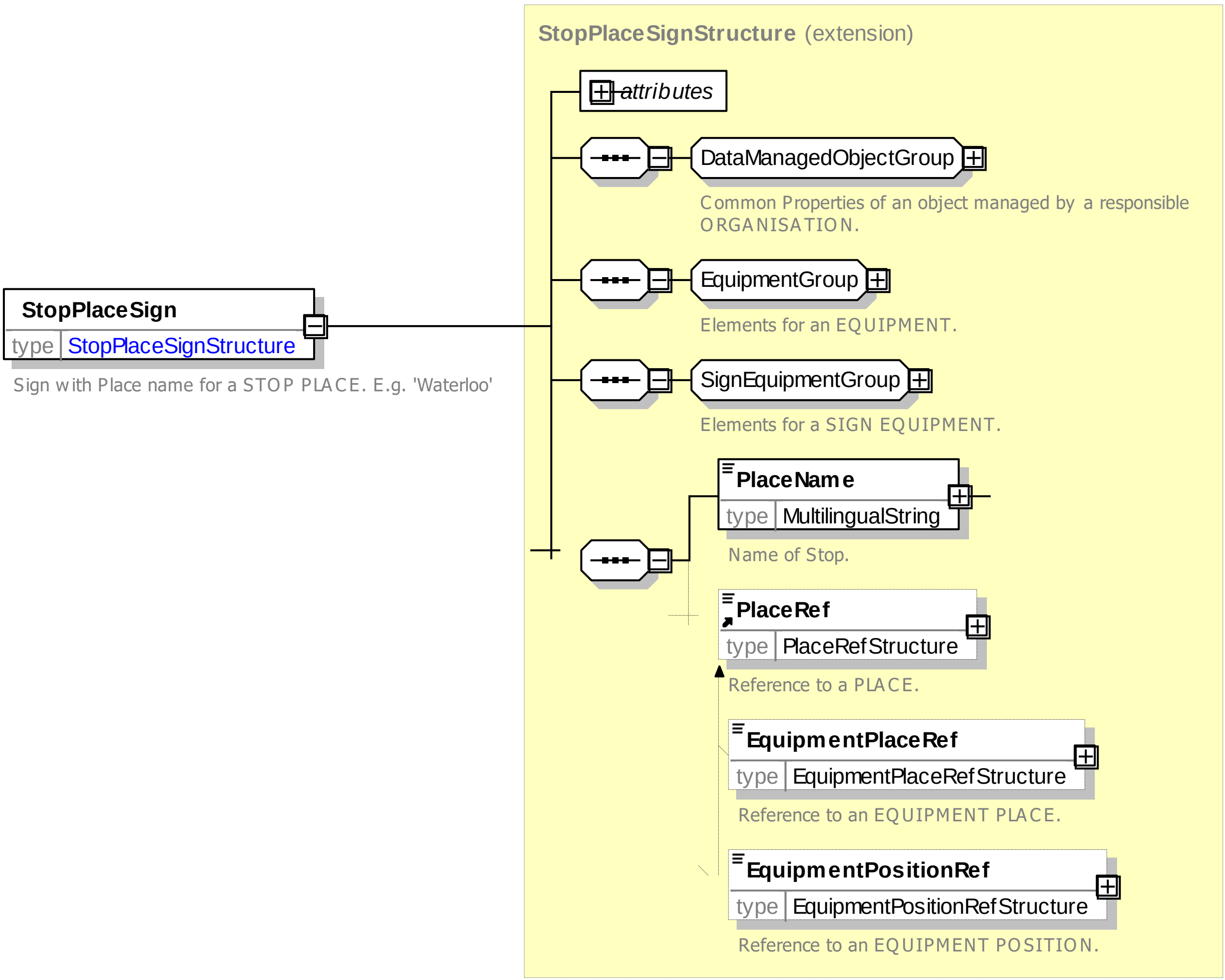


Figure 603 – *PlaceSign – XSD*

8.4.8.10 Ticketing Equipment elements

8.4.8.10.1 TICKETING EQUIPMENT – Conceptual MODEL

The TICKETING EQUIPMENT describes elements relevant for ticketing. The following figure shows the Conceptual model for TICKETING EQUIPMENT.

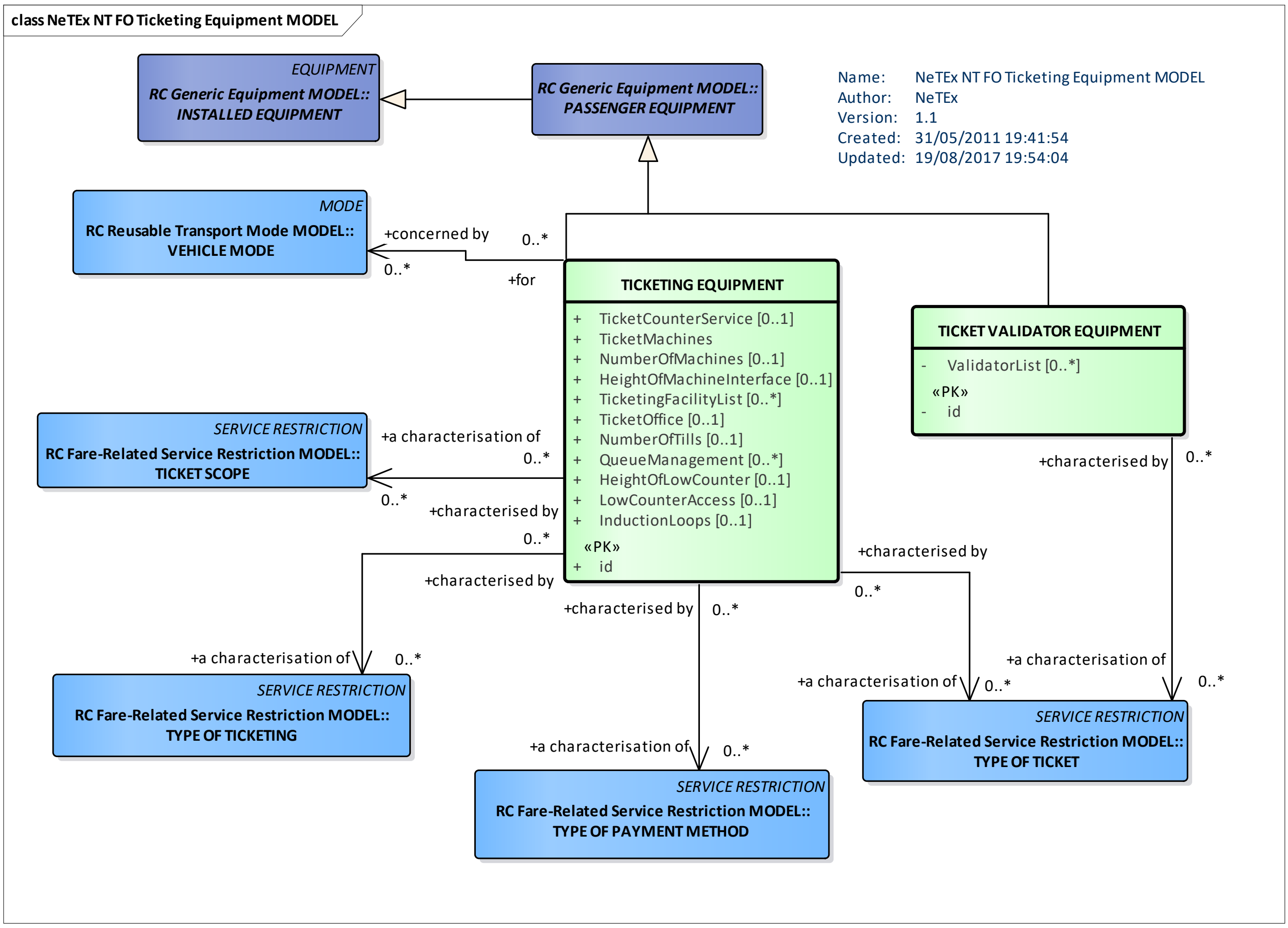


Figure 604 – TICKETING EQUIPMENT – Conceptual MODEL (UML)

8.4.8.10.2 Ticketing Equipment – Physical Model

The following shows detailed attributes of the TICKETING EQUIPMENT model elements.

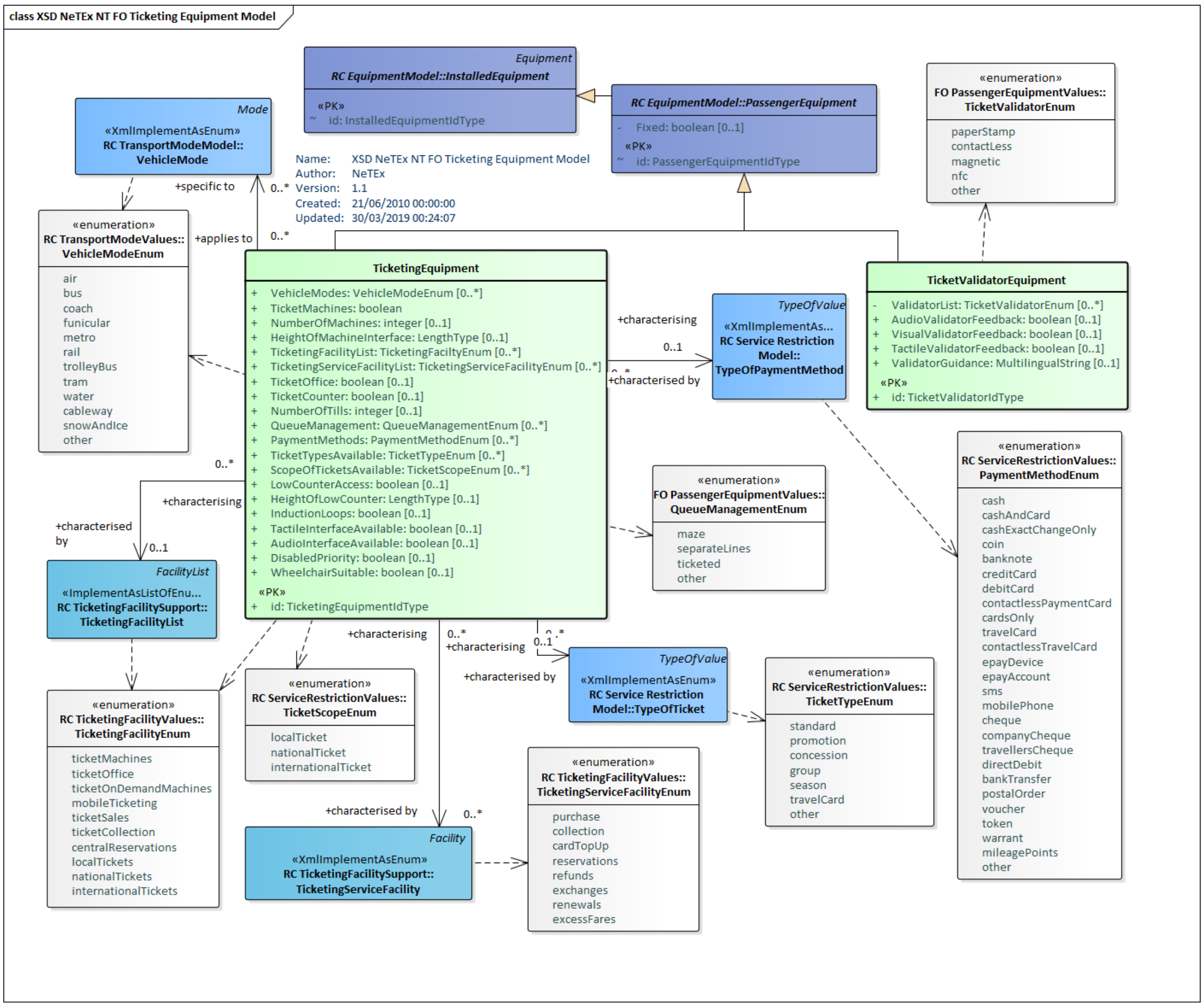


Figure 605 – Ticketing Equipment – Physical Model (UML)

8.4.8.10.3 Ticketing Equipment – Attributes and XSD

8.4.8.10.3.1 TicketValidatorEquipment – Model Element

Specialisation of PASSENGER EQUIPMENT (PLACE EQUIPMENT) for TICKET VALIDATOR.

Table 507 – *TicketValidatorEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PassengerEquipment*	::>	TICKET VALIDATOR EQUIPMENT inherits from PASSENGER EQUIPMENT.
«PK»	***id***	*TicketValidatorIdType*	1:1	Identifier of TICKET VALIDATOR EQUIPMENT.
«enum»	***TicketValidator-Type***	*TicketValidatorEnum*	0:*	Type of TICKET VALIDATOR. See allowed values below.
	AudioValidation-Feedback	xsd:boolean	0:1	Whether there is audio feedback for validation. +v1.1

	VisualValidation-Feedback	xsd:boolean	0:1	Whether there is visual feedback for validation. +v1.1
	TactileValidation-Feedback	xsd:boolean	0:1	Whether there is tactile feedback for validation. +v1.1
	Validation-Guidance	MultilingualString	0:1	Text describing how to find and use the validator. +v1.1

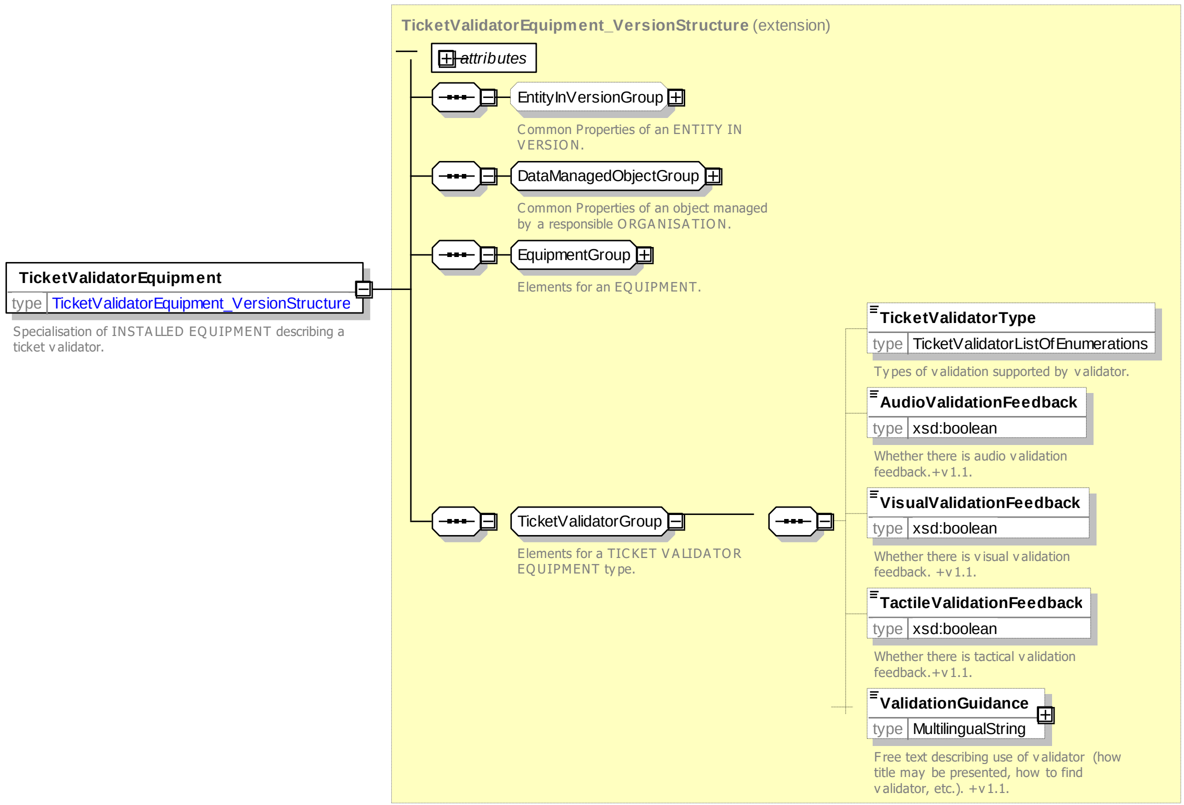


Figure 606 – *TicketValidatorEquipment – XSD*

TicketValidatorType – Allowed values

Allowed values for ***TicketValidatorType*** *(TicketValidatorEnumeration).*

Table 508 – *TicketValidatorType* – Allowed Values

Value	Description
paperStamp	Paper stamp
contactLess	Contact less card
magnetic	Magnetic strip
nfc	Near field communication
other	Other validation mechanism

8.4.8.10.3.2 TicketingEquipment – Model Element

Specialisation of PASSENGER EQUIPMENT for TICKETING.

Table 509 – *TicketingEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PassengerEquipment*	::>	TICKETING EQUIPMENT inherits from PASSENGER EQUIPMENT.
«PK»	***id***	*TicketingEquipmentIdType*	1:1	Identifier of TICKETING EQUIPMENT.
«enum»	***VehicleModes***	*VehicleModeEnum*	0:*	MODEs for which ticketing services apply. See TRANSPORT MODE fro allowed values.
XGRP	***TicketingEquipment PropertiesGroup***	***xmlGroup***	0:1	General properies of TICKETING EQUIPMENT.
«enum»	***PaymentMethods***	*PaymentMethodEnum*	0:*	Payment methods allowed. See allowed values in RC Service Restrictirons.
«enum»	***TicketTypes-Available***	*TicketTypeEnum*	0:*	Types of ticket available. See allowed values below.
«enum»	***ScopeOfTickets-Available***	*TicketScopeEnum*	0:*	Scope of the available ticets; local, national, international, etc. See allowed values below.
XGRP	***TicketingEquipment AccesibilityGroup***	***xmlGroup*** *e*	0:1	TICKETING EQUIPMENT elements relating to accessibility.

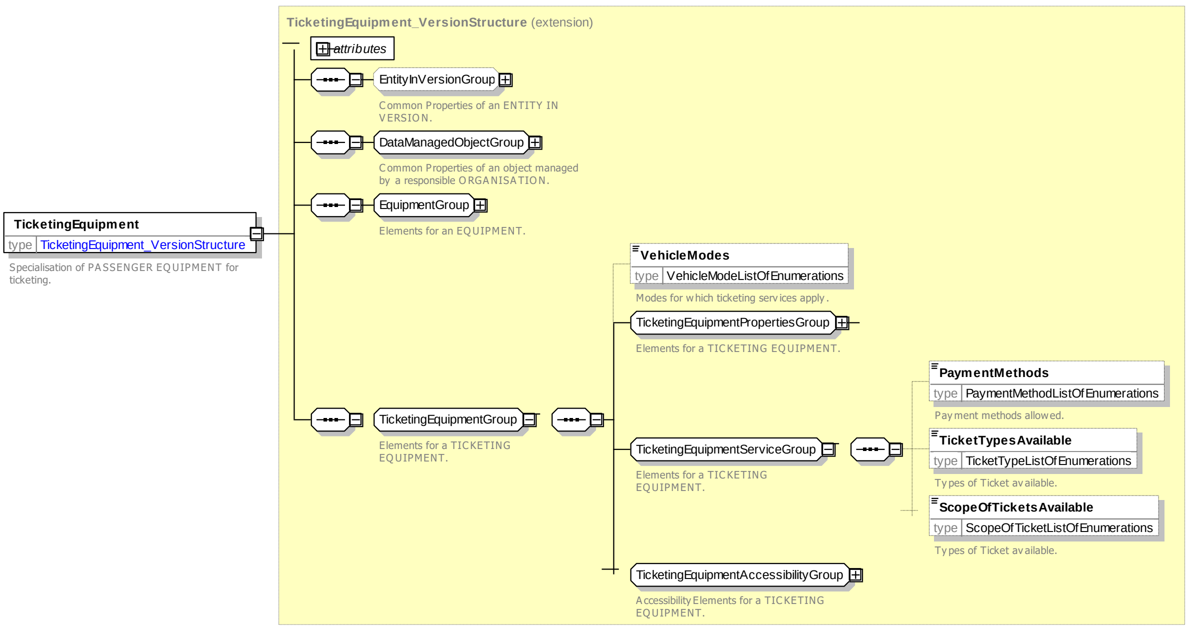


Figure 607 – *TicketingEquipment – XSD*

8.4.8.10.3.3 TicketingEquipmentPropertiesGroup – XML Group

Elements characterising f TICKETING EQUIPMENT.

Table 510 – *TicketingEquipmentPropertiesGroup – Group*

Classification	Name	Type	Cardinality	Description
	TicketMachines	*xsd:boolean*	0:1	Whether there are ticket machines.
	NumberOfMachines	*xsd:integer*	0:1	Number of ticket machines.
	HeightOfMachine-Interface	*LengthType*	0:1	Height of the ticket machine interface. +v1.1
«enum»	***TicketingFacilityList***	*TicketingFaciltyEnum*	0:*	Types of TICKETING available. See allowed values below.
«enum»	***TicketingService-FacilityList***	*TicketingServiceFacility-Enum*	0:*	Types of ticketing service available. See allowed values in RC Site facilities.
	TicketOffice	*xsd:boolean*	0:1	Whether there is a distinct ticket office.
	TicketCounter	*xsd:boolean*	0:1	Whether there is a ticket counter.
	NumberOfTills	*xsd:integer*	0:1	Number of tills selling tickets.
«enum»	***QueueManagement***	*QueueManagement*Enum	0:*	Type of Queue Management. See allowed values below.

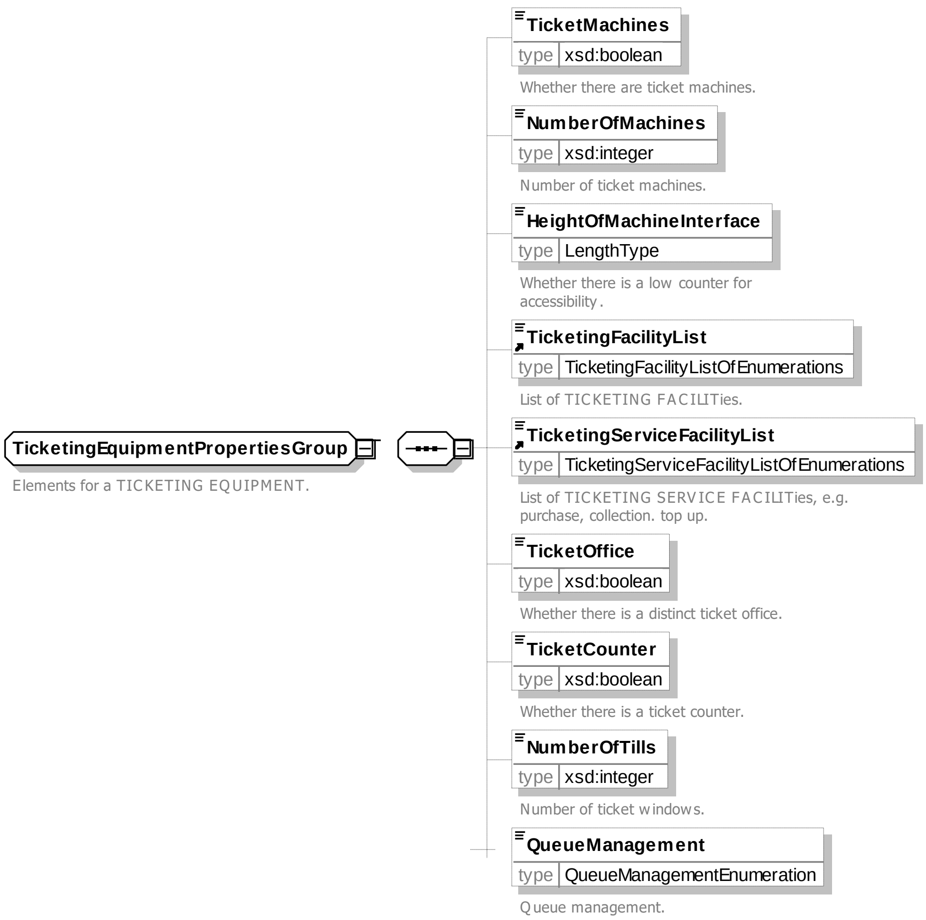


Figure 608 – *TicketingEquipmentPropertiesGroup – XSD*

QueueManagementType – Allowed values

Allowed values for ***QueueManagement*** *(QueueManagementEnuerationm).*

Table 511 – *QueueManagementType* – Allowed Values

Value	Description
maze	*Maze*
separateLines	*Separate lines.*
ticketed	*Queue tickets*
other	*Other queue equipment*

8.4.8.10.3.4 **TicketingEquipmentAccesibilityGroup – XML Group**

Elements characterising accessibility of TICKETING EQUIPMENT.

Table 512 – *TicketingEquipmentAccesibilityGroup – Group*

Classification	Name	Type	Cardinality	Description
	LowCounterAccess	*xsd:boolean*	0:1	Whether there is a low ticket office counter for accessibility.
	HeightOfLow-Counter	*LengthType*	0:1	Height of counter for accessibility.
	InductionLoops	*xsd:boolean*	0:1	Whether there are induction loops.
	TactileInterface-Available	*xsd:boolean*	0:1	Whether there is a tactile interface available. +v1.1
	AudioInterface-Available	*xsd:boolean*	0:1	Whether there is a tactile interface available. +v1.1
	DisabledPriority	*xsd:boolean*	0:1	Whether there are is priority for disabled users +v1.1.
	WheelchairSuitable	*xsd:boolean*	0:1	Whether ticketing Machoine is wheelchair suitable +v1.1.

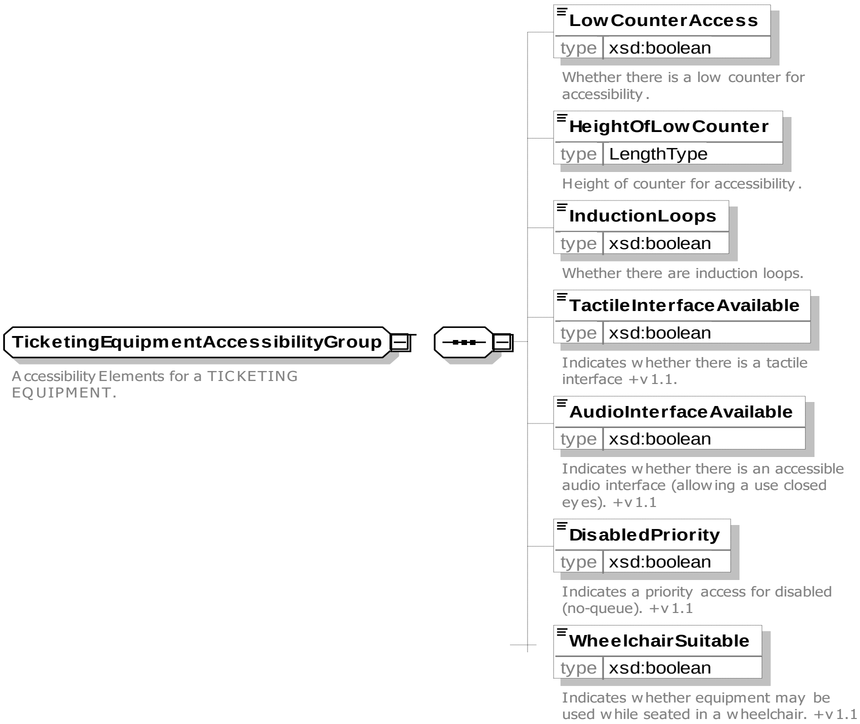


Figure 609 – *TicketingEquipmentAccessibilityGroup* – *XSD*

TicketingFacility – Allowed values

Allowed values for ***TicketingFacility*** (*TicketingFacilityEnumeration)*.

Table 513 – *TicketingFacility* – Allowed Values

Value	Description
ticketMachines	Ticket machines
ticketOffice	Ticket office
ticketOnDemandMachines	Ticket on demand machines
mobileTicketing	Mobile ticketing.
ticketSales	Ticket sales
ticketCollection	Ticket collection
centralReservations	Central reservations can be made
localTickets	Local Ticket sales
nationalTickets	National Ticket sales
internationalTickets	International Ticket sales

TicketType – Allowed values

Allowed values for ***TicketType*** (*TicketTypeEnumeration)*.

Table 514 – *TicketType* – Allowed Values

Value	Description
standard	Standard ticket
promotion	Promotional ticket
concession	Concessionary ticket
group	Group ticket
season	Season ticket

travelCard	Travel card ticket	*other*	Other ticket

TicketScope – Allowed values

Allowed values for ***TicketScope*** (*TicketScopeEnumeration*).

Table 515 – *TicketScope* – Allowed Values

Value	Description
localTicket	Local ticket
nationalTicket	National ticket
internationalTicket	International ticket

8.4.8.11 Local Service

The following figure shows detailed attributes of the LOCAL SERVICE model elements.

8.4.8.11.1 LOCAL SERVICE – Conceptual MODEL

The LOCAL SERVICE model elements provide information on available services at a SITE or on board.

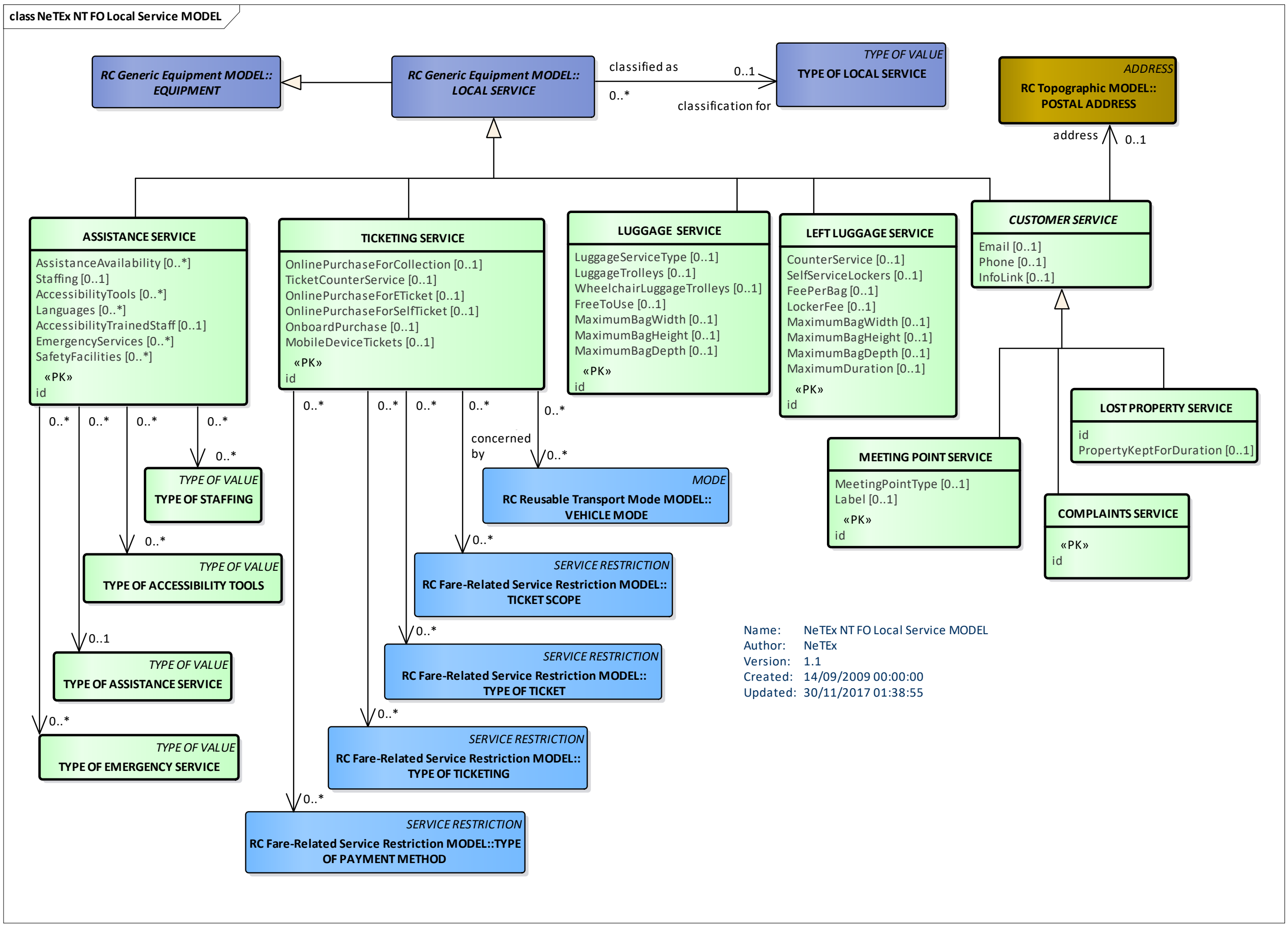


Figure 610 – LOCAL SERVICE– Conceptual MODEL (UML)

8.4.8.11.2 Local Service – Physical Model

The following figure shows detailed attributes of the LOCAL SERVICE model elements.

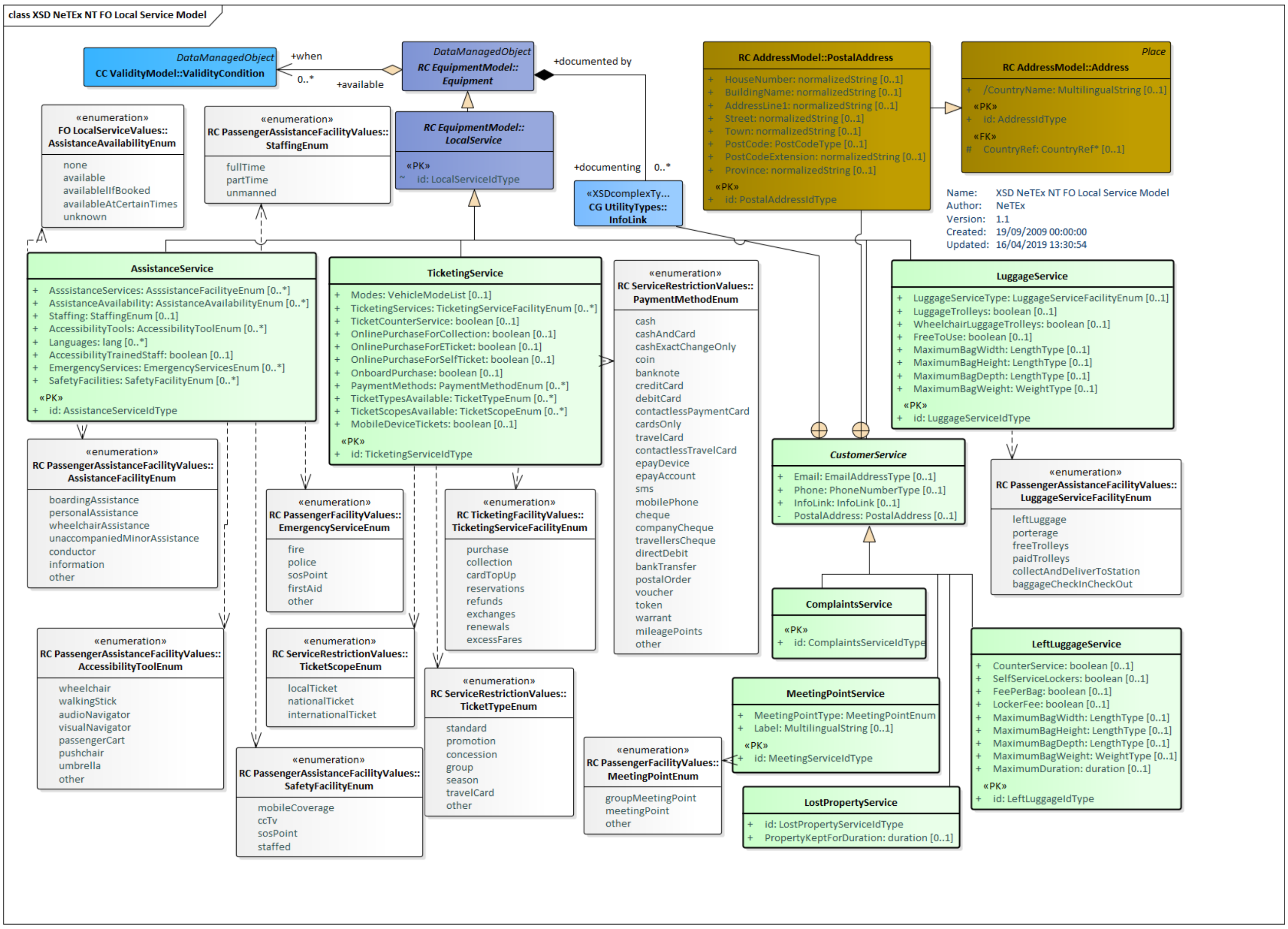


Figure 611 – Local Service– Physical Model (UML)

8.4.8.11.3 Local Service – Attributes and XSD

8.4.8.11.3.1 AssistanceService – Model Element

Specialisation of LOCAL SERVICE for ASSISTANCE providing information like language, accessibility trained staff, etc.

Table 516 – *AssistanceService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LocalService*	::>	ASSISTANCE SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*AssistanceServiceIdType*	1:1	Identifier of ASSISTANCE SERVICE.
«enum»	***AssistanceFacility***	*AssistanceFacilityEnum*	0:*	List of ASSISTANCE FACILITies. See allowed values below.
«enum»	***Assistance-Availability***	*AssistanceAvailability-Enum*	0:*	Availability of assistance service. See allowed values below. See allowed values below.
«enum»	***Staffing***	*StaffingEnum*	0:1	Whether the service is staffed. See allowed values below.

«enum»	***AccessibilityTools***	*AccessibilityToolEnum*	0:*	Whether accessibility tools such as wheelchairs are available. See allowed values below.
	Languages	*xsd:lang*	0:*	Which languages are spoken?
«enum»	***Accessibility-TrainedStaff***	*xsd:boolean*	0:1	Whether staff are accessibility trained.
«enum»	***Emergency-Services***	*EmergencyServicesEnum*	0:*	Emergency services available that may be relevant for accessibility. See allowed values below.
«enum»	***SafetyFacilityList***	*SafetyFacilityEnum*	0:*	Safety facilities available.

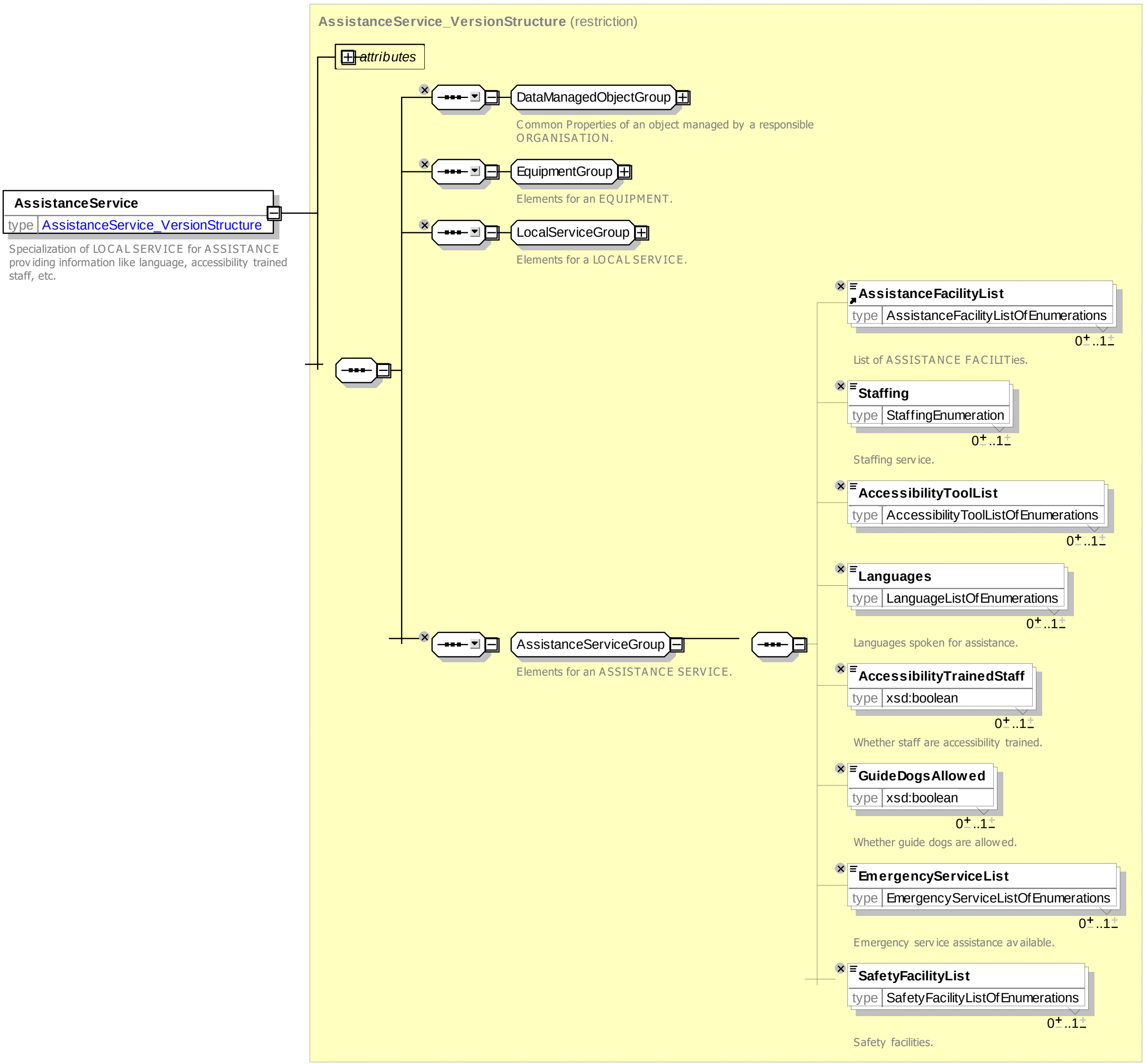


Figure 612 – *AssistanceService – XSD*

AssistanceService – Allowed values

Allowed values for ***AssistanceService*** *(AssistanceServiceEnumeration).*

Table 517 – *AssistanceService* – Allowed Values

Value	Description
boardingAssistance	Boarding assistance
personalAssistance	Personal assistance
wheelchairAssistance	Wheelchair assistance
unaccompaniedMinorAssistance	Assistance for unaccompanied minors
conductor	There is a conductor
information	Information facility is available
other	Other information

Staffing – Allowed values

Allowed values for ***Staffing*** (*StaffingEnumeration)*.

Table 518 – *Staffing* – Allowed Values

Value	Description
fullTime	Full Time staffing
partTime	Part Time staffing
unmanned	Unmanned

AccessibilityTool – Allowed values

Allowed values for ***AccessibilityTool*** (*AccessibilityToolEnum*).

Table 519 – *AccessibilityTool* – Allowed Values

Value	Description
wheelchair	Wheelchairs available or passenger use
walkingStick	Walking sticks available or passenger use
audioNavigator	Audio navigators available for passenger use
visualNavigator	Audio navigation devices available
other	Other devices available

EmergencyServiceType – Allowed values

Allowed values for ***EmergencyServiceType*** (*EmergencyServiceEnum*).

Table 520 – *EmergencyServiceType* – Allowed Values

Value	Description
fire	Fire brigade emergency services
police	Police services
sosPoint	SOS Point
firstAid	First aid services
other	Other emergency service

SafetyFacility – Allowed values

Allowed values for ***SafetyFacility*** (SafetyFacilityEnum).

Table 521 – *SafetyFacility* – Allowed Values

Value	Description
mobileCoverage	There is mobile phone coverage
ccTv	There is close circuit television camera surveillance
sosPoint	There is an SOS point
staffed	There are staff

8.4.8.11.3.2 LuggageService – Model Element

Specialisation of CUSTOMER SERVICE for LUGGAGE SERVICE (provides luggage service attributes like luggage trolley, free to use, etc.).

Table 522 – *LuggageService* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*LocalService*	*::>*	LUGGAGE SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*LuggageServiceIdType*	1:1	Identifier of LUGGAGE SERVICE.
«enum»	***LuggageService-FacilityList***	*LuggageServiceFacilityEnum*	0:*	Type of luggage services available. See allowed values below.
	LuggageTrolleys	*xsd:boolean*	0:1	Whether luggage trolleys are available.
	Wheelchair-LuggageTrolleys	*xsd:boolean*	0:1	Whether there are luggage trolleys for wheelchair users.
	FreeToUse	*xsd:boolean*	0:1	Whether luggage trolleys are free to use.
	MaximumBag-Width	*LengthType*	0:1	Maxiimum width of luggage accepted by service.
	MaximumBag-Height	*LengthType*	0:1	Maxiimum height of luggage accepted by service.
	MaximumBag-Depth	*LengthType*	0:1	Maxiimum depth of luggage accepted by service.
	MaximumBag-Weight	*LengthType*	0:1	Maxiimum weight of luggage accepted by service. +v1.1

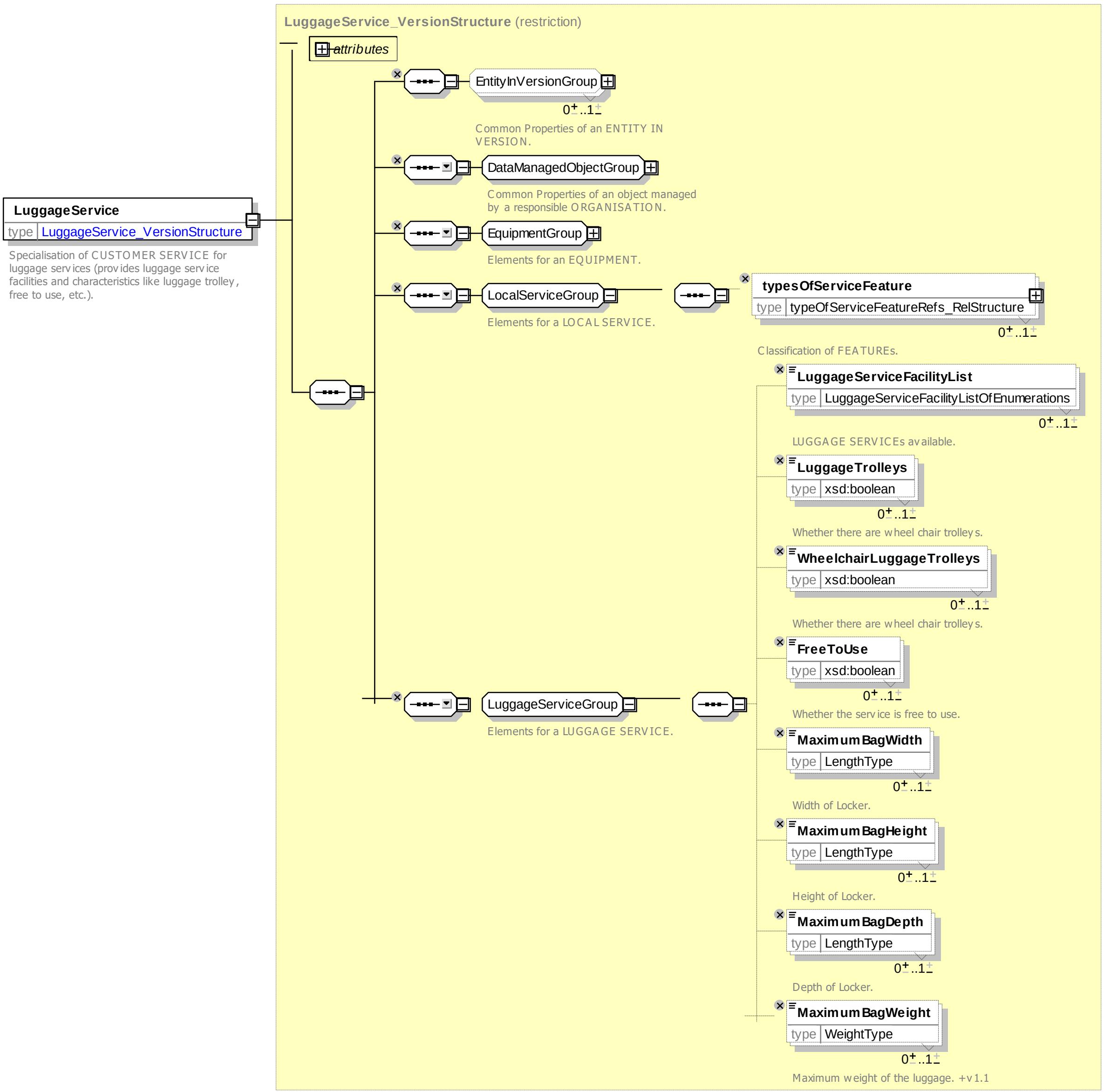


Figure 613 – *LuggageService – XSD*

LuggageServiceFacility – Allowed values

Allowed values for ***LuggageServiceFacility*** (*LuggageServiceFacilityEnumeration*).

Table 523 – *LuggageServiceFacility* – Allowed Values

Value	Description
leftLuggage	Left luggage
porterage	Porterage
freeTrolleys	Free luggage trolleys
paidTrolleys	Luggage trolleys require payment to use
collectAndDeliverToStation	Luggage Collection and delivery service

8.4.8.11.3.3 LeftLuggageService – Model Element

Specialisation of CUSTOMER SERVICE for LEFT LUGGAGE SERVICE (provides left luggage attributes like self-service locker, locker free, etc.).

Table 524 – *LeftLuggageService* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*LocalService*	*::>*	LEFT LUGGAGE SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*LeftLuggageIdType*	1:1	Identifier of LEFT LUGGAGE SERVICE.
	CounterService	*xsd:boolean*	0:1	Whether left luggage is a counter service.
	SelfServiceLockers	*xsd:boolean*	0:1	Whether there are self-service lockers for left luggage.
	FeePerBag	*xsd:boolean*	0:1	Whether there is a fee per bag.
	LockerFee	*xsd:boolean*	0:1	Whether there is a locker fee.
	MaximumBagWidth	*LengthType*	0:1	Maximum width of luggage allowed.
	MaximumBagHeight	*LengthType*	0:1	Maximum height of luggage allowed.
	MaximumBagDepth	*LengthType*	0:1	Maximum depth of luggage allowed.
	MaximumBagWeight	*LengthType*	0:1	Maxiimum weight of luggage allowed. +v1.1
	MaximumDuration	*xsd:duration*	0:1	Maximum time for which luggage can be left. +v1.1

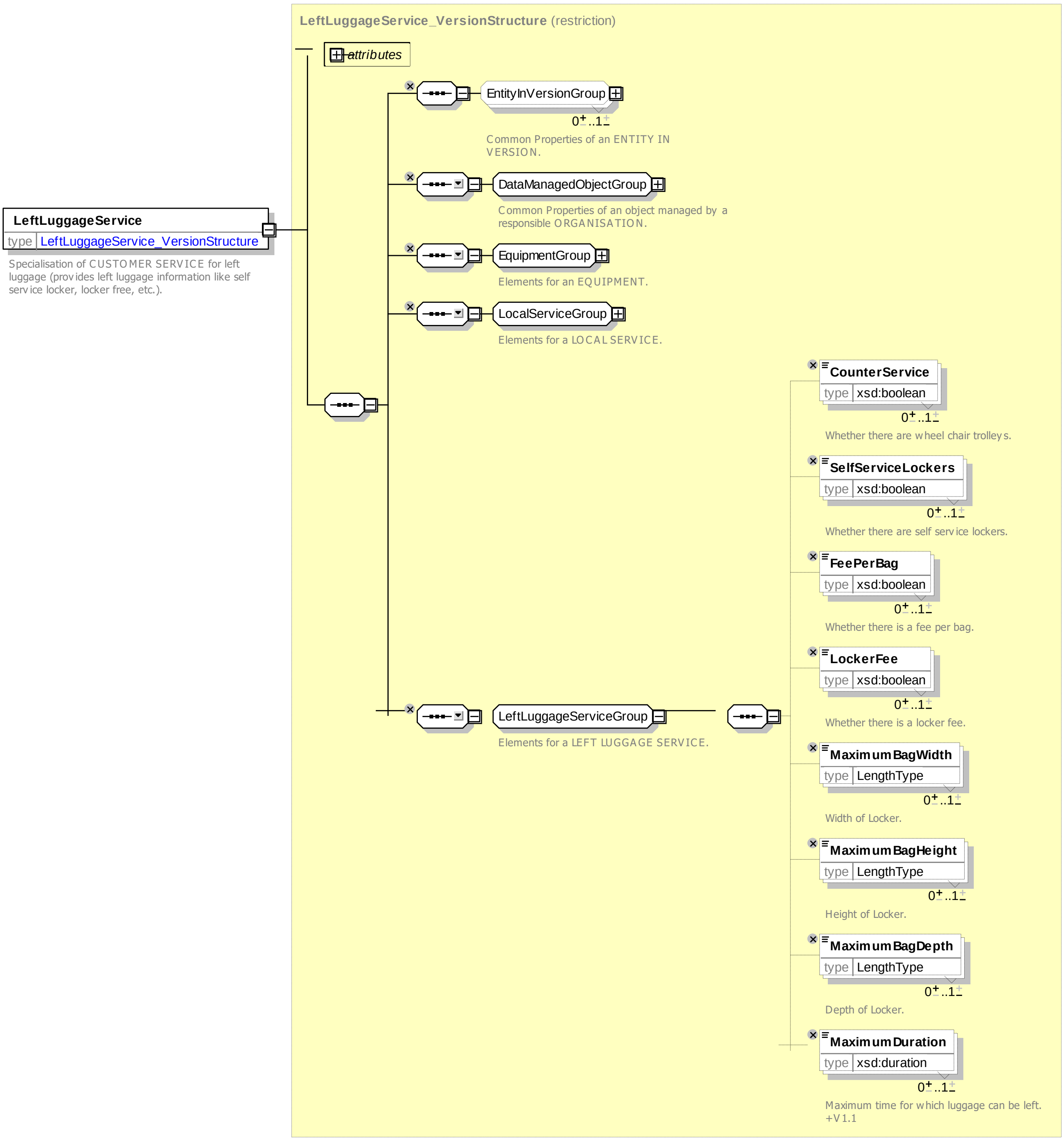


Figure 614 – *LeftLuggageService – XSD*

8.4.8.11.3.4 CustomerService – Model Element

Generic specialization of LOCAL SERVICE for CUSTOMER SERVICEs (lost properties, meeting point, complaints, etc.).

Table 525 – *CustomerService* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*LocalService*	::>	CUSTOMER SERVICE inherits from LOCAL SERVICE.
	Email	*EmailAddressType*	0:1	Email for CUSTOMER SERVICE.
	Phone	*PhoneNumberType*	0:1	Phone for CUSTOMER SERVICE.
	InfoLink	*InfoLink*	0:1	URL for CUSTOMER SERVICE.
	PostalAddress	*PostalAddress*	0:1	Postal Address for CUSTOMER SERVICE.

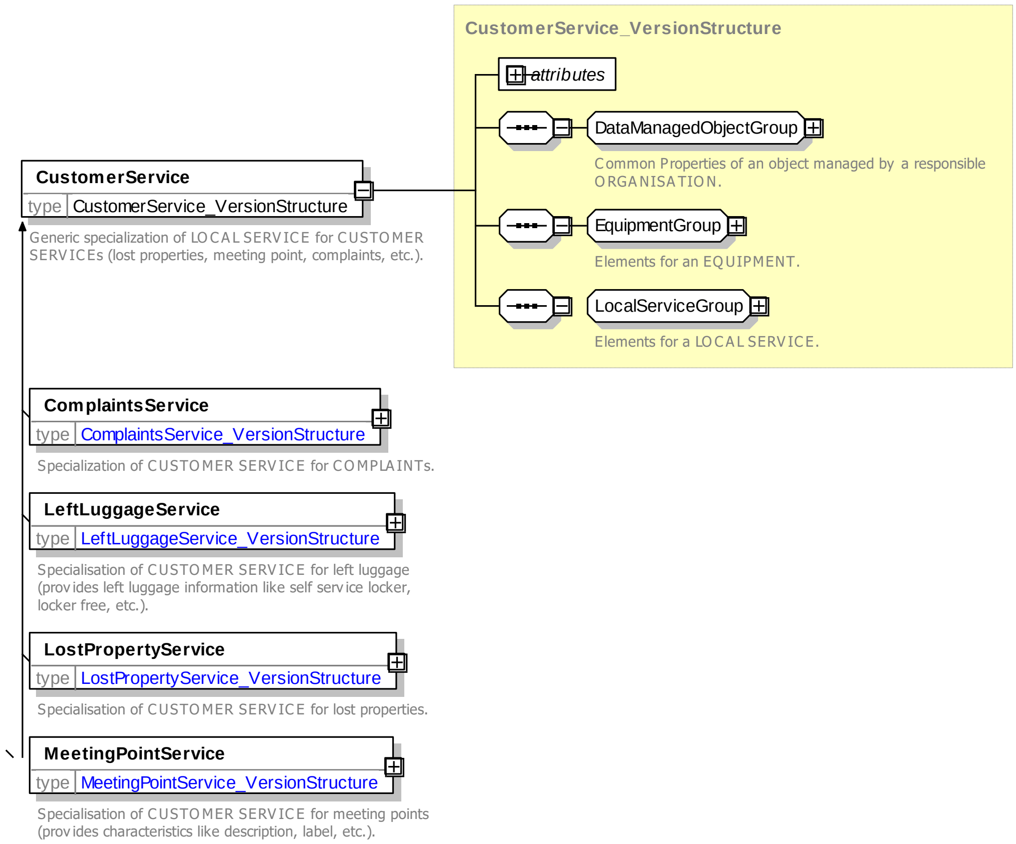


Figure 615 – *CustomerService – XSD*

8.4.8.11.3.5 ComplaintsService – Model Element

Specialisation of CUSTOMER SERVICE for COMPLAINTs.

Table 526 – *ComplaintsService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*CustomerService*	::>	COMPLAINTS SERVICE inherits from CUSTOMER SERVICE.
«PK»	***id***	*ComplaintsService-IdType*	1:1	Identifier of COMPLAINTS SERVICE.

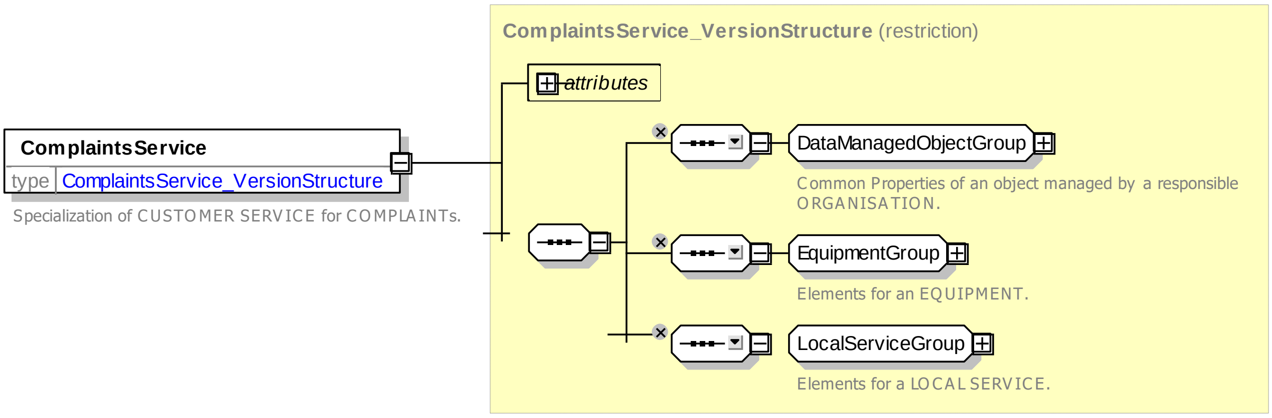


Figure 616 – *ComplaintsService – XSD*

8.4.8.11.3.6 LostPropertyService – Model Element

Specialisation of CUSTOMER SERVICE for LOST PROPERTies.

Table 527 – *LostPropertyService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LocalService*	::>	LOST PROPERTY SERVICE inherits from CUSTOMER SERVICE.
«PK»	***id***	*LostPropertyServiceIdType*	1:1	Identifier of LOST PROPERTY SERVICE.
	PropertyKept-ForDuration	*xsd:duration*	0:1	Period for which lost property is kept - after this time it may be disposed of. +V1.1

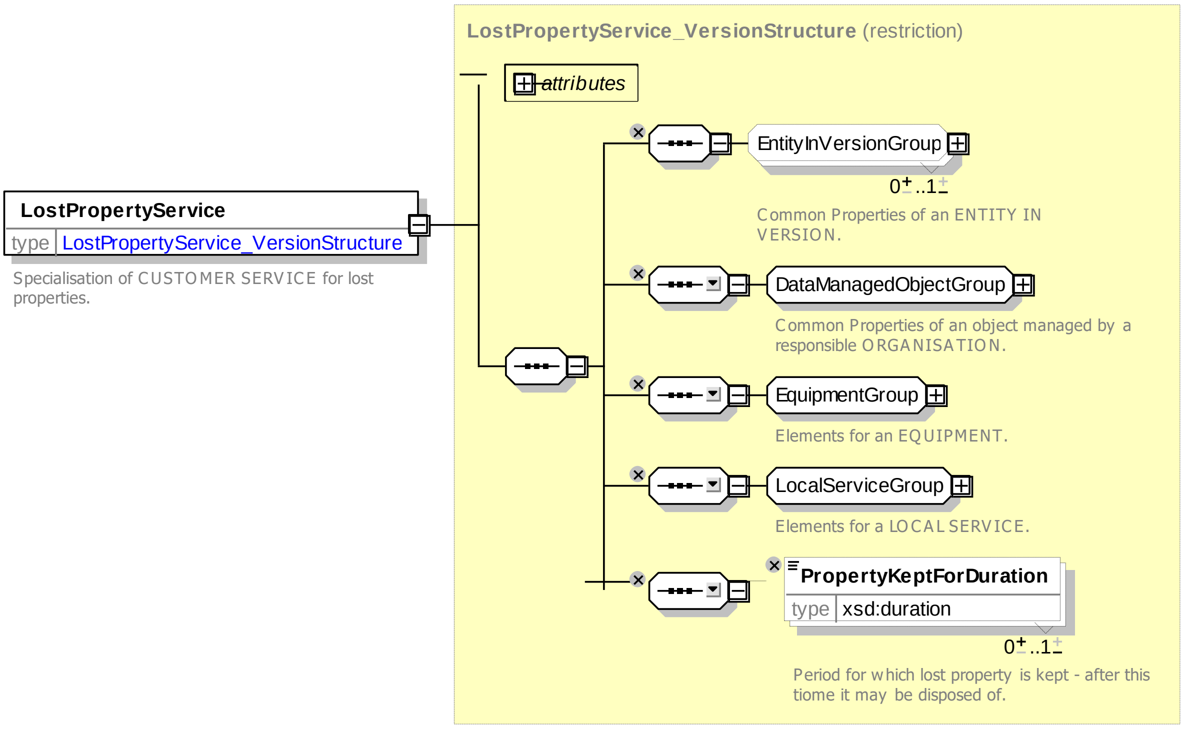


Figure 617 – *LostPropertyService – XSD*

8.4.8.11.3.7 MeetingPointService – Model Element

Specialisation of CUSTOMER SERVICE for MEETING POINTs (provides attributes like description, label, etc.).

Table 528 – *MeetingPoint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LocalService*	::>	MEETING POINT SERVICE inherits from CUSTOMER SERVICE.
«PK»	***id***	*MeetingServiceIdType*	1:1	Identifier of MEETING POINT SERVICE.
«enum»	***MeetingPointType***	*MeetingPointEnum*	1:1	Type of meeting point. See allowed values below.
	Label	*MultilingualString*	0:1	Label of meeting point.

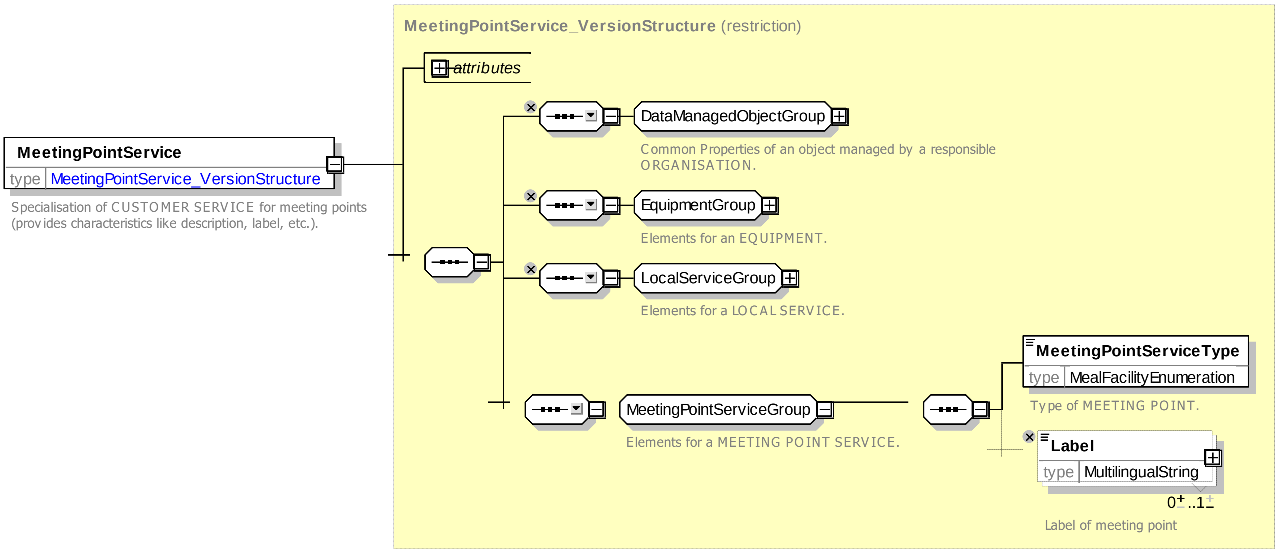


Figure 618 – *MeetingPointService – XSD*

MeetingPointType – Allowed values

Allowed values for ***MeetingPointType*** (*MeetingPointEnumeration*).

Table 529 – *MeetingPointType* – Allowed Values

Value	Description
groupMeetingPoint	Group meeting point
meetingPoint	Meeting point

8.4.8.11.3.8 TicketingService – Model Element

Specialisation of LOCAL SERVICE for TICKETING, providing ticket counter and online purchase information, also associated with PAYMENT METHOD and TYPE OF TICKET.

Table 530 – *TicketingService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*LocalService*</u>	::>	TICKETING SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*TicketingServiceIdType*	1:1	Identifier of TICKETING SERVICE.
«enum»	***Modes***	*VehicleModeList*	0:1	TRANSPORT MODEs for which TICKETING SERVICE applies. See TRANSPORT MODE for allowed values.
«enum»	***TicketingServices***	*TicketingServiceFacility-Enum*	0:*	Type of ticketing service provided. See allowed values below.
	TicketCounter-Service	*xsd:boolean*	0:1	Whether there is an over the counter sale of tickets.

	OnlinePurchaseForCollection	*xsd:boolean*	0:1	Whether there is an on-line sale of tickets for collection in the station.
	OnlinePurchaseForETicket	*xsd:boolean*	0:1	Whether there is an on-line sale of tickets for etickets.
	OnlinePurchaseForSelfTicket	*xsd:boolean*	0:1	Whether there is an on-line sale of tickets for self-print tickets.
	OnboardPurchase	*xsd:boolean*	0:1	Whether there is an on-board purchase of tickets.
«enum»	***PaymentMethods***	*PaymentMethodEnum*	0:*	Method of payment for ticket. See Service Restrictions model for allowed values.
«enum»	***TicketTypes-Available***	*TicketTypeEnum*	0:*	Types of ticket available. See allowed values below.
«enum»	***TicketScopes-Available***	*TicketScopeEnum*	0:*	Scope of tickets available. See allowed values below.
	MobileDevice-Tickets	*xsd:boolean*	0:1	Whether there is a purchase of tickets from a mobile device.

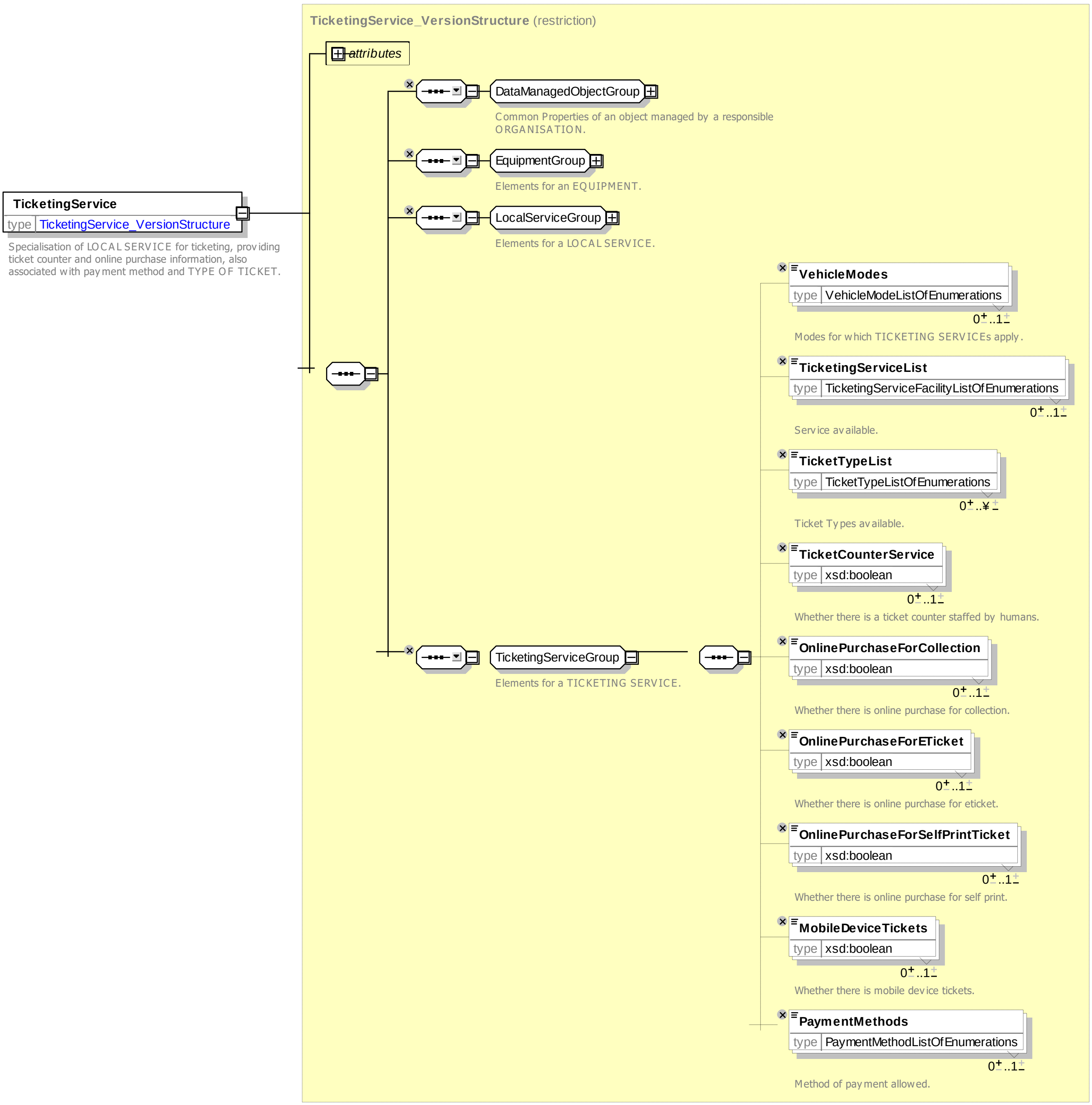


Figure 619 – *TicketingService – XSD*

TicketingFacility – Allowed values

Allowed values for ***TicketingFacility*** (*TicketingFacilityEnumeration)*.

Table 531 – TicketingFacility – Allowed Values

Value	Description
ticketMachines	Ticket machines.
ticketOffice	Ticket office.
ticketOnDemand-Machines	Ticket on demand machines.
mobileTicketing	Mobile ticketing.
ticketSales	Ticket sales.

ticketCollection	Ticket collection.
centralReservations	Central reservations can be made.
localTickets	Local Ticket sales.
nationalTickets	National Ticket sales.
internationalTickets	International Ticket sales.

TicketingServiceFacility – Allowed values

Allowed values for ***TicketingService*** (*TicketingServiceFacilityEnumeration*).

Table 532 – *TicketingServiceFacility* – Allowed Values

Value	Description
purchase	Purchase.
collection	Collection.
cardTopUp	Card Top up.
reservations	Reservations.

8.4.8.12 Assistance Booking Service

8.4.8.12.1 ASSISTANCE BOOKING SERVICE – Conceptual MODEL

The ASSISTANCE BOOKING SERVICE provides information about how to book assistance for wheelchair and disabled users.

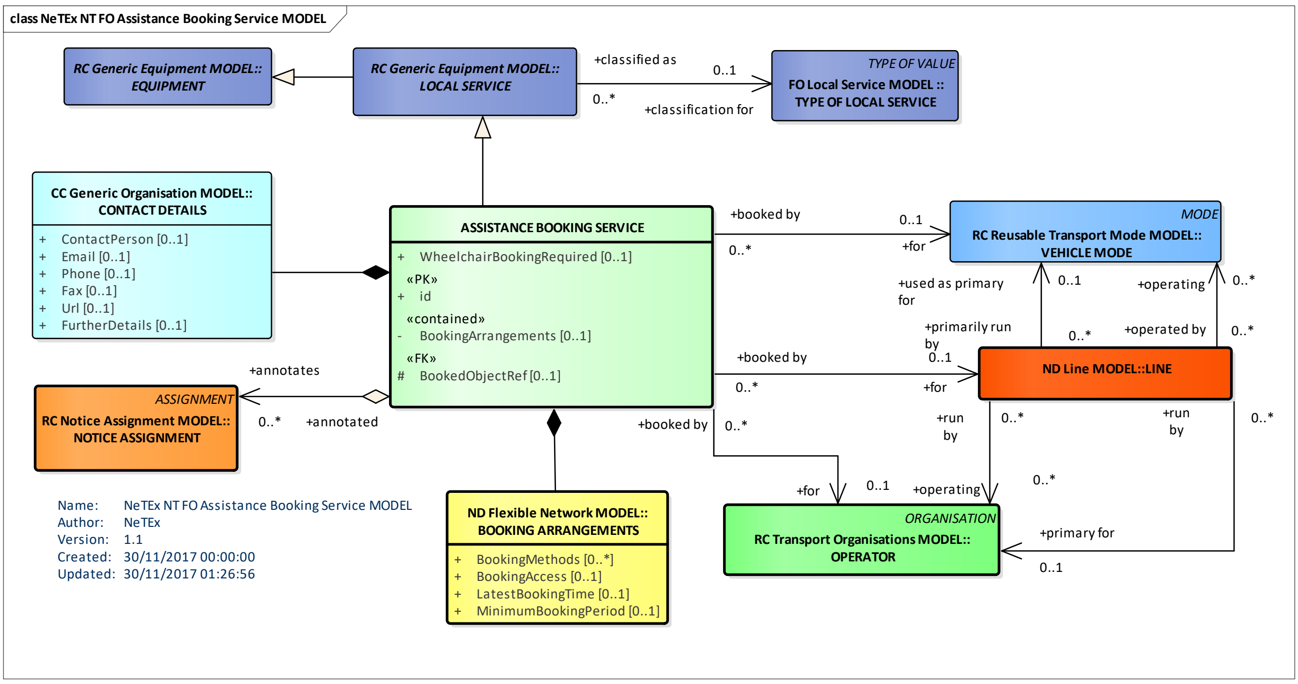


Figure 620 – ASSISTANCE BOOKING SERVICE– Conceptual MODEL (UML)

8.4.8.12.2 Assistance Booking Service – Physical Model

The following figure shows detailed attributes of the ASSISTANCE BOOKING SERVICE model elements.

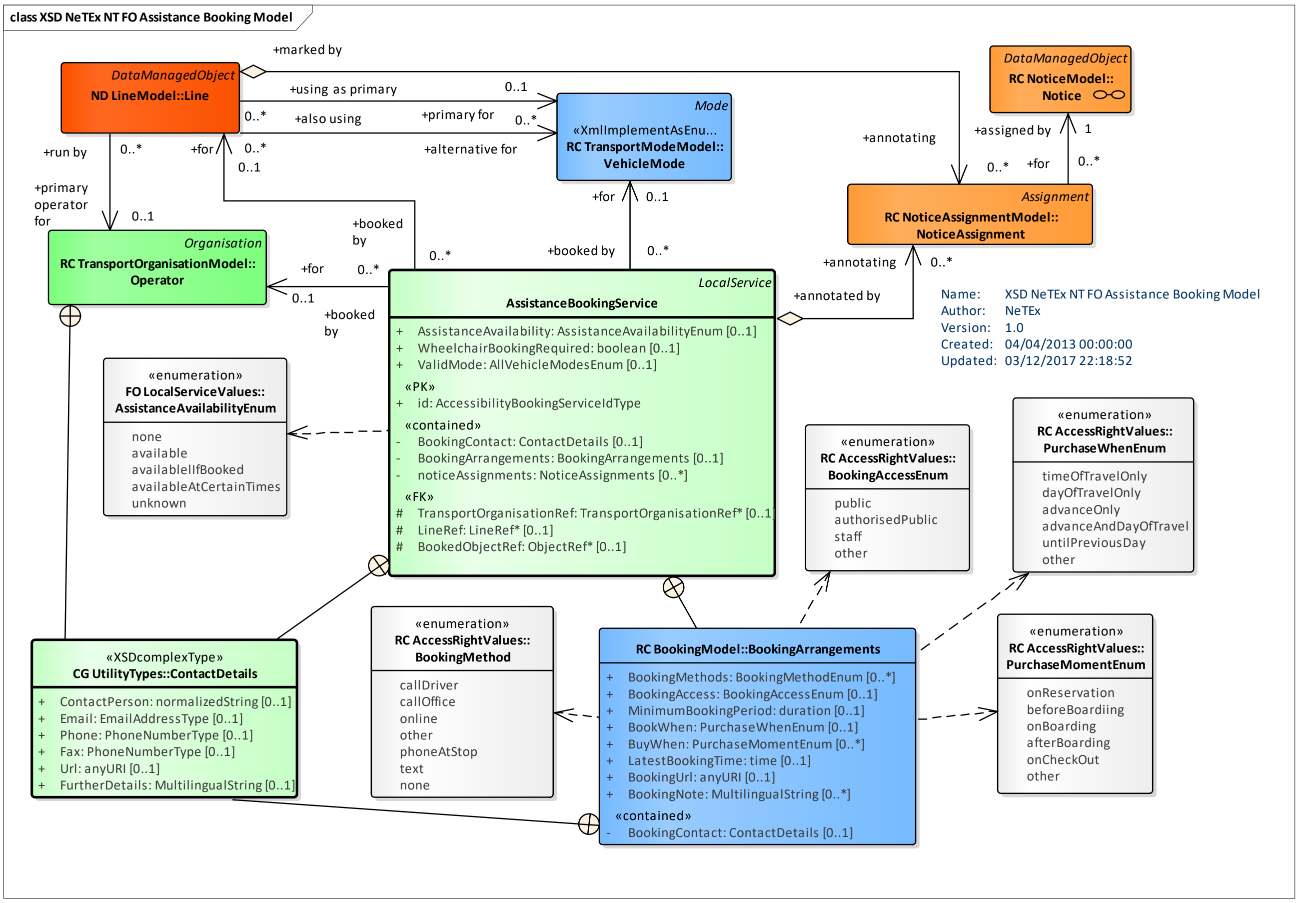


Figure 621 – Assistance Booking Service– Physical Model (UML)

8.4.8.12.3 Assistance Booking Service – Attributes and XSD

8.4.8.12.3.1 AssistanceBookingService – Model Element

A specialisation of LOCAL SERVICE for ASSISTANCE BOOKING providing information like language, accessibility trained staff, etc.

Table 533 – *LocalService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LocalService*	::>	ASSISTANCE BOOKING SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*AssistanceBookingService-IdType*	1:1	Identifier of ASSISTANCE BOOKING SERVICE.
«enum»	***Assistance-Availability***	*AssistanceAvailabilityEnum*	0:1	When assistance is available. See allowed values below.
	Wheelchair-BookingRequired	*boolean*	0:1	Whether booking is required for wheelchairs.
«cntd»	***BookingContact***	*ContactDetails*	0:1	Contact details for ASSISTANCE BOOKING.

«cntd	***Booking-Arrangements***	*<u>BookingArrangements</u>*	0:1	Booking arrangements for users with accessibility requirements.
«enum»	***ValidMode***	*AllModesEnumeration*	0:1	Mode for which valid. See TRANSPORT MODE for allowed values.
«FK»	***Transport-OrganisationRef***	*(TransportOrganisationRef) OperatorRef | AuthorityRef*	0:1	OPERATOR or AUTHORITY to whom arrangements apply.
«FK»	***LineRef***	*LineRef+*	0:1	LINE to which arrangements apply.
«FK»	***BookedObjectRef***	*(VersionOfObjectRef)*	0:1	Specific object to which booking relates, e.g. SCHEDULED STOP POINT, STOP, VEHICLE JOURNEY, etc.
«cntd»	***notice-Assignments***	*<u>NoticeAssignment</u>*	0:*	Additional notes about ASSISTANCE BOOKING SERVICE.

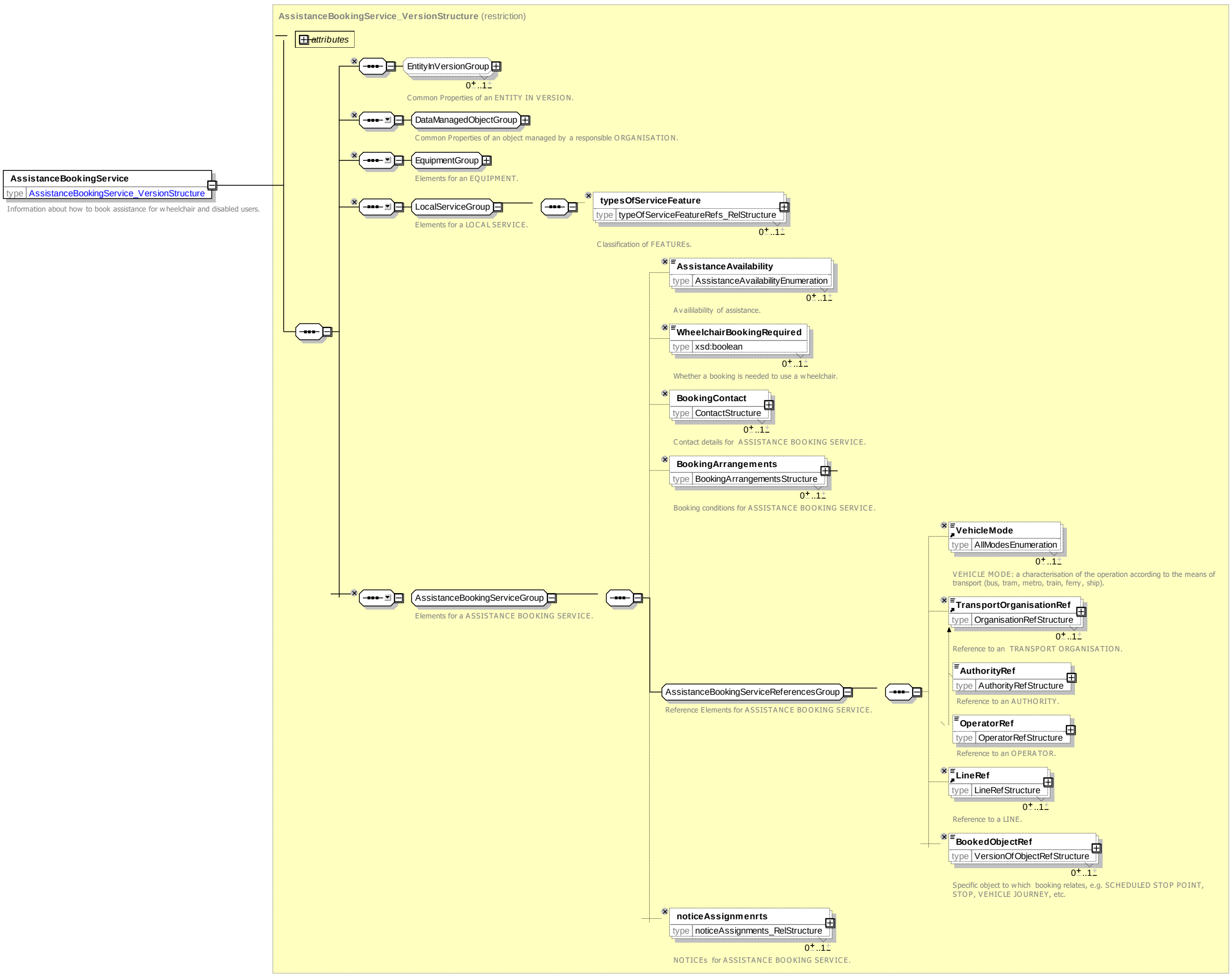


Figure 622 – *AssistanceBookingService – XSD*

AssistanceAvailability – Allowed values

Allowed values for ***AssistanceAvailability*** type *(AssistanceAvailability Enumeration).*

Table 534 – *AssistanceAvailability* – Allowed Values

Value	Description
none	No assistance available.
available	Assistance available.
availableIfBooked	Available if booked.
availableAtCertainTimes	Assistance available at certain times.
unknown	Availability unknown.

8.4.8.13 Commercial Service

8.4.8.13.1 COMMERCIAL SERVICE – Conceptual MODEL

The following figure shows the COMMERCIAL SERVICE model elements.

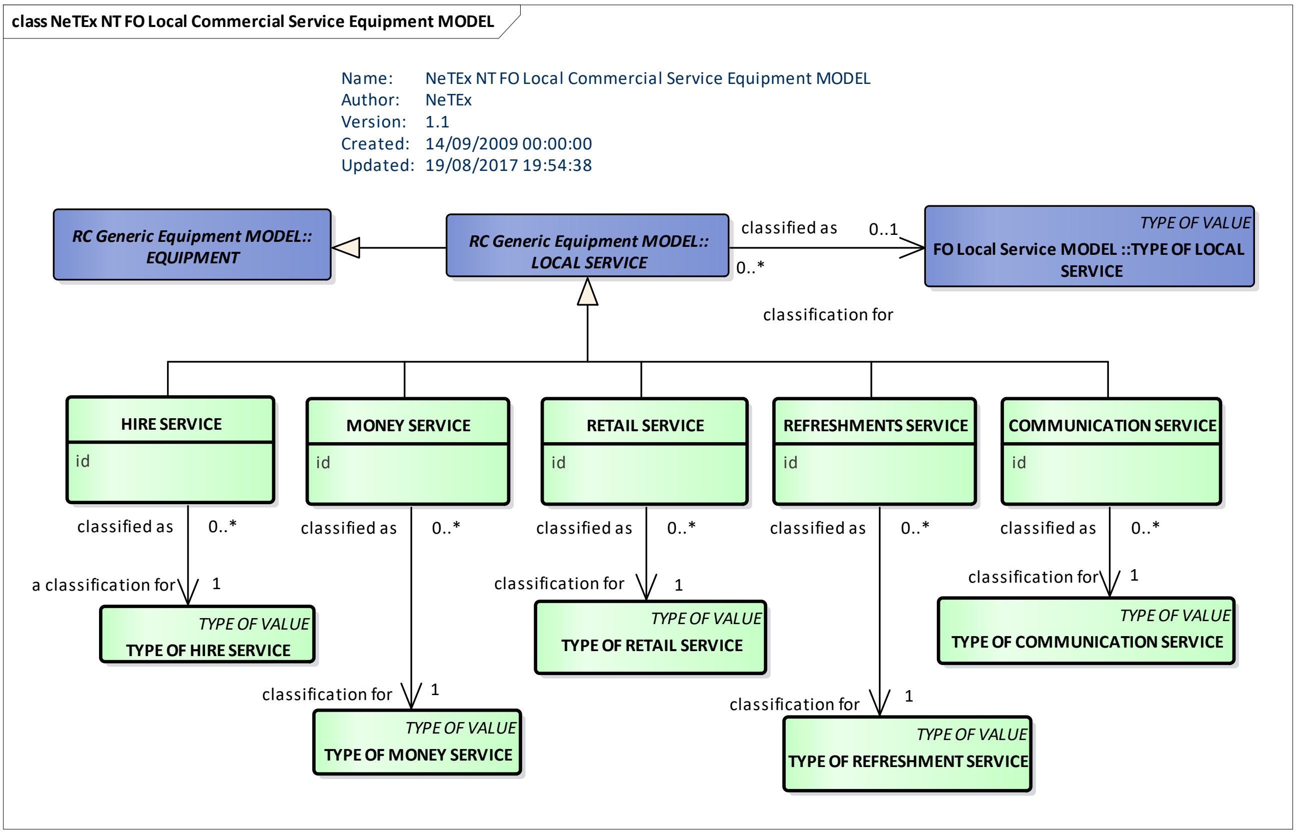


Figure 623 – COMMERCIAL SERVICE – Conceptual MODEL (UML)

8.4.8.13.2 Commercial Service – Physical Model

The following figure shows detailed attributes of the COMMERCIAL SERVICE model elements.

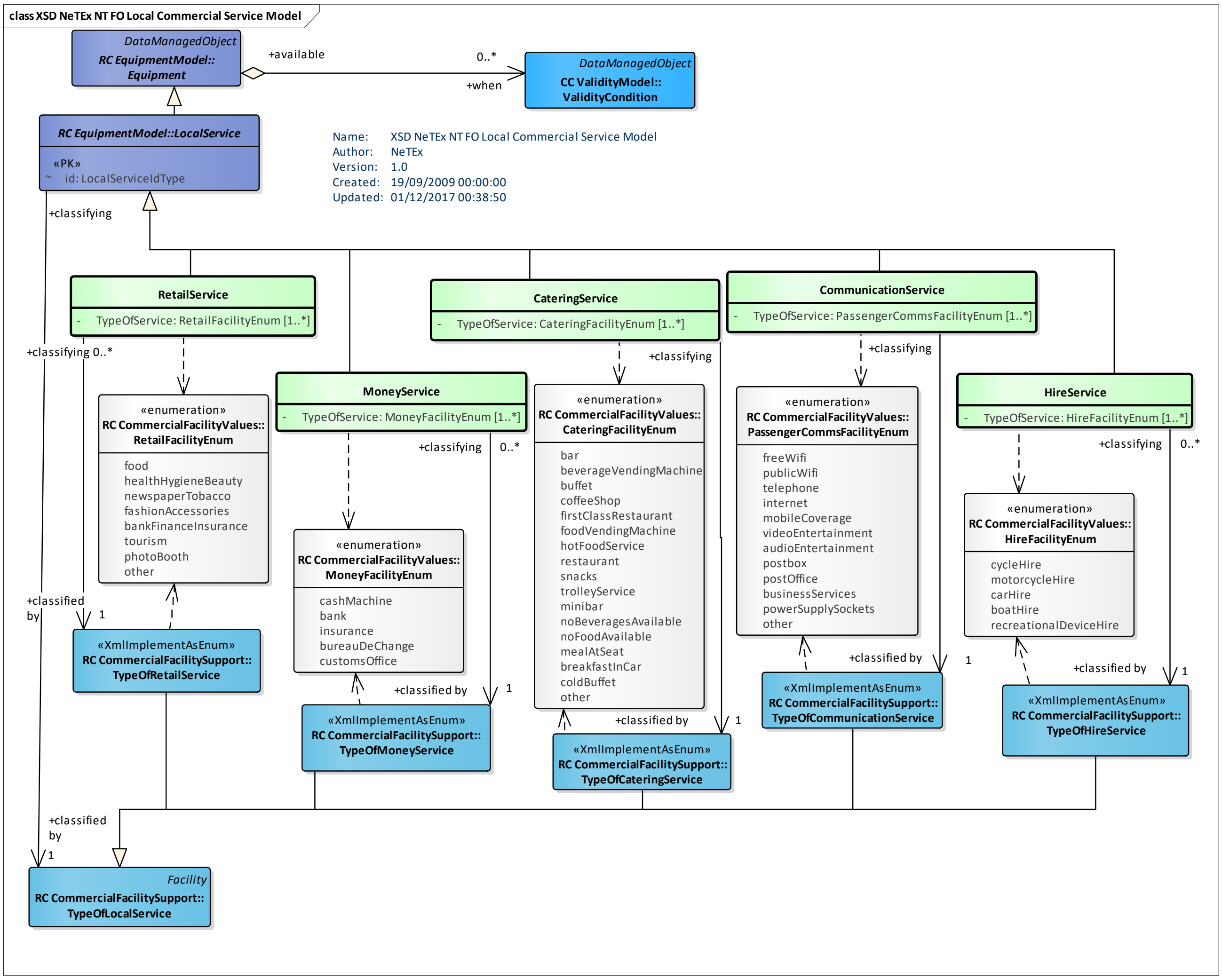


Figure 624 – Commercial Local Service – Physical Model (UML) – Detail

8.4.8.13.3 Commercial Service – Attributes and XSD

8.4.8.13.3.1 RetailService – Model Element

Specialisation of LOCAL SERVICE for RETAIL SERVICEs, classified by TYPE OF RETAIL SERVICE.

Table 535 – *RetailService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*LocalService*</u>	::>	RETAIL SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*RetailService*IdType	1:1	Identifier of RETAIL SERVICE.
«enum»	***TypeOfService***	*RetailFacilityEnum*	1:*	Classifications of RETAIL SERVICE. See allowed values below.

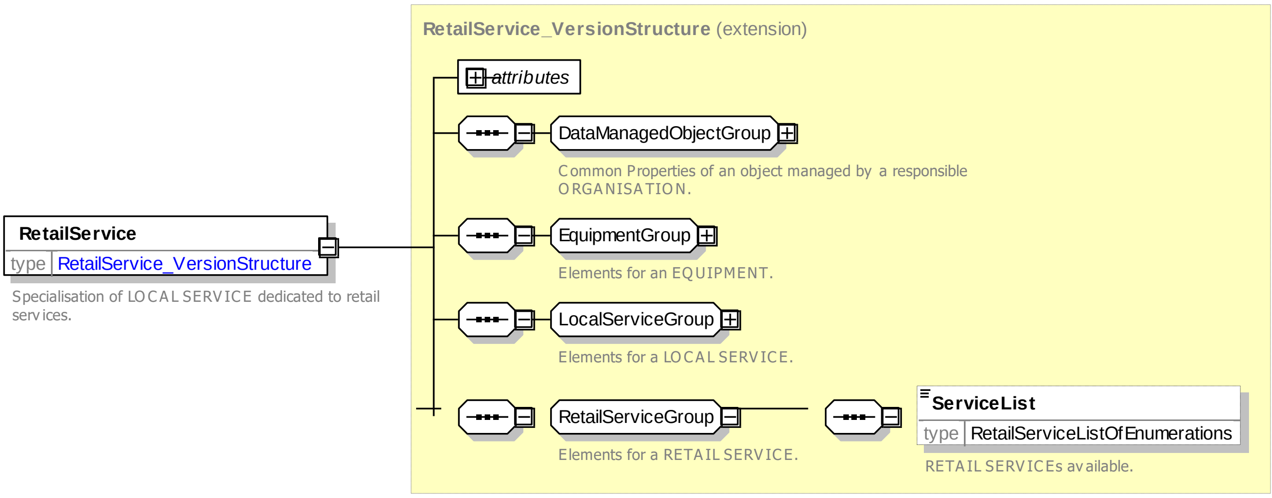


Figure 625 – *RetailService – XSD*

RetailFacility – Allowed values

Allowed values for Retail FACILITY (*RetailFacilityEnumeration*).

Table 536 – *RetailFacility* – Allowed Values

Value	Description
food	Food retail outlets
healthHygieneBeauty	Health and hygiene retail outlets
newspaperTobacco	Newspaper and tobacco retail outlets
fashionAccessories	Fashion and accessory retail outlets
bankFinanceInsurance	Banks
tourism	Tourist shops
photoBooth	Photo booth
other	Other retail outlets

8.4.8.13.3.2 CommunicationService – Model Element

Specialisation of LOCAL SERVICE for COMMUNICATION classified by a TYPE OF COMMUNICATION SERVICE.

Table 537 – *CommunicationService* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*LocalService*	*::>*	COMMUNICATION SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*CommunicationService-IdType*	1:1	Identifier of COMMUNICATION SERVICE.
«enum»	***TypeOfService***	*PassengerCommsFacility-Enum*	1:*	Classifications of COMMUNICATION SERVICE. See allowed values below.

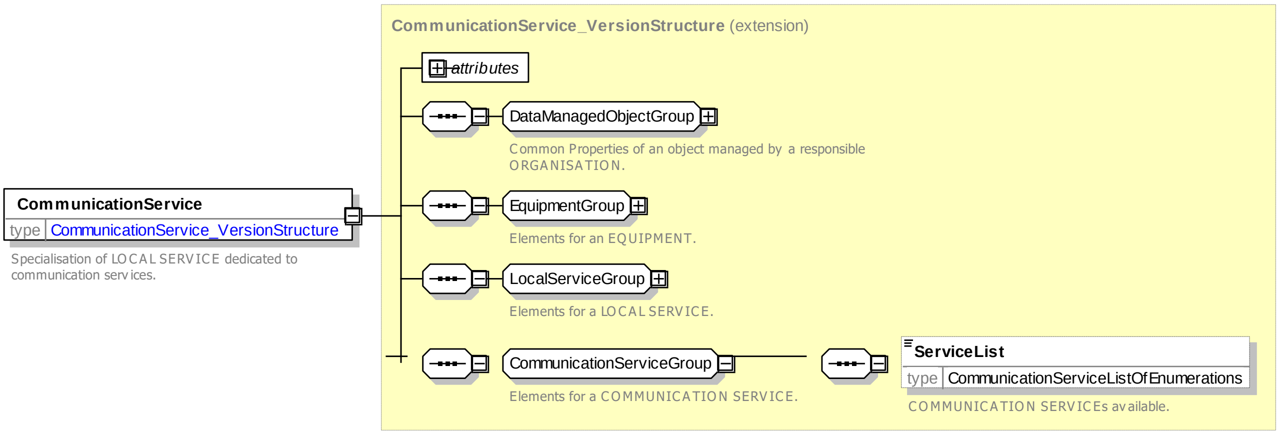


Figure 626 – *CommunicationService – XSD*

PassengerCommsFacilityType – Allowed values

Allowed values for PASSENGER COMMS Facility (*PassengerCommsFacilityEnumeration)*.

Table 538 – *PassengerCommsFacility* – Allowed Values

Value	Description
freeWifi	Free Wifi
publicWifi	Public Wifi, may be charged
telephone	Telephone
internet	Internet access
mobileCoverage	Mobile phone coverage
videoEntertainment	Video entertainment
audioEntertainment	Audio entertainment
postbox	Post box
postOffice	Post office
businessServices	Business service
powerSupplySockets	Power supply sockets

8.4.8.13.3.3 HireService – Model Element

Specialisation of LOCAL SERVICE for HIRE SERVICE classified a TYPE OF HIRE SERVICE.

Table 539 – *HireService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*LocalService*</u>	::>	HIRE SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*HireServiceIdType*	1:1	Identifier of HIRE SERVICE.
«enum»	***TypeOfService***	*HireFacilityEnum*	1:*	Classifications of HIRE SERVICE. See allowed values below.

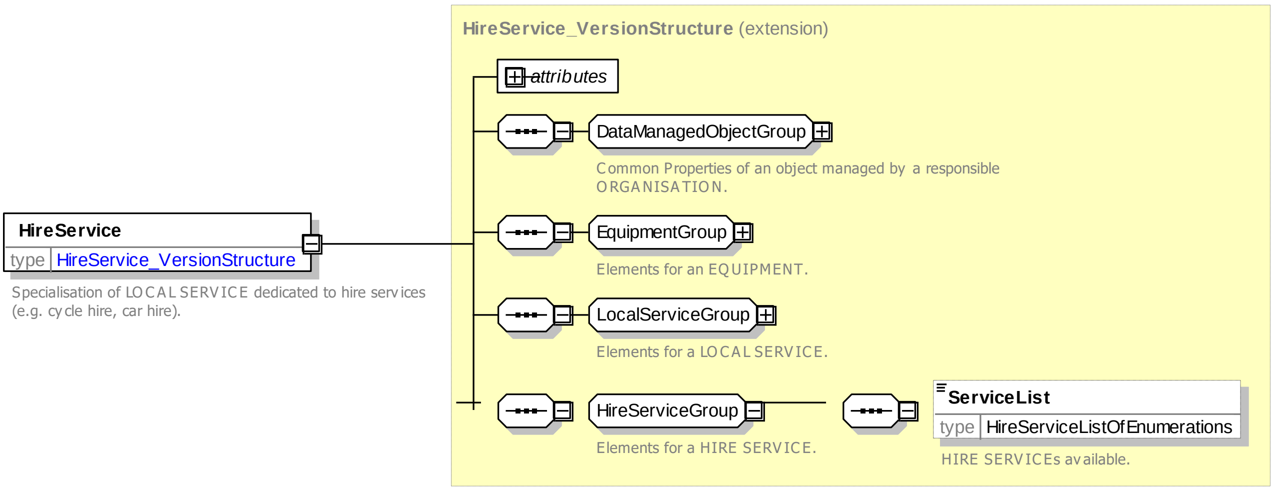


Figure 627 – *HireService – XSD*

HireFacilityType – Allowed values

Allowed values for Hire FACILITY (*HireFacilityEnumeartion*).

Table 540 – HireFacilityType - Allowed Values

Value	Description
cycleHire	Cycle hire
motorcycleHire	Motorcycle hire
carHire	Car hire
boatHire	Boat hire
recreationalDeviceHire	Recreational device hire

8.4.8.13.3.4 *MoneyService* – Model Element

Specialisation of LOCAL SERVICE for MONEY SERVICE classified a TYPE OF MONEY SERVICE.

Table 541 – *MoneyService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LocalService*	::>	MONEY SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*MoneyServiceIdType*	1:1	Identifier of MONEY SERVICE.
«enum»	***TypeOfService***	*MoneyFacilityEnum*	1:*	Classifications of MONEY SERVICE. See allowed values below.

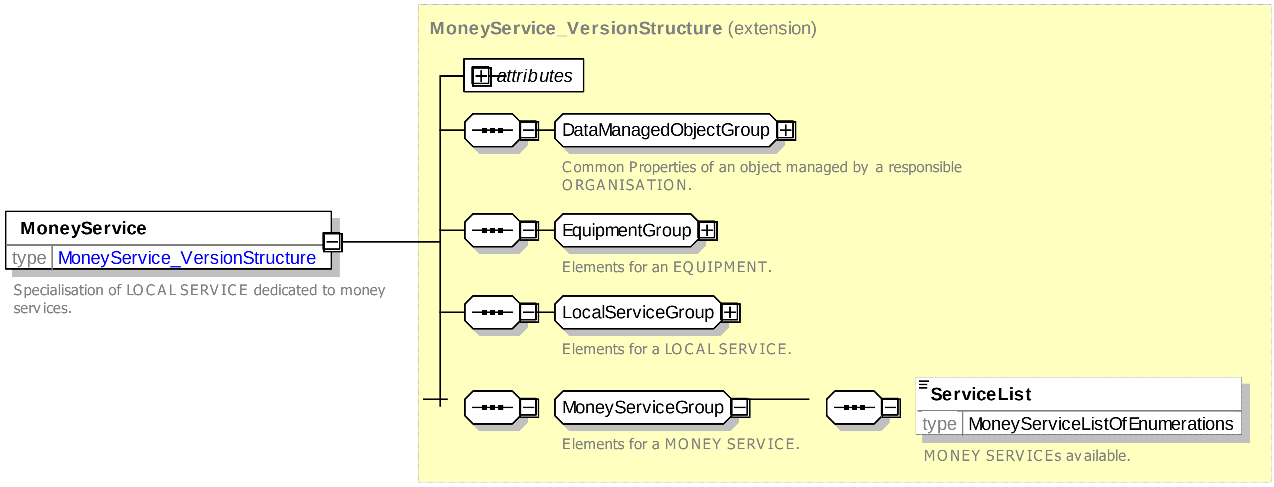


Figure 628 – *MoneyService – XSD*

MoneyFacility – Allowed values

Allowed values for Money FACILITY (*MoneyFacilityEnumeration*).

Table 542 – *MoneyFacility* – Allowed Values

Value	Description
cashMachine	Cash machine
bank	Bank
insurance	Insurance
bureauDeChange	Bureau de change

8.4.8.13.3.5 **CateringService – Model Element**

Specialisation of LOCAL SERVICE for CATERING SERVICE classified as a TYPE OF CATERING SERVICE.

Table 543 – *CateringService* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*LocalService*</u>	::>	CATERING SERVICE inherits from LOCAL SERVICE.
«PK»	***id***	*CateringServiceIdType*	1:1	Identifier of CATERING SERVICE.
«enum»	***TypeOfService***	*CateringServiceEnum*	1:*	Classifications of CATERING SERVICE. See allowed values below.

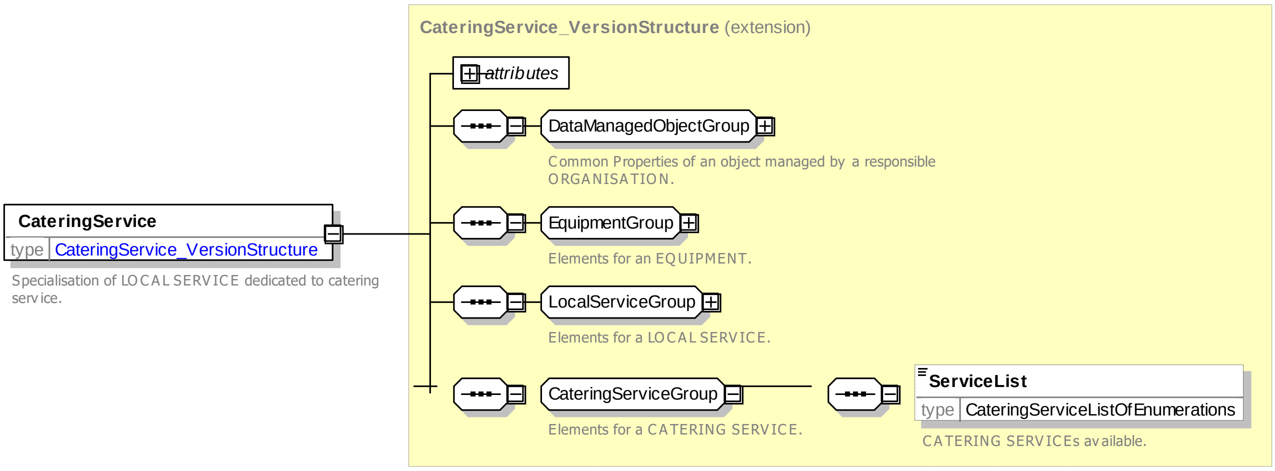


Figure 629 – *CateringService – XSD*

CateringFacility – Allowed values

Allowed values for CATERING FACILITY (*CateringFacilityEnumeration*).

Table 544 – *CateringFacility* – Allowed Values

Value	Description
bar	Bar
beverageVendingMachine	Beverage vending machines
buffet	Buffet
coffeeShop	Coffee shop
firstClassRestaurant	First class restaurant
foodVendingMachine	Food vending machines
hotFoodService	Hot food available
restaurant	Restaurant
snacks	Snacks
trolleyService	Trolley service
minibar	Mini bar
noBeveragesAvailable	No beverage available
noFoodAvailable	No food available
mealAtSeat	Meal at seat
breakfastInCar	Breakfast car
coldBuffet	Cold buffet
other	Other Catering facility

8.4.8.14 Parking Equipment

8.4.8.14.1 PARKING EQUIPMENT – Conceptual MODEL

The following figure shows detailed attributes of the PARKING EQUIPMENT model elements.

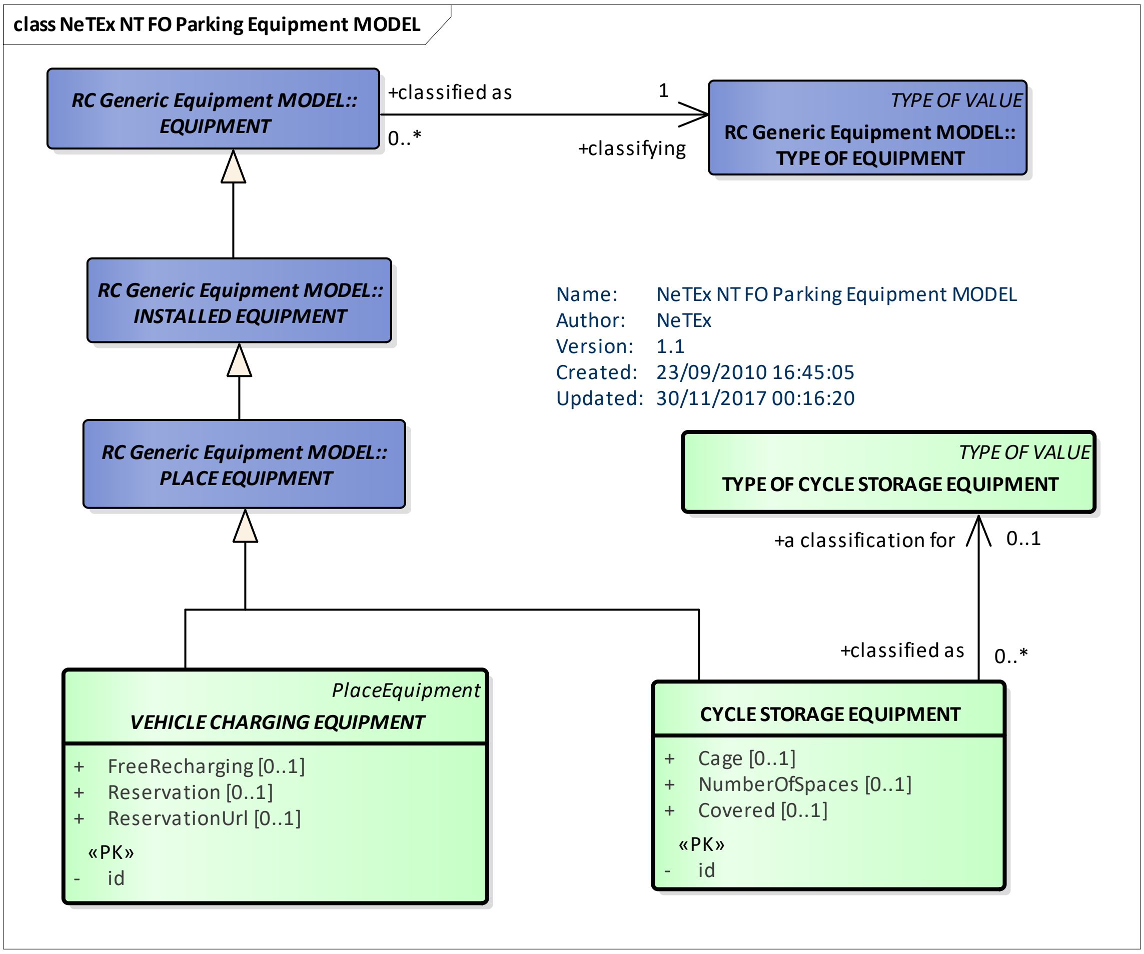


Figure 630 – PARKING EQUIPMENT – Conceptual MODEL (UML)

8.4.8.14.2 Parking Equipment – Physical Model

The following figure shows detailed attributes of the PARKING EQUIPMENT model elements.

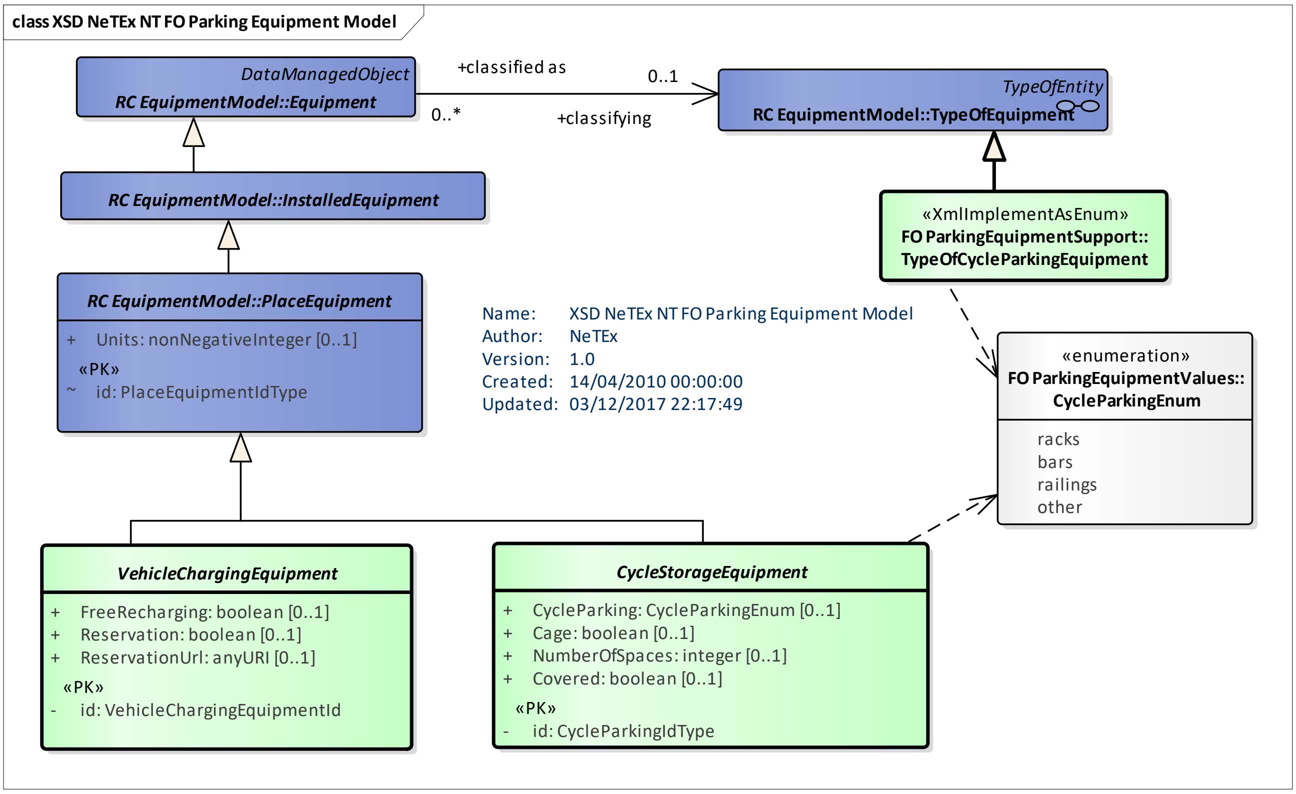


Figure 631 – Parking Equipment – Physical Model (UML) – Detail

8.4.8.14.3 Parking Equipment – Attributes and XSD

8.4.8.14.3.1 CycleParkingEquipment – Model Element

EQUIPMENT concerned with describing cycle parking facilities at a PLACE.

Further Examples of Equipment can be seen on the section on representing Stairs & Lifts

Table 545 – *CycleParkingEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PlaceEquipment*	::>	CYCLE STORAGE EQUIPMENT inherits from PLACE EQUIPMENT.
«PK»	***id***	*CycleStorageIdType*	1:1	Identifier of CYCLE STORAGE EQUIPMENT.
	NumberOfSpaces	*xsd:integer*	0:1	Number of spaces available.
«enum»	***CycleStorageType***	*CycleStorageEnum*	0:1	Nature of CYCLE STORAGE EQUIPMENT. See allowed values below.
	Cage	*xsd:boolean*	0:1	Whether there is a cage.
	Covered	*xsd:boolean*	0:1	Whether the parking is covered.

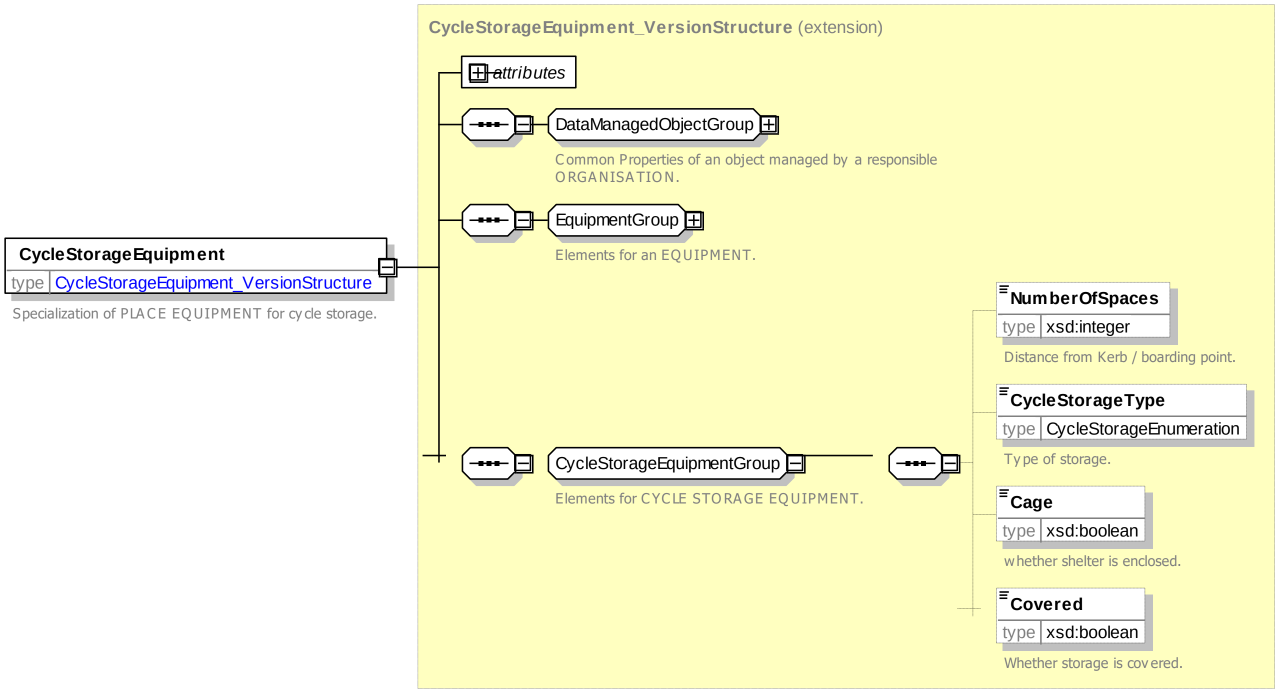


Figure 632 – *CycleStorageEquipment – XSD*

CycleStorage – Allowed values

Allowed values for CycleStorageType (*CycleParkingEnumeration*).

Table 546 – *CycleStorageType* – Allowed Values

Value	Description
racks	Racks for cycles
bars	Bars for cycles
railings	Railings for cycles
other	Other cycle parking

.

8.4.8.14.3.2 VehicleChargingEquipment – Model Element

Vehicle charching EQUIPMENT concernes with describing vehicle charging facilities at a PLACE.

Table 547 – *VehicleChargingEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PlaceEquipment*	::>	VEHICLE CHARGING EQUIPMENT inherits from PLACE EQUIPMENT.
«PK»	***id***	*VehicleChargingIdType*	1:1	Identifier of VEHICLE CHARGING EQUIPMENT.
	FreeRecharging	*xsd:boolean*	0:1	Whether recharging is free.
	Reservation-Required	*xsd:boolean*	0:1	Whether a reservation is required.

	ReservationUrl	*xsd:uri*	0:1	URL for reservation.

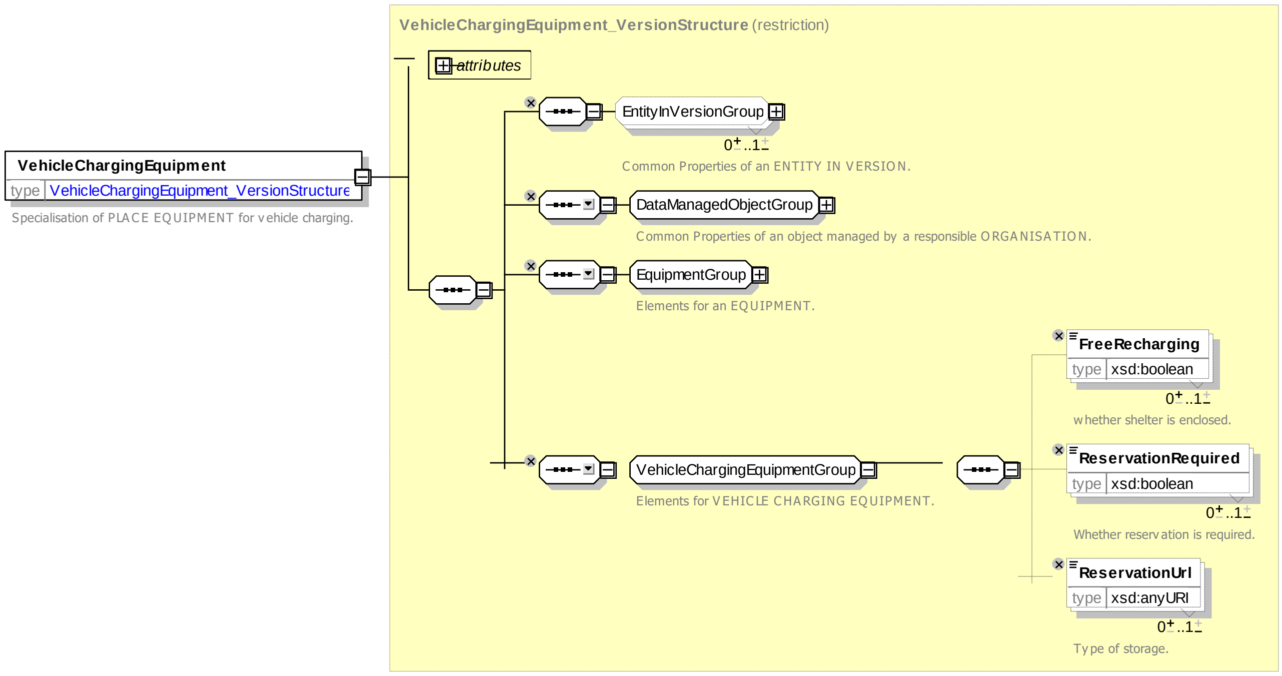


Figure 633 – *VehicleChargingEquipment – XSD*

8.4.8.14.4 XML Example of Entrance Equipment

The following XML code fragment shows an ***Entrance*** barrier with ***EntranceEquipment*** for six non-wheelchair accessible gates and one wheelchair accessible gate. The Equipment is not precisely located.

EXAMPLE XML Example of Entrance Equipment

```
<Entrance created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_A1_EI1">
    <Name>Internal Entrance to Upper Concourse from Ticket Hall</Name>
    <ParentZoneRef>tbd:9100WIMBLDN_A2</ParentZoneRef>
    <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
    <placeEquipments>
        <EntranceEquipment id="tbd:9100WIMBLDN_A2_EE1_B1">
            <Width>0.5</Width>
            <NumberOfGates>6</NumberOfGates>
            <EntranceRequiresTicket>true</EntranceRequiresTicket>
            <WheelChairPassable>false</WheelChairPassable>
        </EntranceEquipment>
        <EntranceEquipment id="tbd:9100WIMBLDN_A2_EE1_B2">
            <Width>2</Width>
            <NumberOfGates>1</NumberOfGates>
            <EntranceRequiresTicket>true</EntranceRequiresTicket>
            <WheelChairPassable>true</WheelChairPassable>
        </EntranceEquipment>
    </placeEquipments>
    <EntranceType>ticketBarrier</EntranceType>
    <isExternal>false</isExternal>
    <isEntry>true</isEntry>
```

```
    <isExit>true</isExit>
    <Width>1.0</Width>
    <Height>3.0</Height>
</Entrance>
```

8.4.8.14.5 XML Example of Local Services

The following XML code fragment shows a TICKETING SERVICE describing available ticketing that might be associated with a STOP PLACE.

EXAMPLE XML Example of Local Service Equipment

```
<TicketingService created="2010-05-17T09:30:47Z">
    <validityConditions>
        <AvailabilityCondition created="2010-05-17T09:30:47Z" id=" tbd:AC_01_Main_Opening ">
        </AvailabilityCondition>
    </validityConditions>
    <TicketCounterService>true</TicketCounterService>
    <PaymentMethod> cash creditCard debitCard cheque </PaymentMethod>
    <TicketType>all</TicketType>
    <TicketingServiceType>all</TicketingServiceType>
</TicketingService>
```

8.4.8.14.6 XML Example of a Lift

The following XML code fragment shows an ACCESS SPACE of type *lift*. It has a LIFT EQUIPMENT associated with the space that applies to all use of the lift.

EXAMPLE XML Example of Lift Access Space

```
<AccessSpace created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_L1">
    <Name>Lift Shaft to Underground</Name>
    <AccessibilityAssessment created="2010-05-17T09:30:47Z">
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation created="2010-05-17T09:30:47Z">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>false</LiftFreeAccess>
                <AudibleSignalsAvailable>false</AudibleSignalsAvailable>
                <VisualSignsAvailable>true</VisualSignsAvailable>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <Covered>indoors</Covered>
    <LevelRef> </LevelRef>
    <equipmentPlaces>
        <EquipmentPlace id="tbd:9100WIMBLDN_L1ep1">
            <placeEquipments>
                <LiftEquipment id="tbd:9100WIMBLDN_A2b_L2">
                    <Name>Lift to Platforms 5 and 6</Name>
                    <Width>1.5</Width>
                    <WheelchairTurningCircle>1</WheelchairTurningCircle>
                    <ThroughLoader>false</ThroughLoader>
                    <Automatic>true</Automatic>
                </LiftEquipment>
            </placeEquipments>
        </EquipmentPlace>
```

```xml
    </equipmentPlaces>
    <Description>Lift Shaft 1 to District line</Description>
    <AccessSpaceType>lift</AccessSpaceType>
    <accessSpaceEntrances>
        <EntranceRef>:id>tbd:9100WIMBLDN_A2b-EL1g</EntranceRef>
        <EntranceRef>tbd:9100WIMBLDN_A5_EL1</EntranceRef>
    </accessSpaceEntrances>
</AccessSpace>
::>.
```

8.4.8.14.7 XML Example of a Simple Stairs

The following XML code fragment shows a staircase represented just as a PATH LINK that has a STAIR EQUIPMENT associated with it describing a staircase of 26 steps. These models the staircase as a single piece of equipment associated with a link.

EXAMPLE XML Example of Stair Access Space

```xml
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_S2_5n6_S2">
    <Name>From Upper Concourse Corridor to Platforms 5 and 6 by stairs 2</Name>
    <AccessibilityAssessment created="2010-05-17T09:30:47Z">
        <MobilityImpairedAccess>false</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation created="2010-05-17T09:30:47Z">
                <WheelchairAccess>false</WheelchairAccess>
                <StepFreeAccess>false</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <Covered>indoors</Covered>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_A2</PlaceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
        <EntranceRef>tbd:9100WIMBLDN_A2_ES2g</EntranceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_5n6</PlaceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_PL</LevelRef>
        <EntranceRef>tbd:9100WIMBLDN_5n6_ES1pl</EntranceRef>
    </To>
    <Distance>20.00</Distance>
    <NumberOfSteps>26</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>down</FromToUpDown>
    <AccessFeatureType>stairs</AccessFeatureType>
    <TransferDuration>
        <DefaultDuration>PT3M</DefaultDuration>
        <FrequentTravellerDuration>PT5M</FrequentTravellerDuration>
        <OccasionalTravellerDuration>PT5M</OccasionalTravellerDuration>
        <MobilityRestrictedTravellerDuration>PT10M</MobilityRestrictedTravellerDuration>
    </TransferDuration>
    <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
    <placeEquipments>
        <StaircaseEquipment id="tbd:9100WIMBLDN_A2b_S2">
            <Name>Stairs to Platforms 5 and 6</Name>
            <NumberOfSteps>26</NumberOfSteps>
            <StepHeight>0.15</StepHeight>
```

```xml
            <HandrailHeight>1.2</HandrailHeight>
            <HandrailType>bothSides</HandrailType>
            <SpiralStair>false</SpiralStair>
        </StaircaseEquipment>
    </placeEquipments>
</SitePathLink>
```

8.4.8.14.8 XML Example of a Multi-flight Staircase

The following XML code fragment shows a more elaborate representation of a staircase of 26 steps described as two flights of stairs – this would allow a step by step description or visualisation, but does not add further information. There is an ***AccessSpace*** for the staircase with an overall ***StairEquipment*** associated with it. There are two distinct ***SitePathLink*** instances, one for each flight of 13 steps and an intermediate path Junction.

EXAMPLE XML Example of Multi-flight Stairs

```xml
<accessSpaces>
    <AccessSpace created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_A5">
        <Name>Underground District Line Stairs</Name>
        <AccessibilityAssessment created="2010-05-17T09:30:47Z">
            <MobilityImpairedAccess>true</MobilityImpairedAccess>
            <limitations>
                <AccessibilityLimitation created="2010-05-17T09:30:47Z">
                    <WheelchairAccess>false</WheelchairAccess>
                    <StepFreeAccess>false</StepFreeAccess>
                    <EscalatorFreeAccess>true</EscalatorFreeAccess>
                    <LiftFreeAccess>true</LiftFreeAccess>
                </AccessibilityLimitation>
            </limitations>
        </AccessibilityAssessment>
        <Covered>covered</Covered>
        <LevelRef>tbd:9100WIMBLDN_Lvl_PL</LevelRef>
        <equipmentPlaces>
            <EquipmentPlace id="tbd:9100WIMBLDN_A5_PE">
                <Description>District line lower concourse with access to platforms</Description>
                <AccessSpaceType>concourse</AccessSpaceType>
                <accessSpaceEntrances>
                    <EntranceRef>tbd:9100WIMBLDN_A5_EL1</EntranceRef>
                    <EntranceRef>tbd:9100WIMBLDN_A5_SL1</EntranceRef>
                </accessSpaceEntrances>
                    </AccessSpace>
                    <AccessSpace created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_S1">
                <Name>Staircase to DL </Name>

                <Covered>indoors</Covered>
                <placeEquipments>
                    <StaircaseEquipment id="tbd:9100WIMBLDN_S1_Eq-Sc1">
                        <Name>Stair case 1</Name>
                        <Width>2</Width>
                        <NumberOfSteps>26</NumberOfSteps>
                        <StepHeight>0.25</StepHeight>
                        <HandrailType>bothSides</HandrailType>
                        <SpiralStair>false</SpiralStair>
                        <NumberOfFlights>2</NumberOfFlights>
                    </StaircaseEquipment>
                </placeEquipments>
                <Description>Staircase down to DL concourse - two flights of 13 steps</Description>
                <AccessSpaceType>staircase</AccessSpaceType>
                <accessSpaceEntrances>
```

```
                    <EntranceRef>tbd:9100WIMBLDN_A5_ES1pl</EntranceRef>
                    <EntranceRef>tbd:9100WIMBLDN_A2_ES1g</EntranceRef>
                </accessSpaceEntrances>
</AccessSpace>
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A2_A5_S1-top">
    <Name>From Upper Concourse to district line concourse by stairs 1 - top flight </Name>
    <Covered>indoors</Covered>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_A2</PlaceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
        <EntranceRef>tbd:9100WIMBLDN_A2_ES1g</EntranceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_S1_j1</PlaceRef>
    </To>
    <Distance>10.00</Distance>
    <NumberOfSteps>13</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>down</FromToUpDown>
    <AccessFeatureType>stairs</AccessFeatureType>
    <TransferDuration>
        <DefaultDuration>PT30s</DefaultDuration>
        <MobilityRestrictedTravellerDuration>PT3M</MobilityRestrictedTravellerDuration>
    </TransferDuration>
</SitePathLink>
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A2_A5_S1-bot">
    <Name>From Upper Concourse to district line concourse by stairs 1 bottom flight </Name>
    <Covered>indoors</Covered>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_S1_j1</PlaceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_A5</PlaceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_PL</LevelRef>
        <EntranceRef>tbd:9100WIMBLDN_A5_ES1</EntranceRef>
    </To>
    <Distance>10.00</Distance>
    <NumberOfSteps>13</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>down</FromToUpDown>
    <AccessFeatureType>stairs</AccessFeatureType>
    <TransferDuration>
        <DefaultDuration> PT30s </DefaultDuration>
        <MobilityRestrictedTravellerDuration>PT3M</MobilityRestrictedTravellerDuration>
    </TransferDuration>
</SitePathLink>
```


8.4.8.14.9 XML Example of an Escalator

The following XML code fragment shows an ***EscalatorEquipment*** describing an escalator.

EXAMPLE XML Example of Escalator Equipment


```
<EscalatorEquipment id="tbd:9100WIMBLDN_A2b_E2">
    <DirectionOfUse>down</DirectionOfUse>
    <PassengersPerMinute>70</PassengersPerMinute>
    <StepHeight>0.25</StepHeight>
    <HandrailHeight>035</HandrailHeight>
    <TactileActuators>true</TactileActuators>
```

```
    <EnergySaving>true</EnergySaving>
</EscalatorEquipment>
```

8.4.8.14.10 XML Example of an Ramp

The following XML code fragment shows a ***RampEquipment*** describing a ramp.

EXAMPLE Example of Ramp Equipment

```
<RampEquipment id="tbd:9100WIMBLDN_P1_RM-1">
    <Name>Ramp from car park</Name>
    <Width>2</Width>
    <DirectionOfUse>both</DirectionOfUse>
    <Length>40</Length>
    <GradientType>gentle</GradientType>
    <HandrailType>none</HandrailType>
</RampEquipment>
```

8.4.9 Path Links

8.4.9.1 PATH LINK – Conceptual MODEL

The possible paths for passengers between the points of an interchange are represented as PATH LINKs. PATH LINKs connect the points of an interchange creating a network of possible paths. Each PATH LINK connects with a SITE COMPONENT, such as a QUAY (i.e. platform or stop) or ACCESS SPACE (i.e. hall concourse or passage) or an intermediate PATH JUNCTION.

— Each end of a PATH LINK may specify an ENTRANCE to indicate the point of connection. There doesn't have to be an ENTRANCE: for example, a ticket hall may have a well-defined entrance, but a platform or on-street stop may well not have an entrance, but rather be accessible over a whole edge.

— Each PATH LINK also describes any change in LEVEL, for example, between the concourse and lower ground platforms, as well as any EQUIPMENT (lift, steps, etc.) associated with that path link and the time taken for the path link. PATH LINKs state in which direction they can be used, and can have accessibility attributes.

— Where a QUAY is nested, for example, 'Platform 3 & 4' is made up of 'Platform 3' and 'Platform 4'. It is sufficient to have only PATH LINKs to the containing QUAY and to infer the connectivity to the contained children. Thus a smaller number of links and paths are needed to describe an interchange.

PATH LINKs are intended to describe a detailed topology for a station. For an outline topology NAVIGATION PATHs and/or CONNECTIONs should be used instead.

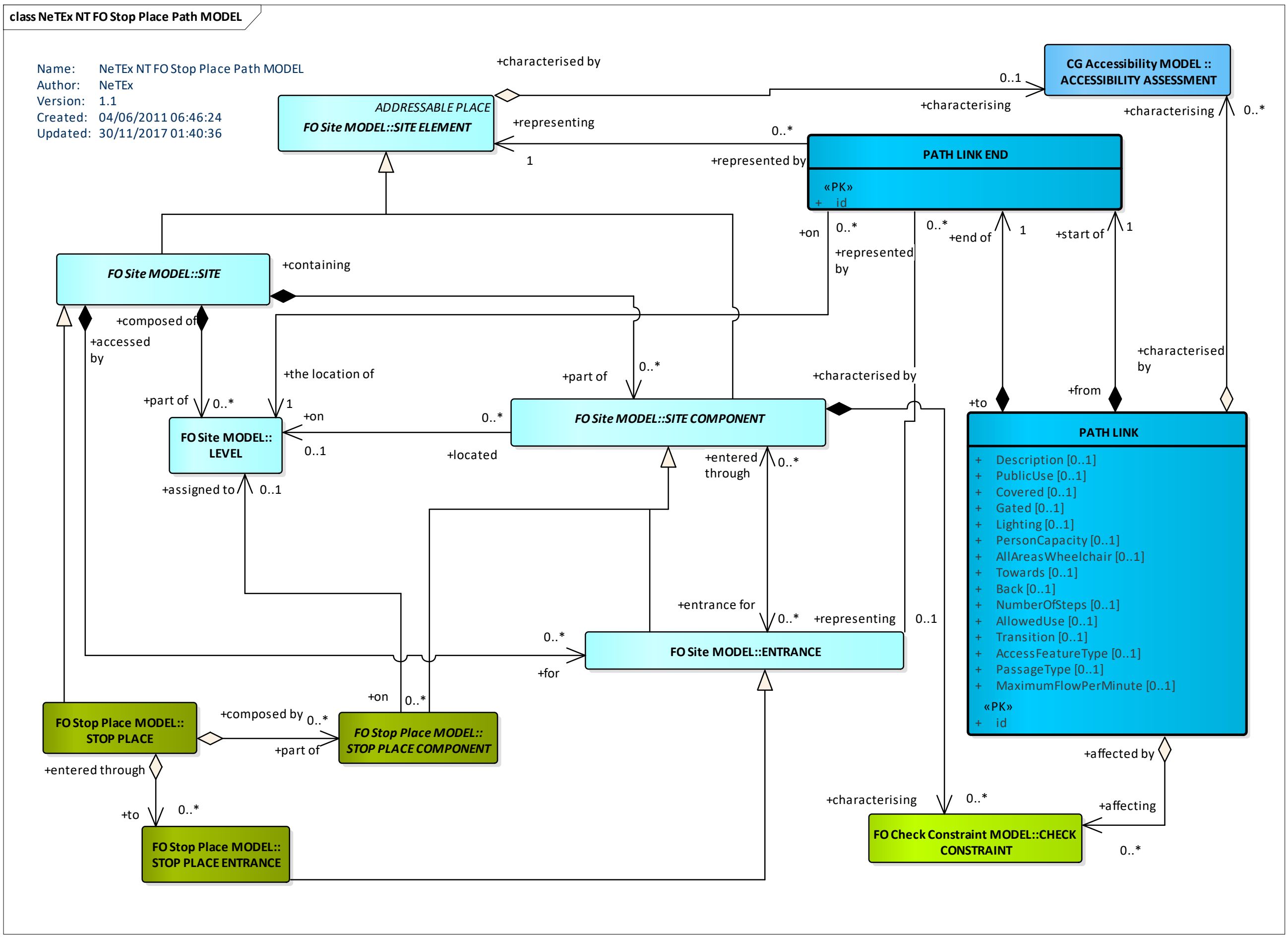


Figure 634 – PATH LINK – Conceptual MODEL (UML)

8.4.9.2 Path Links – Examples

8.4.9.2.1 Simple examples of Path Links

Each end of a PATH LINK can optionally indicate an ENTRANCE and a LEVEL.

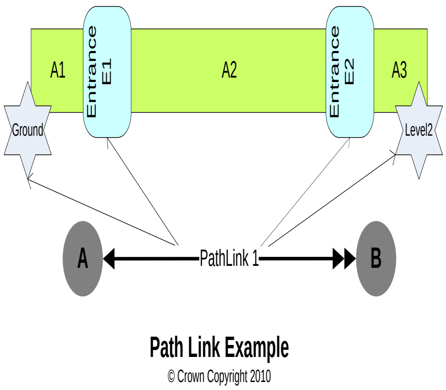


Figure 635 – Example of a single Path Link

PATH LINKs can be connected up in sequences either to STOP COMPONENTs or to the intermediate PATH JUNCTION points.

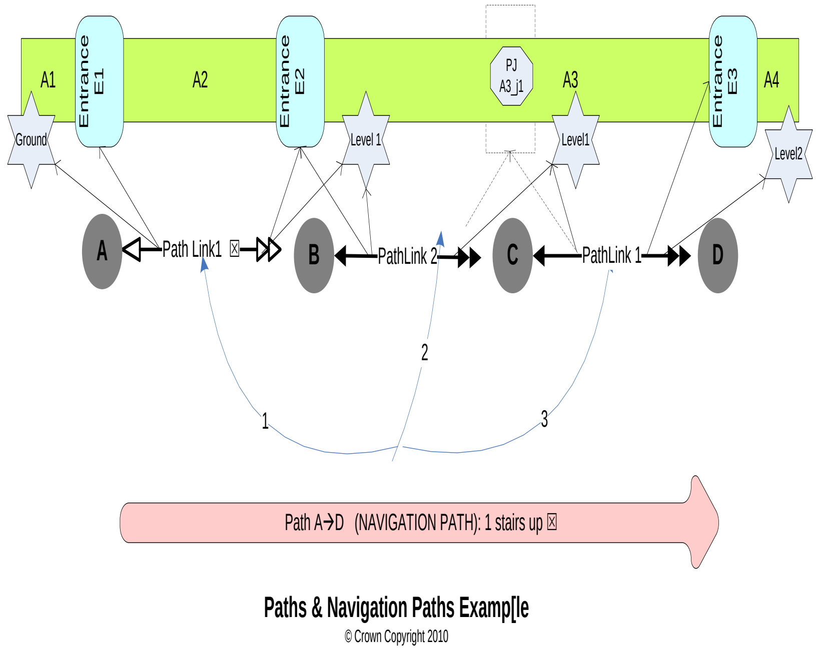


Figure 636 – Example of a sequence of Path Links

8.4.9.2.2 Simple examples of Path Links in a Stop Place

The following diagram shows an example of the use of path links to describe the topology of a simple station. There are two external entrances to a ticket hall and then a stairway to the platforms. There are two platforms, the furthest of which is reached using a barrow crossing over the tracks.

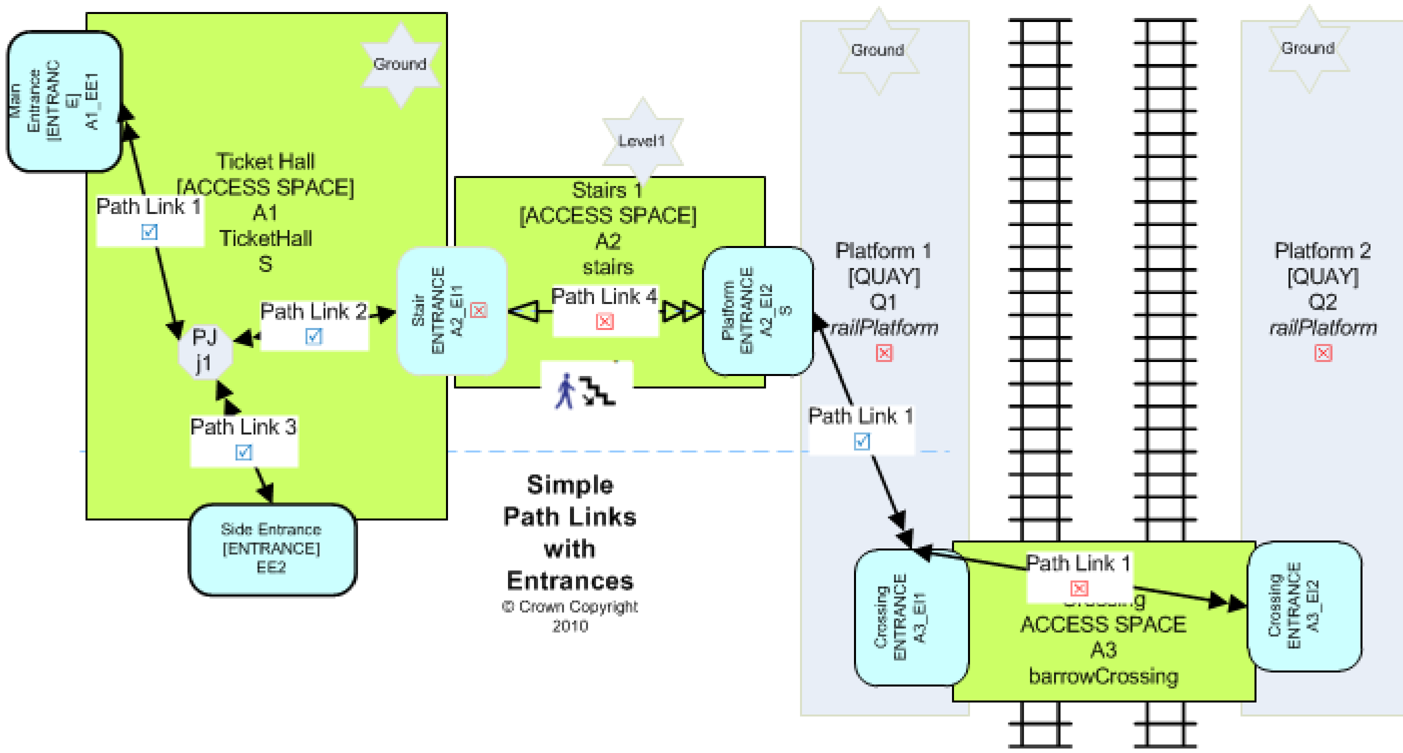


Figure 637 – Example of Path Links used to connect Access and platforms

Sometimes a path link is used simply to indicate that two spaces are connected, without specifying an ENTRANCE, for example when there a long open edge between them. The following figure shows a simple ferry stop with a long quay and an open sided shelter opening onto an access area.

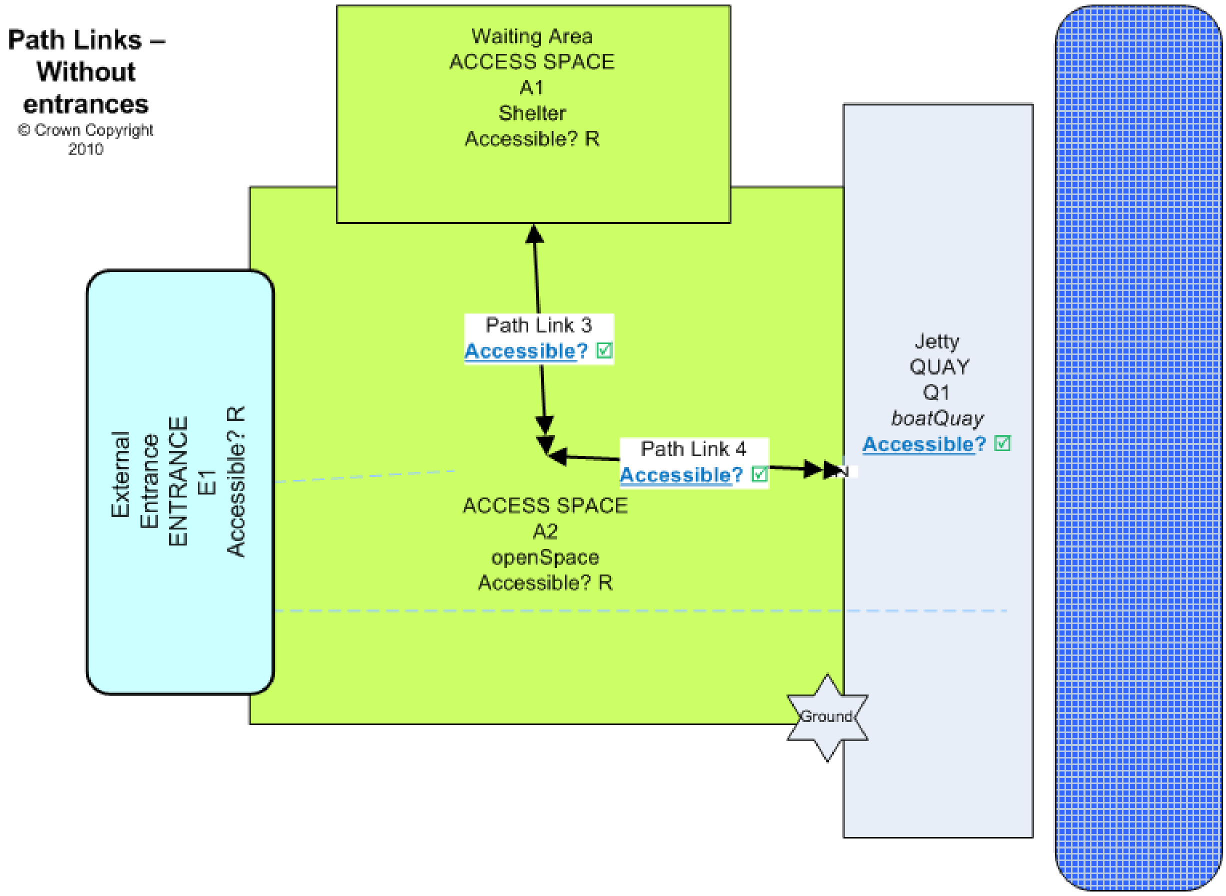


Figure 638 – Example of Path links between open areas.

8.4.9.3 Path Links – Physical Model

8.4.9.3.1 Path Link Introduction – Physical Model

The following diagram introduces the PATH LINK model: A PATH LINK can connect PLACEs and intermediate PATH JUNCTIONS. A SITE PATH LINK can conenct SITEs (that is STOP PLACEs, POINTs of INTEREST and PARKINGs) and intermediate PATH JUNCTIONS.

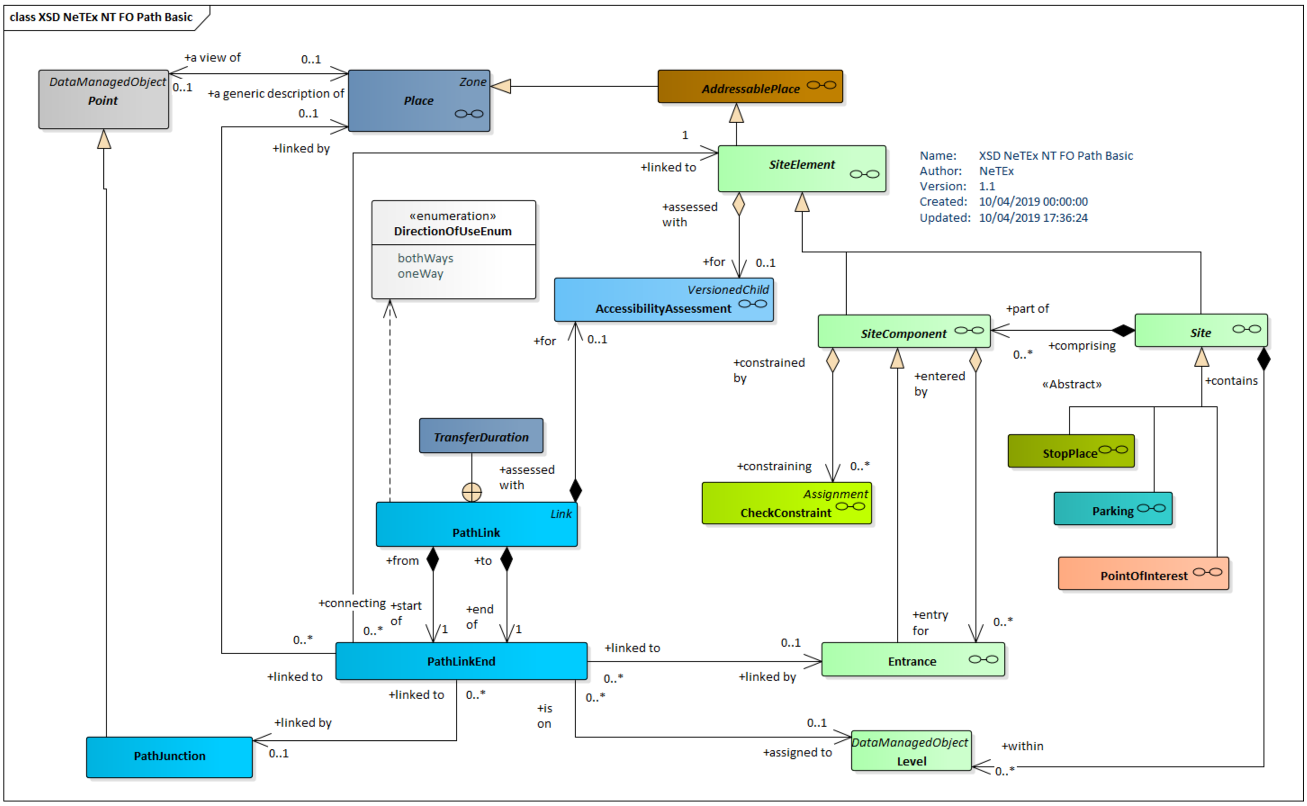


Figure 639 –Path Links Inntroduction – Physical Model (UML)

8.4.9.3.2 Path Link Details – Physical Model

The following diagram shows detailed attributes of the PATH LINK model.

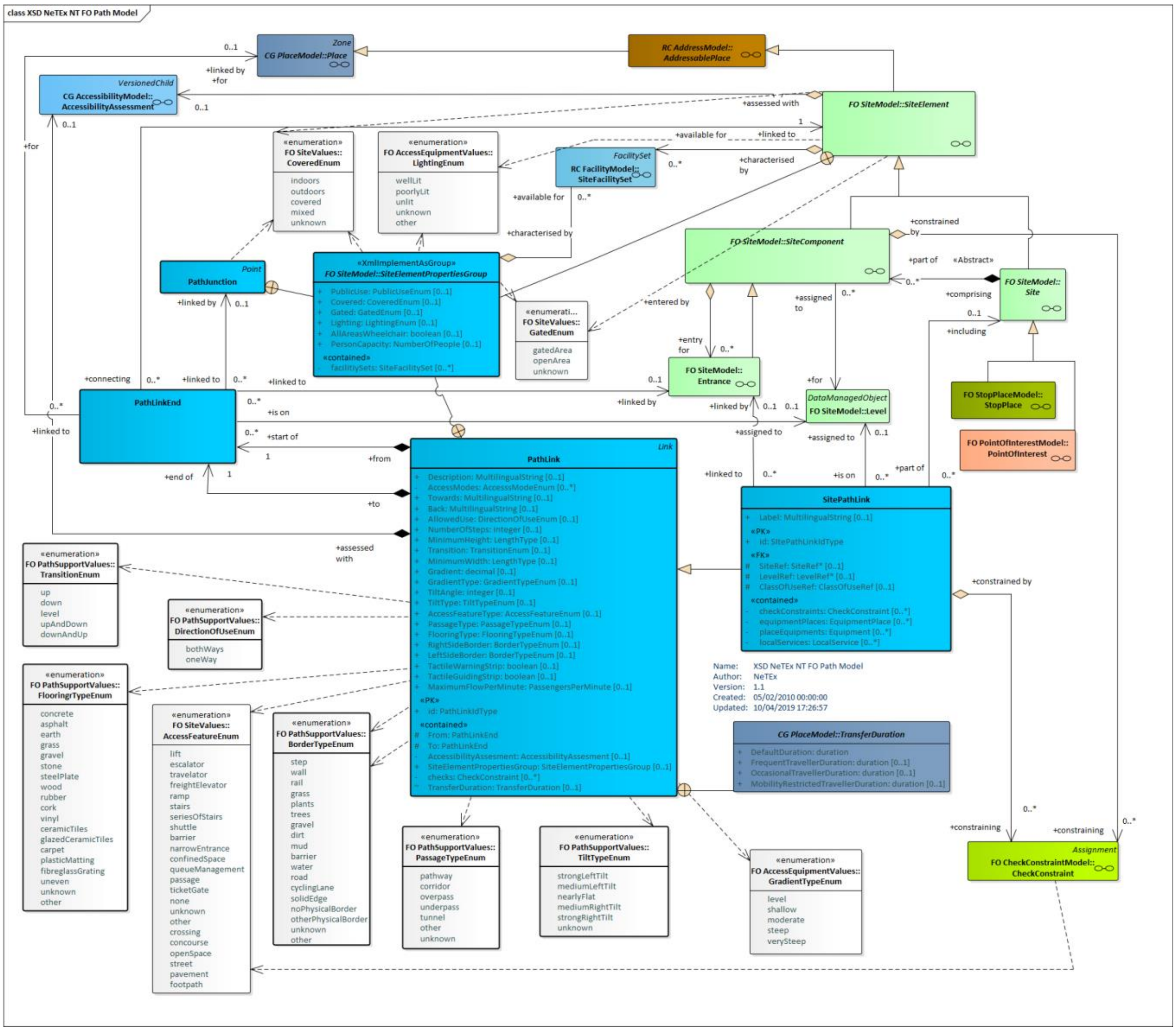


Figure 640 –Path Link – Physical Model (UML) – Detail

8.4.9.4 Path Link – Attributes and XSD

8.4.9.4.1 PathJunction – Model Element

A designated point, inside or outside of a STOP PLACE or POINT OF INTEREST, at which two or more PATH LINKs may connect or branch.

Table 548 – *PathJunction* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Point*	*::>*	PATH JUNCTION inherits from POINT.
«PK»	***id***	*PathJunctionIdType*	1:1	Identifier of a PATH JUNCTION.

«FK»	***ParentZoneRef***	*ParentZoneRef*	0:1	Parent ZONE for PATH JUNCTION.
XGRP	***SiteElement-PropertiesGroup***	***xmlGroup***	0:1	Common properties of a SITE ELEMENT. See SITE ELEMENT earlier.
	Label	*MultilingualString*	0:1	Additional Label of PATH JUNCTION.
«FK»	***SiteComponent-Ref***	*SiteComponentRef*	0:1	LEVEL of a PATH LINPATH JUNCTION is within the referenced SITE COMPONENT.K. to which one end of link connects.

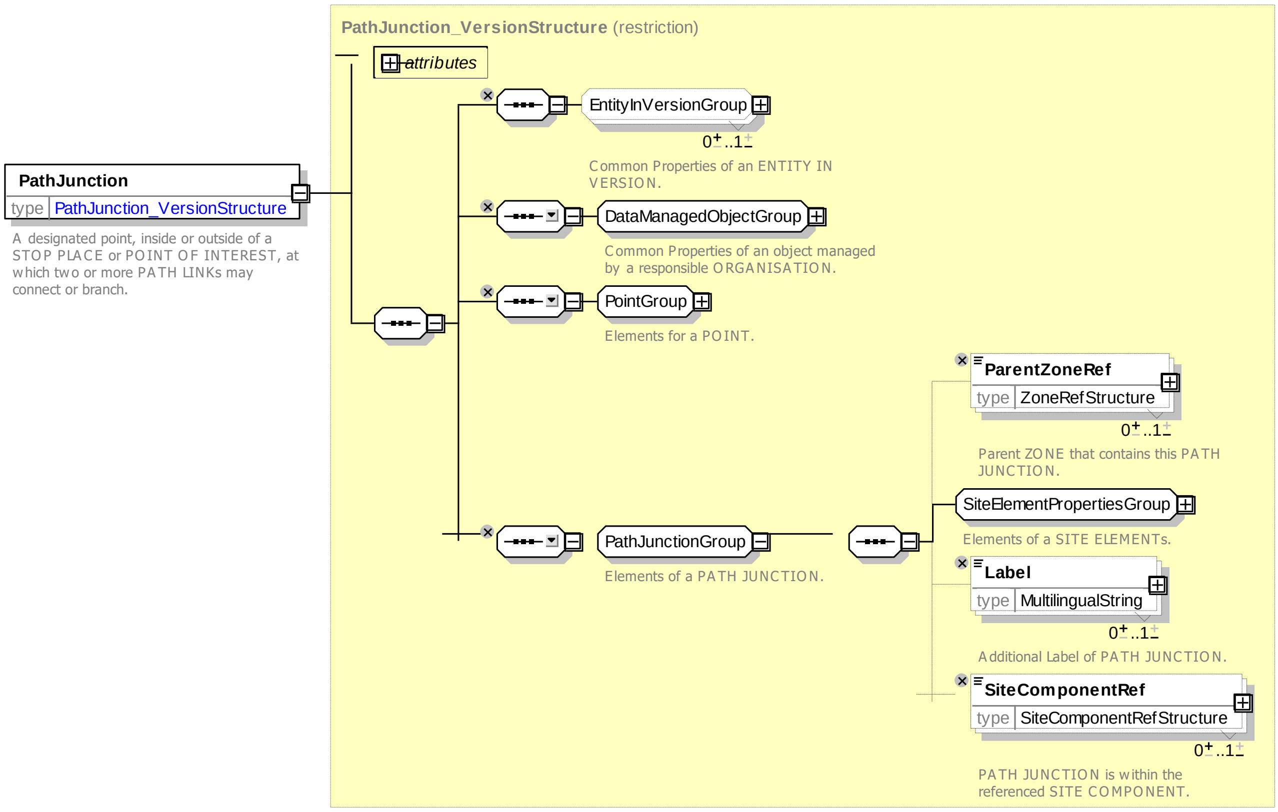


Figure 641 – *PathJunction – XSD*

8.4.9.4.2 PathLink – Model Element

A link between any two PLACEs (that is STOP PLACEs, ACCESS SPACEs or QUAYs, BOARDING POSITIONs, POINTs OF INTEREST, etc. or PATH JUNCTIONs) that represents a step in a possible route for pedestrians, cyclists or other out of vehicle passengers within or between a PLACE.

NOTE It is possible but not mandatory that a PATH LINK projects onto a more detailed set of infrastructure or mapping links that plot the spatial course, allowing it to be represented on maps and to tracking systems.

Table 549 – *PathLink* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Link*	*::>*	PATH LINK inherits from LINK.

«PK»	***id***	*PathLinkIdType*	1:1	Identifier of a PATH LINK.
«FK»	***From***	*PathLinkEnd*	1:1	Start POINT or PLACE of a PATH LINK.
«FK»	***To***	*PathLinkEnd*	1:1	End POINT or PLACE of a PATH LINK.
	Description	*MultilingualString*	0:1	Description of PATH LINK.
«cntd»	***Accessibility-Assessment***	*AccessibilityAssessment*	0:1	ACCESSIBILITY of a PATH LINK.
«enum»	***AccessModes***	*AccesssModeEnum*	0:*	ACCESS MODEs associated with STOP PLACE LINK. See TransportMode for allowed values. See TRANSPORT MODE for allowed values.
XGRP	***SiteElement-PropertiesGroup***	***xmlGroup***	0:1	Common properties of a SITE ELEMENT. See SITE ELEMENT earlier.
XGRP	***PathNavigationGroup***	***xmlGroup***	0:1	Navigation related elements of PATH LINK.
XGRP	***PathDescription-Group***	***xmlGroup***	0:1	Descriptive related elements of PATH LINK.
	MaximumFlow-PerMinute	*PassengersPerMinute*	0:1	Maximum number of passengers that can use a PATH LINK.
«cntd»	***TransferDuration***	*TransferDuration-Structure*	0:1	Time it takes to traverse a PATH LINK. See Generic Place Model.
«cntd»	***checks***	*CheckConstraint*	0:*	CHECK CONSTRAINTs associated with PATH LINK.

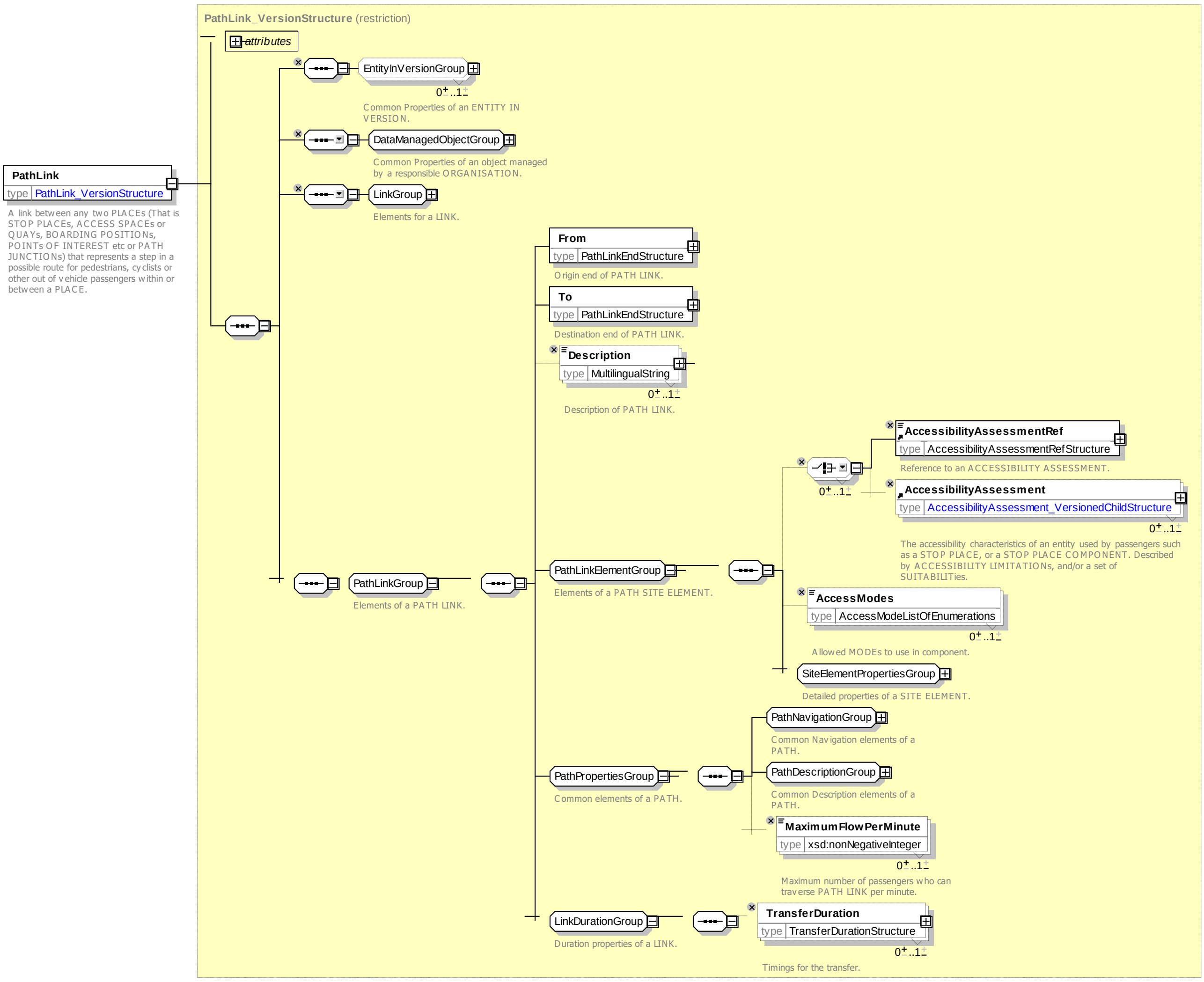


Figure 642 – *PathLink – XSD*

8.4.9.4.2.1 PathNavigationGroup – XML Group

Elements relating to navigating the PATH.

Table 550 – *PathNavigationGroup* – Group

Classification	Name	Type	Cardinality	Description
	Towards	*MultilingualString*	0:1	Direction heading to show for PATH LINK when travelling in its FROM / TO sense.
	Back	*MultilingualString*	0:1	Direction heading to show for PATH LINK when travelling in its TO / FROM sense.
	NumberOfSteps	*xsd:integer*	0:1	Number of steps involved in using a PATH LINK.
	MinimumHeight	*LengthType*	0:1	Minimum height of PATH LINK. +v1.1
	MinimumWidth	*LengthType*	0:1	Minimum width of PATH LINK. +v1.1

«enum»	***AllowedUse***	*DirectionOfUseEnum*	0:1	Allowed direction of use of PATH LINK.
«enum»	***Transition***	*TransitionEnum*	0:1	Transition for with PATH LINK: up, down, level. See allowed values below.
	Gradient	*xsd:integer*	0:1	Maximum gradient in degrees (in the direction of the PATH LINK way). +v1.1
«enum»	***GradientType***	*GradientTypeEnum*	0:1	Coded value of the maximum gradient. See ACCESS EQUIPMENT for allowed values.+v1.1
	TiltAngle	*xsd:integer*	0:1	Maximum Tilt angle in degrees between -20 and 20 (in the direction of the PATH LINK way). See allowed values below. +v1.1
«enum»	***TiltType***	*TiltTypeEnum*	0:1	Coded value of the maximum tilt. See allowed values below. +v1.1

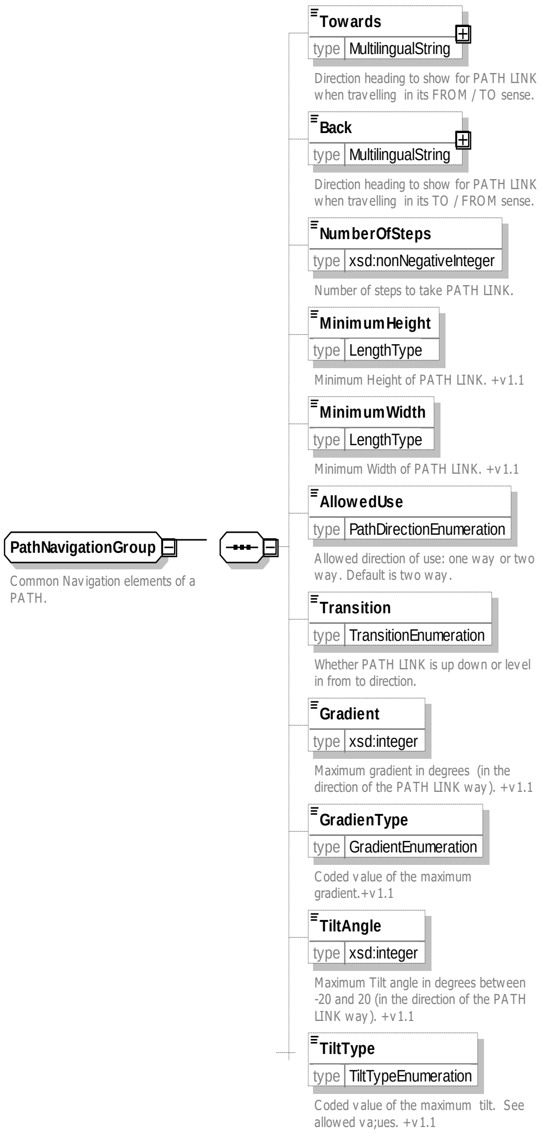


Figure 643 – *PathNavigationGroup – XSD*

AllowedUse – Allowed values

Allowed values for ***AllowedUse*** (*AllowedUseEnumeration*).

Table 551 – *AllowedUse* – Allowed values

Value	Description
oneWay	Path may be followed only one way
twoWay	Path may be followed both ways

TransitionType – Allowed values

The following table shows the allowed values for ***Transition*** (*TransitionEnumeration*).

Table 552 – *TransitionType* – Allowed values

Value	Description
up	Up
down	Down
level	Along level
upAndDown	Up then down
downAndUp	Down then up

TiltType – Allowed values

Allowed values for ***TiltType*** (*TiltTypeEnumeration*).

Table 553 – *TiltType* – Allowed values

Value	Description
strongLeftTilt	*Strong Tilt to left*
mediumLeftTilt	*Medium Tilt to left*
nearlyFlat	*Flat or nearly flat*
mediumRightTilt	*Medium Tilt to right*
strongRightTilt	*Strong Tilt to right*
unlnown	*Unown tilte*

8.4.9.4.2.2 PathDescriptionGroup – XML Group

A link between any two PLACEs (that is STOP PLACEs, ACCESS SPACEs or QUAYs, BOARDING POSITIONs, POINTs OF INTEREST, etc. or PATH JUNCTIONs) that represents a step in a possible route for pedestrians, cyclists or other out of vehicle passengers within or between a PLACE.

NOTE It is possible but not mandatory that a PATH LINK projects onto a more detailed set of infrastructure or mapping links that plot the spatial course, allowing it to be represented on maps and to tracking systems.

Table 554 – *PathDescriptionGroup* – Group

Classification	Name	Type	Cardinality	Description
«enum»	***AccessFeatureType***	*AccessFeatureEnum*	0:1	Access feature type associated with PATH LINK. See allowed values below.
«enum»	***PassageType***	*PassageTypeEnum*	0:1	Type of passage traversed by PATH LINK, if any. This provides a more precise description of the

				AccessFeatureType. See ACCESS SPACE for allowed values.
«enum»	***FlooringType***	*FlooringTypeEnum*	0:1	Type of flooring of the walking surface. See allowed values below. +v1.1
«enum»	***RightSideBorder***	*B**order**TypeEnum*	0:1	Type of border on the right side of the path (in the direction of the PATH LINK). See allowed values below.+v1.1
«enum»	***LeftSideBorder***	*B**order**TypeEnum*	0:1	e of border on the left side of the path (in the direction of the PATH LINK). See allowed values below.+v1.1
«enum»	***TactileWarningStrip***	*TactileWarningStrip-Enum*	0:1	Nature of the tactile warning strips (in the direction of the PATH LINK). See CROSSING EQUIPMENT for allowed values+v1.1
	TactileGuidingStrip	*xsd:boolean*	0:1	Whether all areas of component are accessible in a Wheelchair.

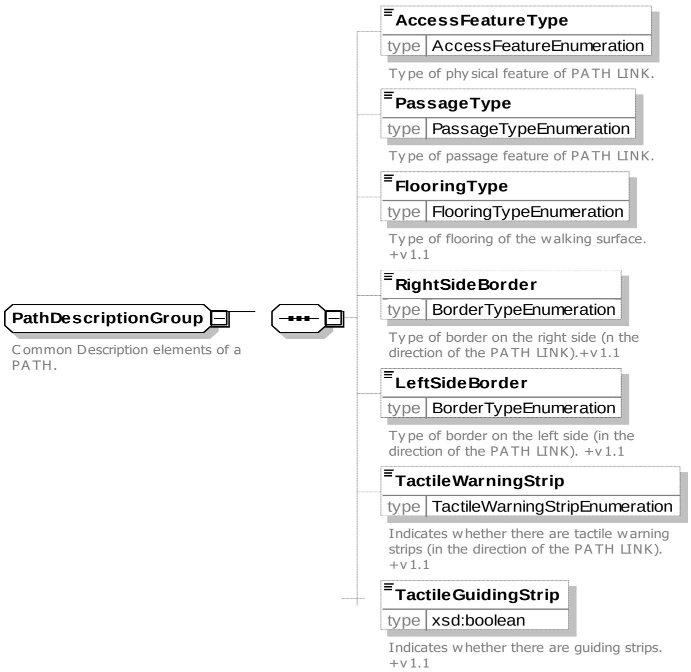


Figure 644 – *PathDescriptionGroup – XSD*

AccessFeatureType – Allowed values

The following table shows the allowed values for ***AccessFeatureType*** (*AccessFeatureEnumeration*).

Table 555 – *AccessFeatureType* – Allowed values

Value	Description
lift	Lift.
stairs	Stairs.
seriesOfStairs	Series of stairs.
escalator	Escalator.
travelator	Travelator.
ramp	Ramp.
shuttle	Shuttle.

crossing	Crossing.
barrier	Barrier.
narrowEntrance	Narrow entrance.
hall	Hall.
concourse	Concourse.
confinedSpace	Confined space.
queueManagement	Queue management.
none	None.
unknown	Unknown.
other	Other.
openSpace	Open space.
street	Street.
pavement	Pavement.

FlooringType – Allowed values

Allowed values for ***FlooringType*** (*FlooringTypeEnumeration*).

Table 556 – FlooringType – Allowed values

Value	Description
concrete	Concrete
asphalt	Asphalt
earth	Earth ("dirt")
grass	Grass
gravel	Gravel
stone	Stone
steelPlate	Steel Plate
wood	Wood
rubber	Rubber
cork	Cork
vinyl	Vinyl
ceramicTiles	Ceramic tiles.
glazedCeramic-Tiles	Glazed ceramic tiles.
carpet	carpet.
plasticMatting	Plastic Matting
fibreglassGrating	Fibreglass grating
uneven	Uneven surface
unknown	Unknown Feature
other	Other Feature

BorderType – Allowed values

Allowed values for ***BorderType*** (*B**order**TypeEnumeration*).

Table 557 – FlooringType – Allowed values

Value	Description
step	Step.
wall	Wall
rail	Rail
grass	Grass
plants	Plants
trees	Trees
gravel	Gravel
dirt	Dirt
mud	Mud
barrier	Barrier
water	Water
road	Road
cyclingLane	Cycling lane
solidEdge	Solid Edge
*noPhysicalB**order***	No physical border
*otherPhysicalB**order***	Other physical border
unknown	Unknown Feature
other	Other Feature

8.4.9.4.2.3 **PathLinkEnd – Model Subelement**

Beginning or end SITE for a PATH LINK. May be linked to a specific LEVEL of the SITE.

Table 558 – *PathLinkEnd* – Structure

Classification	Name	Type	Cardinality	Description
«FK»	***PlaceRef***	(*PlaceRef*)	1:1	Connection POINT or PLACE of a PATH LINK.
«FK»	***LevelRef***	*LevelRef*	0:1	LEVEL of a PATH LINK. to which one end of link connects.
«FK»	***EntranceRef***	(*EntranceRef*)	0:1	ENTRANCE associated with start SITE of a PATH LINK.

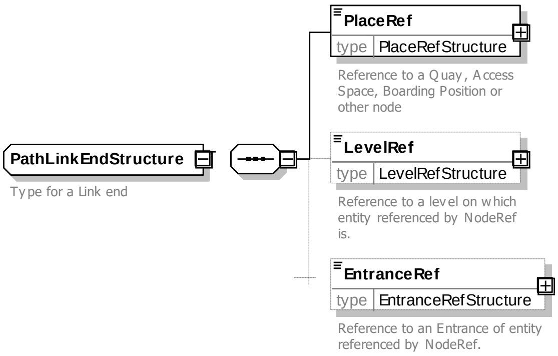


Figure 645 – *PathLinkEnd – XSD*

8.4.9.4.3 SitePathLink – Model Element

A SITE PATH LINK is a PATH LINKbetween two nodes that are SITE components, i.e. within a STOP PLACE or POINT OF INTEREST. It has the additional properties of a SITE COM.PONENT, e.g. a LEVEL and CHECK CONSTRAINTs.

Table 559 – *SitePathLink* **– Element**

Classification	Name	Type	Cardinality	Description
::>	*::>*	*<u>Link</u>*	*::>*	SITE PATH LINK inherits fromPATH LINK.
«PK»	***id***	*PathLinkIdType*	1:1	Identifier of a SITE PATH LINK.
«FK»	***SiteRef***	*(SiteRef)*	0:1	Reference to parent of SITE COMPONENT.
«FK»	***LevelRef***	*LevelRef*	0:1	Reference to LEVEL associated with the SITE COMPONENT.
«FK»	***ClassOfUseRef***	*ClassOfUseRef*	0:1	Reference to CLASS OF USE associated with the SITE COMPONENT.
«cntd»	***checkConstraints***	*<u>CheckConstraint</u>*	0:*	CHECK CONSTRAINTs associated with SITE COMPONENT.
«cntd»	***equipmentPlaces***	*<u>EquipmentPlace</u>*	0:*	EQUIPMENT PLACEs associated with STOP PLACE COMPONENT.

«cntd»	***equipments***	*(Equipment)*	0:*	EQUIPMENTs associated with SITE COMPONENT.
«cntd»	***localServices***	*(LocalService)*	0:*	LOCAL SERVICEs associated with SITE COMPONENT.
	Label	*MultilingualString*	0:1	Additional lanel for the SITE PATH LINK.

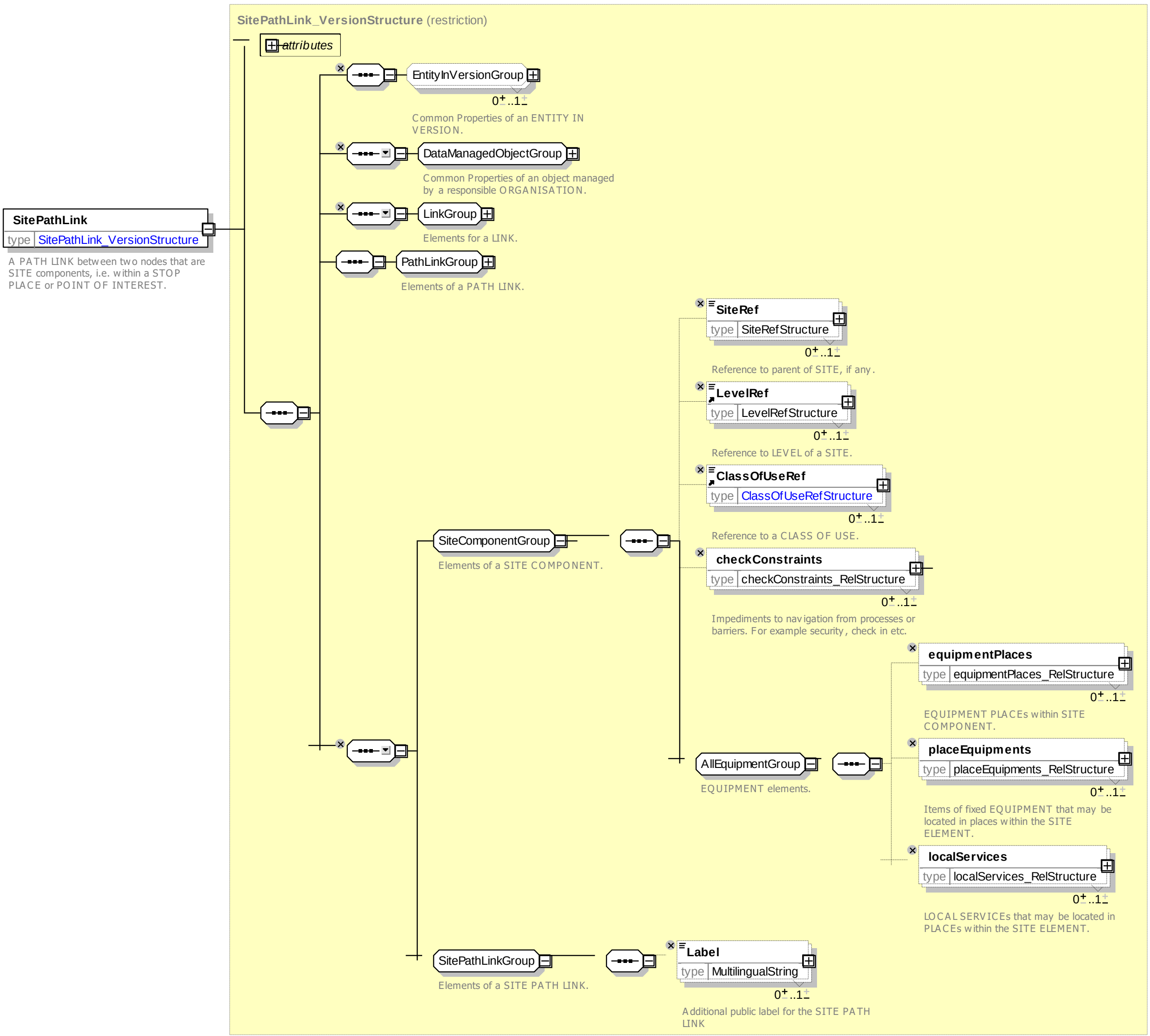


Figure 646 – *SitePathLink – XSD*

8.4.9.5 Path Links – XML Examples

8.4.9.5.1 XML example of a Path Link - Outdoors

The following XML code fragment shows a single PATH LINK that connects an ENTRANCE to a PATH JUNCTION out in the street.

EXAMPLE XML Example of external PathLink

```
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A1_EE1_FC-j1">
    <Name>From Forecourt main entrance to Path Junction w1 in forecourt</Name>
    <Covered>outdoors</Covered>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_A1</PlaceRef>
        <EntranceRef>tbd:9100WIMBLDN_A1_EE1</EntranceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_FC_J1</PlaceRef>
    </To>
    <Distance>5.00</Distance>
    <NumberOfSteps>0</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>level</FromToUpDown>
    <AccessFeatureType>street</AccessFeatureType>
    <LevelRef>tbd:9100WIMBLDN_Lvl_S0</LevelRef>
</SitePathLink>
```

8.4.9.5.2 XML example of a Path Link – Indoors

The following XML code fragment shows a PATH LINK that connects an external ENTRANCE to an internal ENTRANCE. It is further marked with accessibility attributes.

EXAMPLE XML Example of PathLink within a Station

```
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A1-EE1_A1-EI1">
    <Name>From Ticket hall external entrance to Upper concourse internal entrance</Name>
    <AccessibilityAssessment created="2010-05-17T09:30:47Z">
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation created="2010-05-17T09:30:47Z">
         id=" tbd:9100WIMBLDN_lnk_A1-EE1_A1-EI1-acc01">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <Covered>indoors</Covered>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_A1</PlaceRef>
        <EntranceRef>tbd:9100WIMBLDN_A1_EE1</EntranceRef>
<LevelRef>tbd:9100WIMBLDN_Lvl_ST</LevelRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_A1</PlaceRef>
        <EntranceRef>tbd:9100WIMBLDN_A1_EI1</EntranceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
    </To>
    <Distance>4.0</Distance>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>level</FromToUpDown>
    <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
    <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
    <checks>
        <CheckConstraint>
     id="Ck001">
            <CheckProcess>ticketValidation</CheckProcess>
```

```xml
                <AccessFeatureType>barrier</AccessFeatureType>
                <Congestion>queue</Congestion>
                <MinimumLikelyDelay>PT30S</MinimumLikelyDelay>
                <AverageDelay>PT5MS</AverageDelay>
                <MaximumLikelyDelay>PT10M</MaximumLikelyDelay>
            </Check>
        </checks>
    </SitePathLink>
```

8.4.10 Navigation Paths

8.4.10.1 NAVIGATION PATH – Conceptual MODEL

Sequences of PATH LINKs can be assembled into named NAVIGATION PATHs to guide the user through an interchange. The model for NAVIGATION PATHs thus comprises two-levels: the NAVIGATION PATHs, and the the PATHLINKs thatmake them up.

A NAVIGATION PATH normally contains a sequence of one or more simple point to point PATH LINKs that link nodes: nodes may be QUAYs or ACCESS SPACEs or PATH JUNCTIONs – intermediate branch points. PATH JUNCTIONs make it possible to use the same PATH LINKs in many different NAVIGATION PATHS.

It is possible for a NAVIGATION PATH to be used just as a summary – i.e. without PATH LINKs, for example in order to record transfer times.

NAVIGATION PATHs and PATH LINKs are normally specified as properties of the overall SITE for example STOP PLACE or POINT OF INTEREST, rather than a specific component such as a QUAY.

The NAVIGATION PATH that can be given a meaningful name that identifies a route to the user – e.g. "*Entrance Hall to Platform 1'.* NAVIGATION PATHs can also be given accessibility attributes so that searches can be filtered according to the specific needs of the users, for example to avoid steps or escalators, and summary data such as the number of lifts, escalators and traversal time.

Path elements and the whole path itself may be given an ACCESSIBILITY ASSESSMENT.

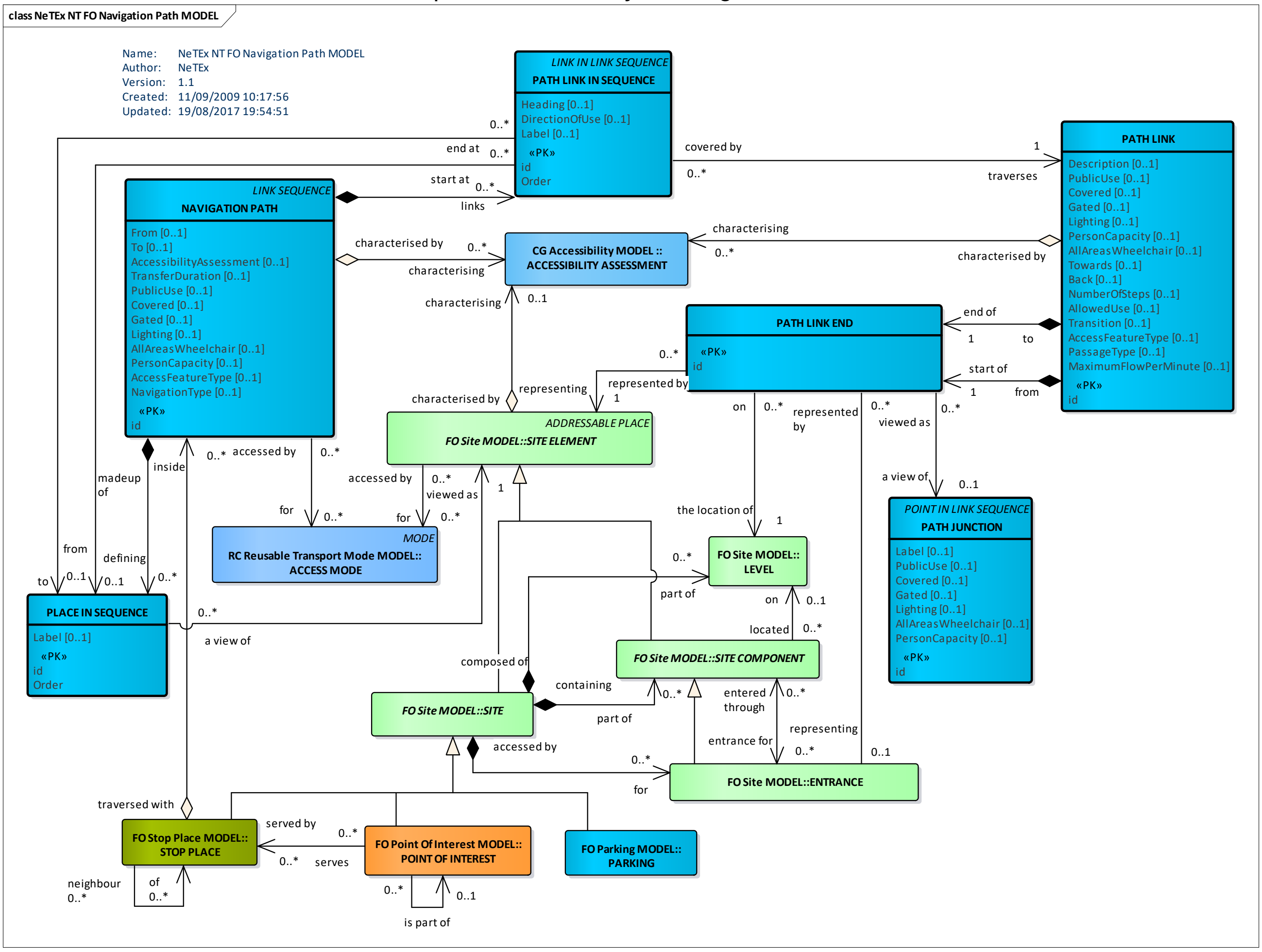


Figure 647 – NAVIGATION PATH – Conceptual MODEL (UML)

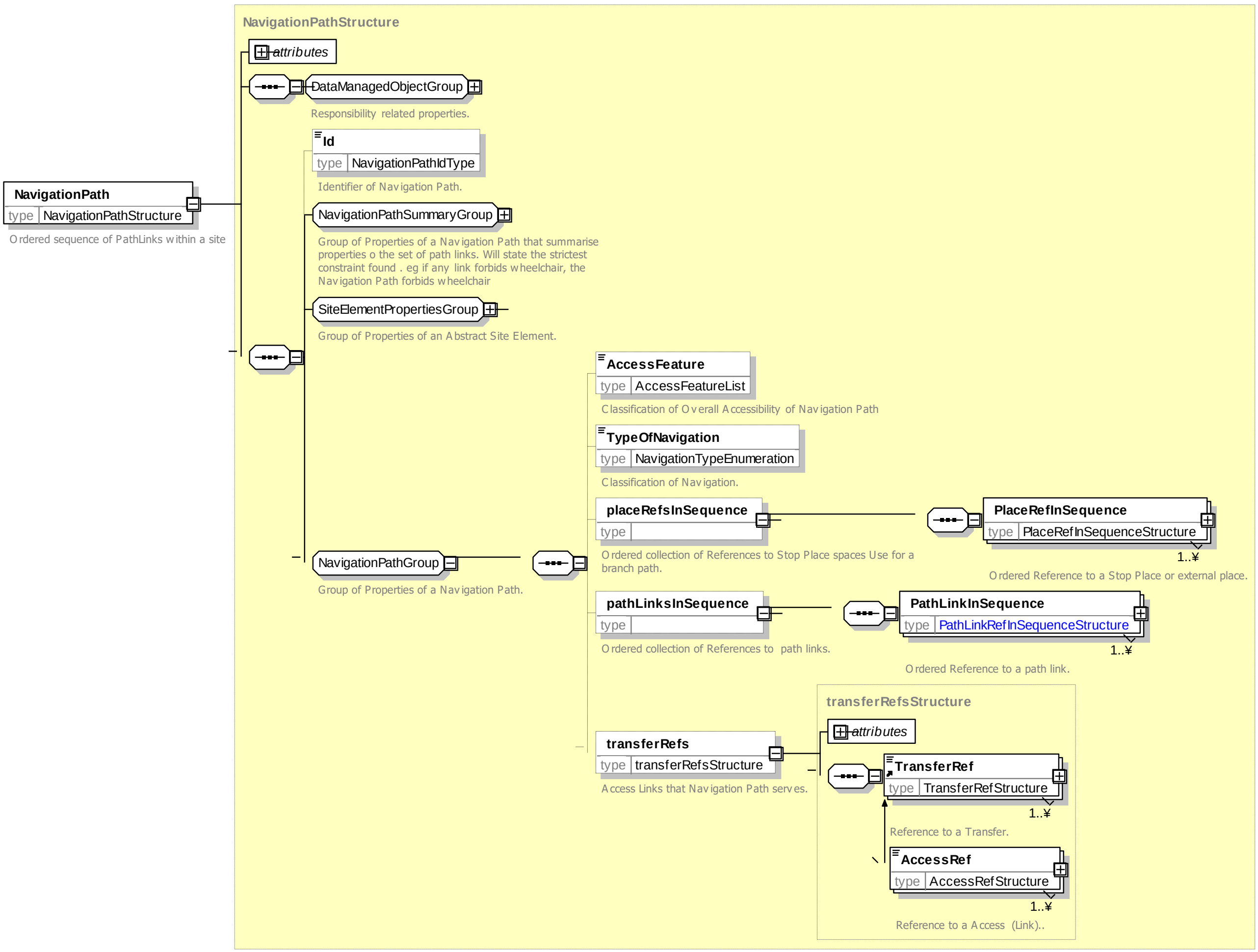


Figure 648 – *NeTEx* Path Links & Navigation Paths

8.4.10.2 Navigation Paths – Examples

8.4.10.2.1 Wimbledon Example showing Nodes and Path Links

The following figure taken from the Wimbledon example gives an example of a medium size interchange that illustrates many of the elements discussed above. It shows Wimbledon station as a set of nodes connected by path links to create a topological model of the interchange.

— Black triangles indicate elements that correspond to *NaPTAN* points.

— The path links (arrowed lines) are shown that connect the access spaces and quays: these can also indicate the use of specific entrances at either end, if relevant. The precise choice of path links will reflect the level of detail that one wishes to capture about the topology. Elements which fall within the gated area (light green), i.e. that require a ticket to access, can be distinguished from areas of unrestricted access.

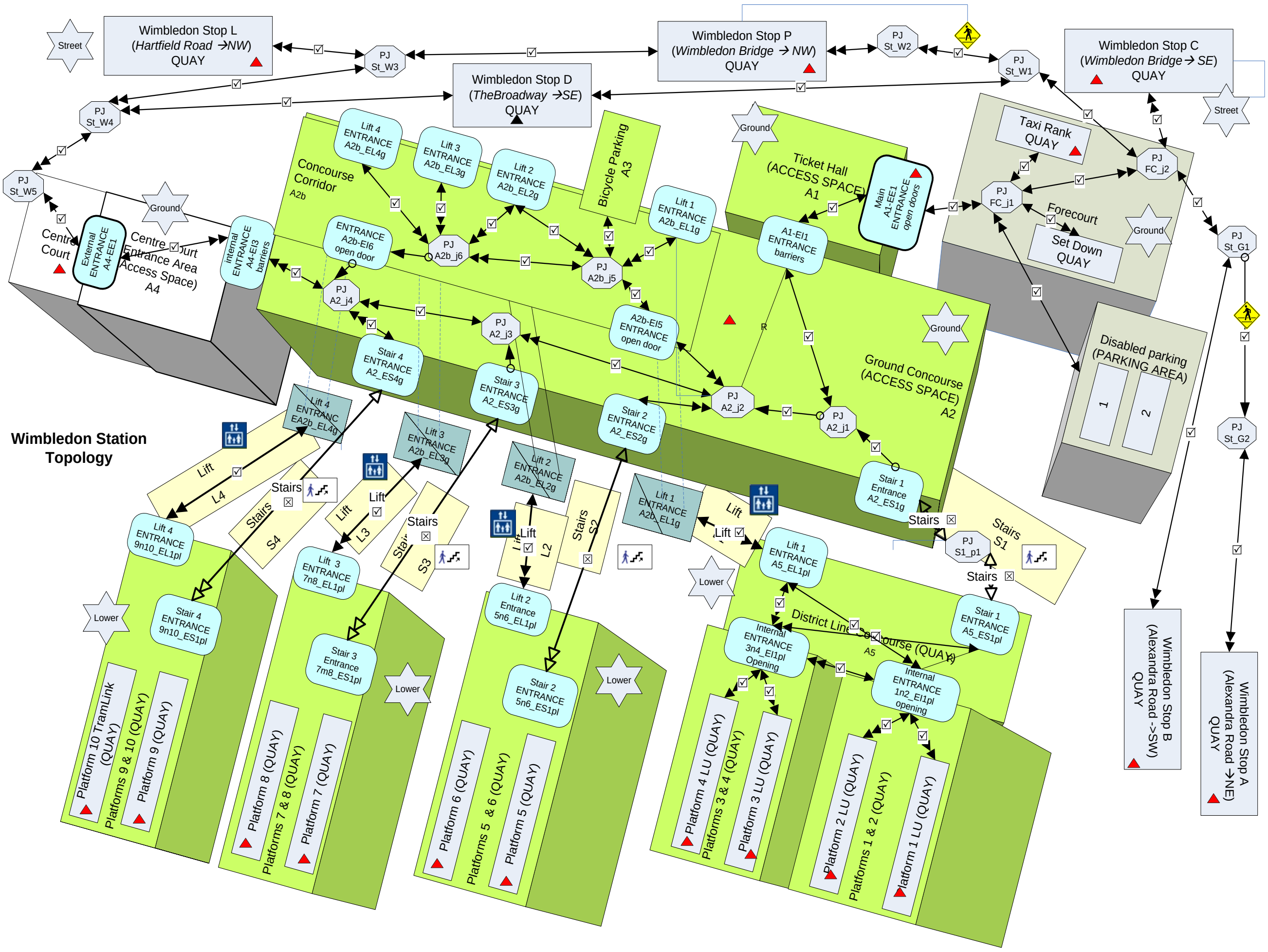


Figure 649 – Example – Wimbledon Nodes & Path links

8.4.10.2.2 Wimbledon Example showing Navigation Paths

PATH LINKs typically describe detailed connectivity between two components. To describe a route through a station a NAVIGATION PATH is used – a sequence of PATH LINKs that can be given a meaningful name to a user – for example "*Platform 1 to Platform 5".* NAVIGATION PATHs may be given an overall accessibility. The following figure shows a few of the possible paths for *Wimbledon*, corresponding to some of those shown in *Direct Enquiries*. As previously noted, the NAVIGATION PATHs may either be created manually, or be computed dynamically by an indoor routing engine.

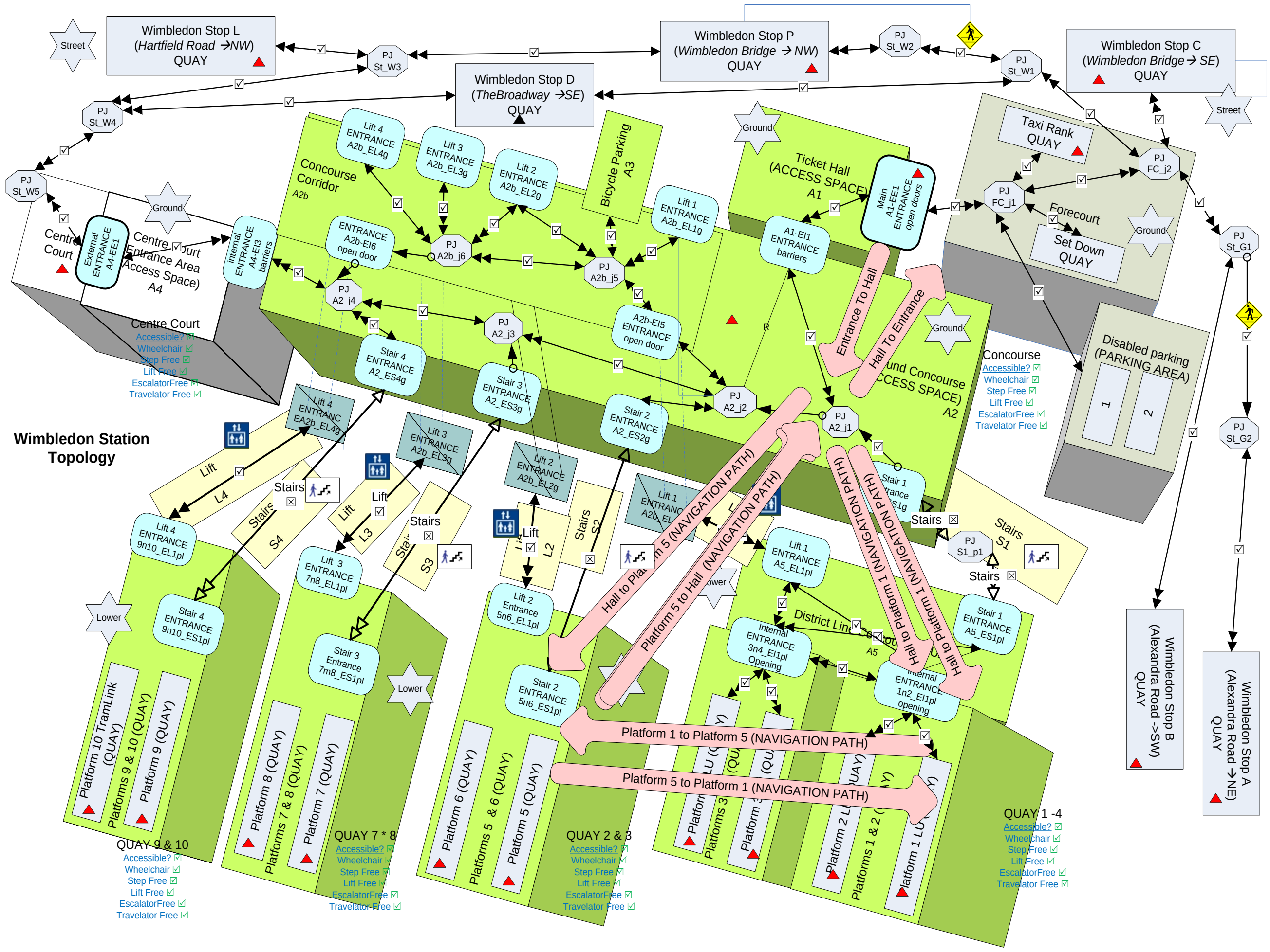


Figure 650 – Example – Some Navigation Paths for Wimbledon

8.4.10.2.2.1 Wimbledon example Navigation Path

The following figure shows an example of a NAVIGATION PATH from a bus stop to a platform: it describes an accessible route that traverses a sequence of spaces by following PATH LINKs. It uses a lift.

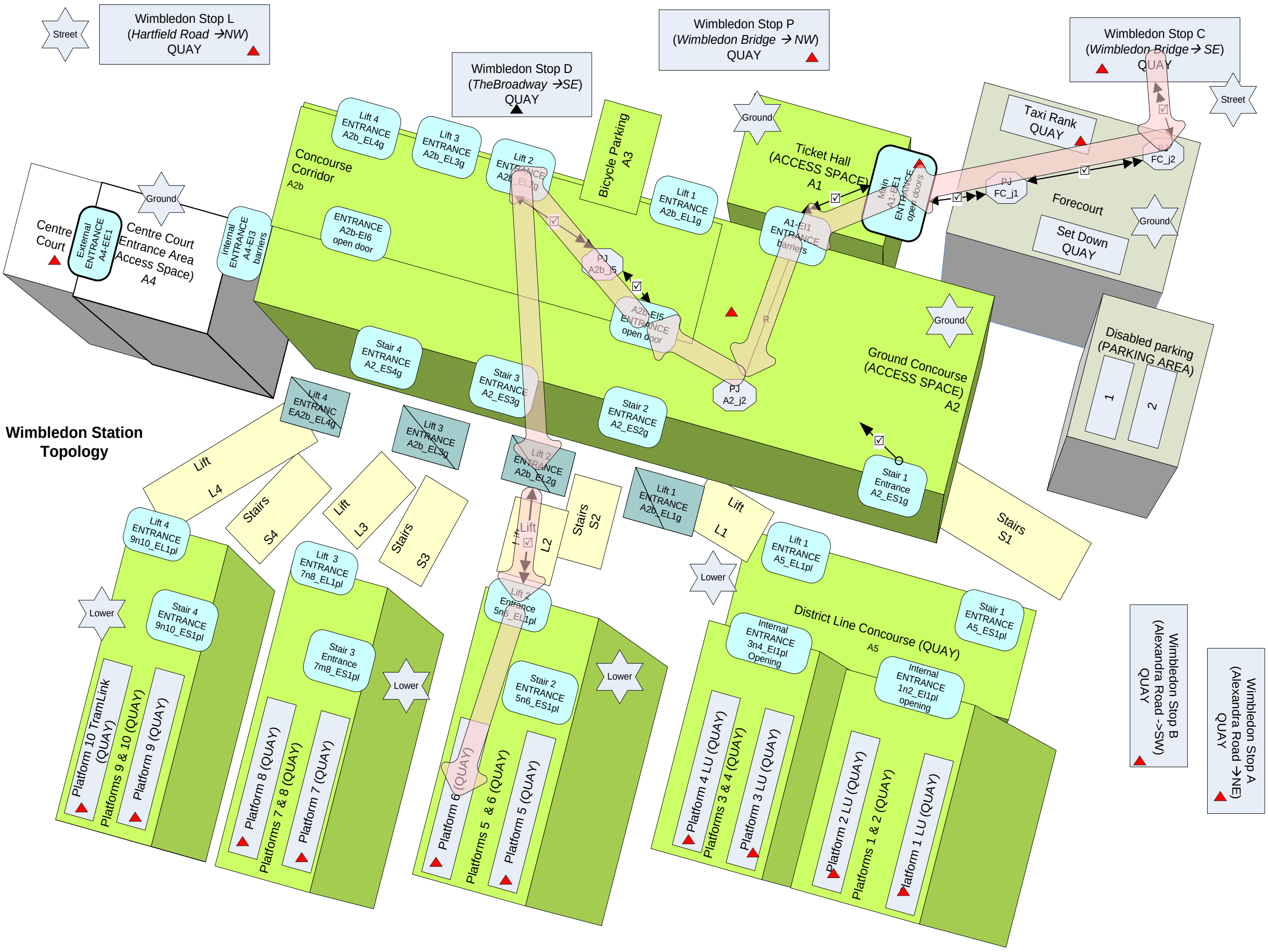


Figure 651 – Example – Wimbledon path from Stop Q to Platform 6

8.4.10.2.3 2012 Olympic example Navigation Path

The following figure of the London 2012 Olympic main site shows a large complex site with a number of different points of interest (arenas, stadia, etc.) which people need to reach from the available public transport stops. Access is only through a number of designated entrances. At peak times there are potential delays from congestion, for example for the security checks.

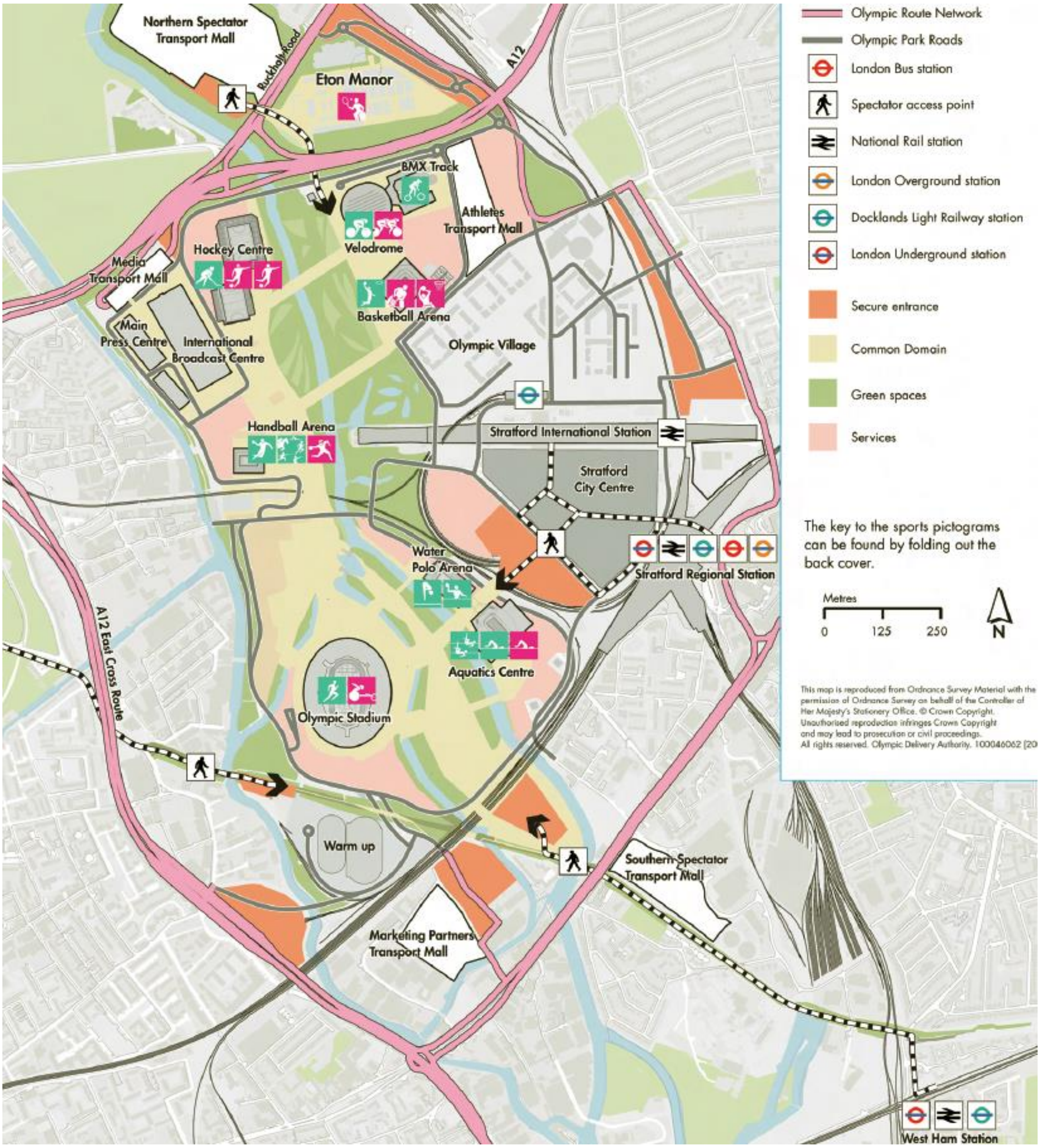


Figure 652 – Example – London2012 Olympics Main Site

The site in the map of the London 2012 Olympic venue shown above is represented in the following figure as a network of POINTs OF INTEREST and STOP PLACES, connected by a PATH LINKs. To get between any two points the links can be navigated in sequence to create a NAVIGATION PATH. Processes that take place at particular points such as security checks or queues can be represented by CHECK CONSTRAINTs.

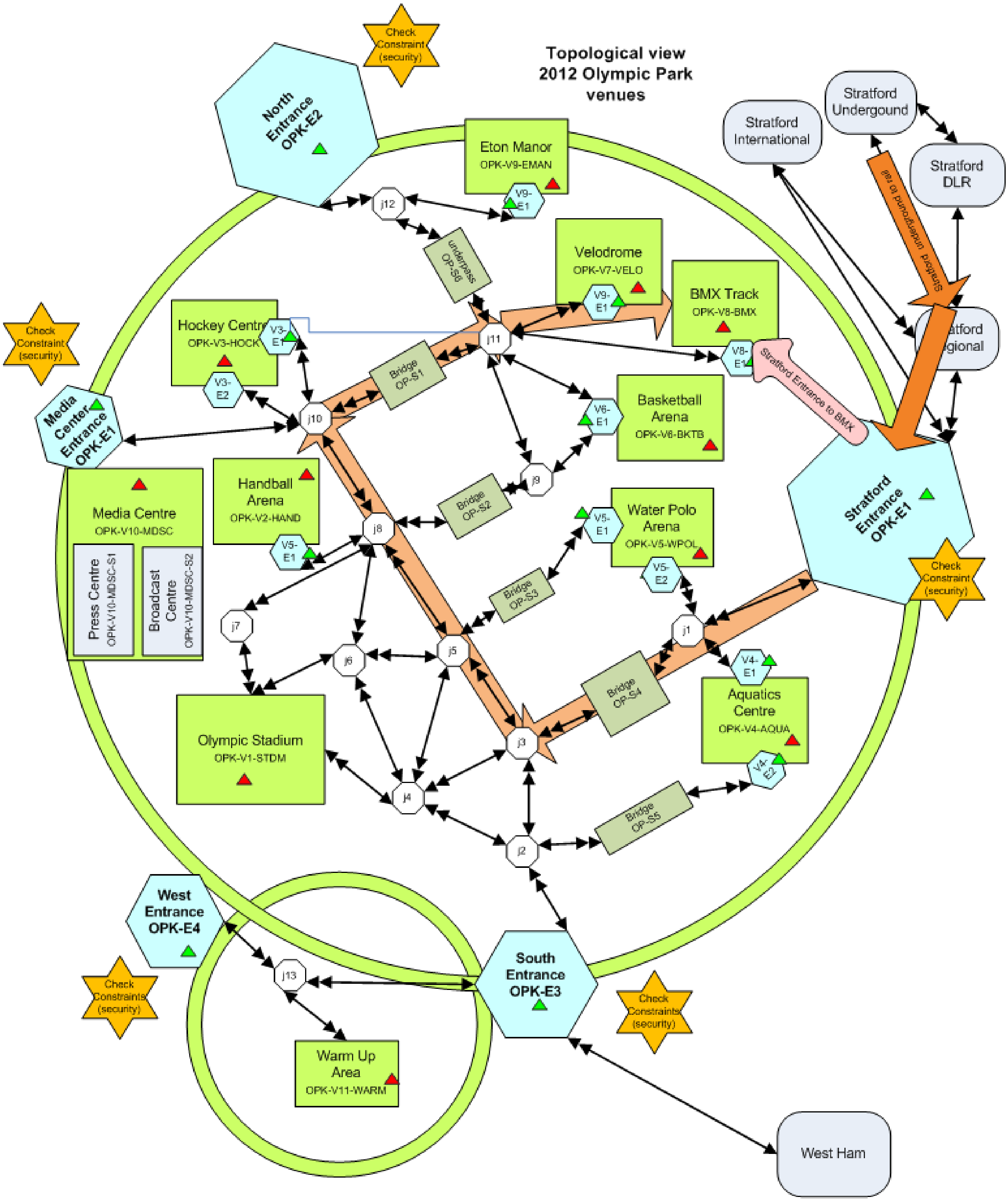


Figure 653 – Example – London2012 Olympic venue Navigation Path

8.4.10.2.4 Creating Navigation paths

In an implementation, NAVIGATION PATHs can either be predefined statically by hand or be computed dynamically from path links by a micro journey planner. Since the number of permutations of point to paths for different accessibility characteristics can be large even for a small station, a computational approach is preferable.

Where NAVIGATION PATHs are created manually it is possible to use NAVIGATION paths at a summary level only, that is not to have detailed PATH LINKs; this at least gives an indication of overall accessibility, albeit without step by step navigation. There may be more than one NAVIGATION PATH between the same two nodes corresponding to different routes.

8.4.10.3 Path Link & Navigation Path direction

A PATH LINK connects any two spaces or PATH JUNCTIONs within a SITE that can be traversed by a passenger. It is also optionally indicating an ENTRANCE if the end point is a QUAY or ACCESS SPACE.

— The same PATH LINK may be reused in many different NAVIGATION PATHs.

— A PATH LINK is **directional** in that it always has a 'from' end and a 'to' end – however it may be used in either direction, unless tagged to indicate it is one way – as say an escalator or one-way subway tunnel might be tagged to indicate that it can only be used in one sense.

— A NAVIGATION PATH references a sequence of PATH LINKs. For each path link, the NAVIGATION PATH indicates whether the use is forwards (i.e. from origin to destination) or backwards (i.e. from destination to origin).

— A NAVIGATION PATH has a single direction from origin to destination.

This is shown in the following figure, where two different NAVIGATION PATHs (*A->D* and *D->A*) use the same three PATH LINKs (*Path Link 1:A->B, Path Link 2 B->C* and *Path Link 3: C->D*) in two different directions. The directionality of the path link is indicated by a double arrowhead on the forward end.

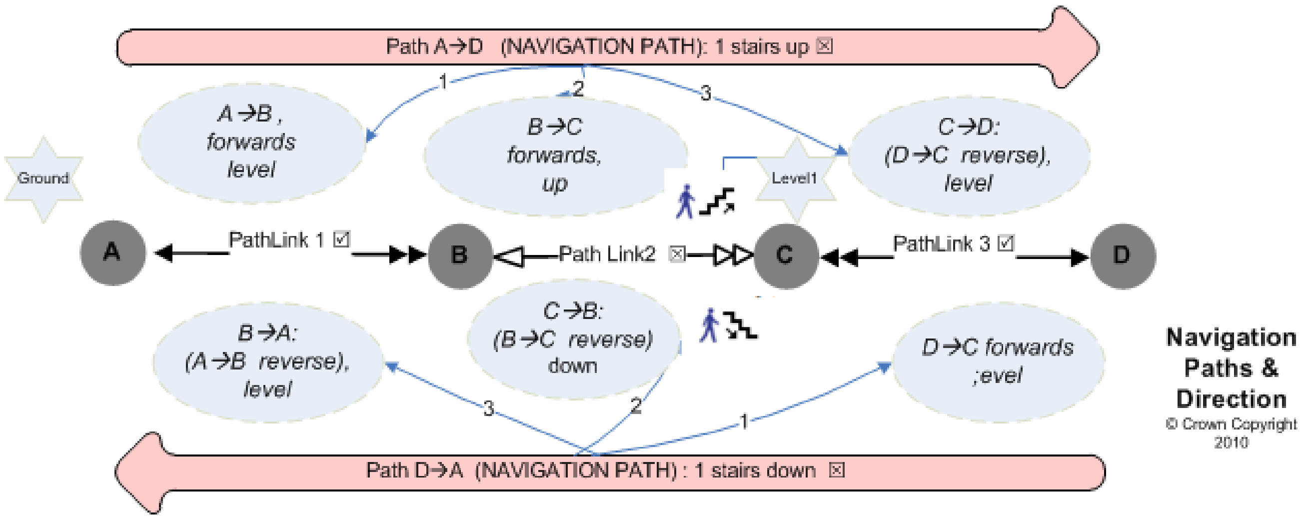


Figure 654 – Exam; if the PLACE ple – Direction of Path Links and Navigation Paths

8.4.10.4 Navigation Path – Physical Model

The following diagram gives an overview the NAVIGATION PATH physical model.

A NAVIGATION PATH connects two PLACEs; it can be viewed as a sequence of PLACEs in SEQUENCE, each identifying a PLACE; the PLACE may be a whole SITE such as a STOP PLACE, or a space within the SITE, such as a STOP PLACE ENTRANCE or a QUAY. A NAVIGATION PATH can also be consiered as a series of PATH LINKs in SEQUENCE, each identifying a PATH LINK. A PATH LINK END may connect either to a PATH JUNCTION, or to a SITE and/or SITE COMPONENT, allowing a specific ENTRANCE to be identified.

Accessibility criteria for the path elements and the overall path can be specified with ACCESSIBILITY ASSESSMENTs. Expected delays at particular times of day can be specified for links and nodes with a CHECK CONSTRAINT.

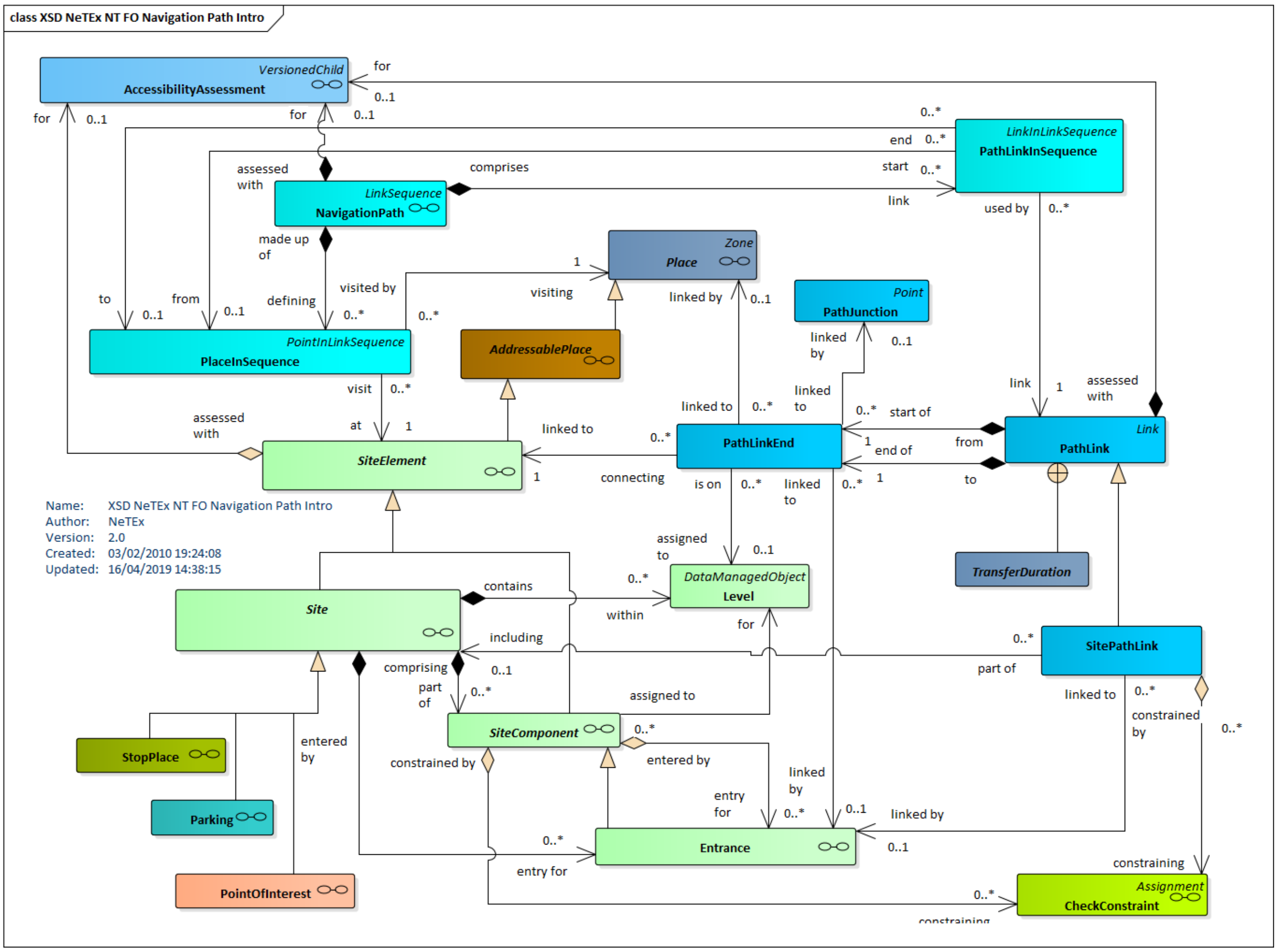


Figure 655 – Navigation Path Overview – Physical Model (UML)

8.4.10.4.1 Navigation Path details – Physical Model

The following diagram shows detailed attributes of the NAVIGATION PATH Physical model.

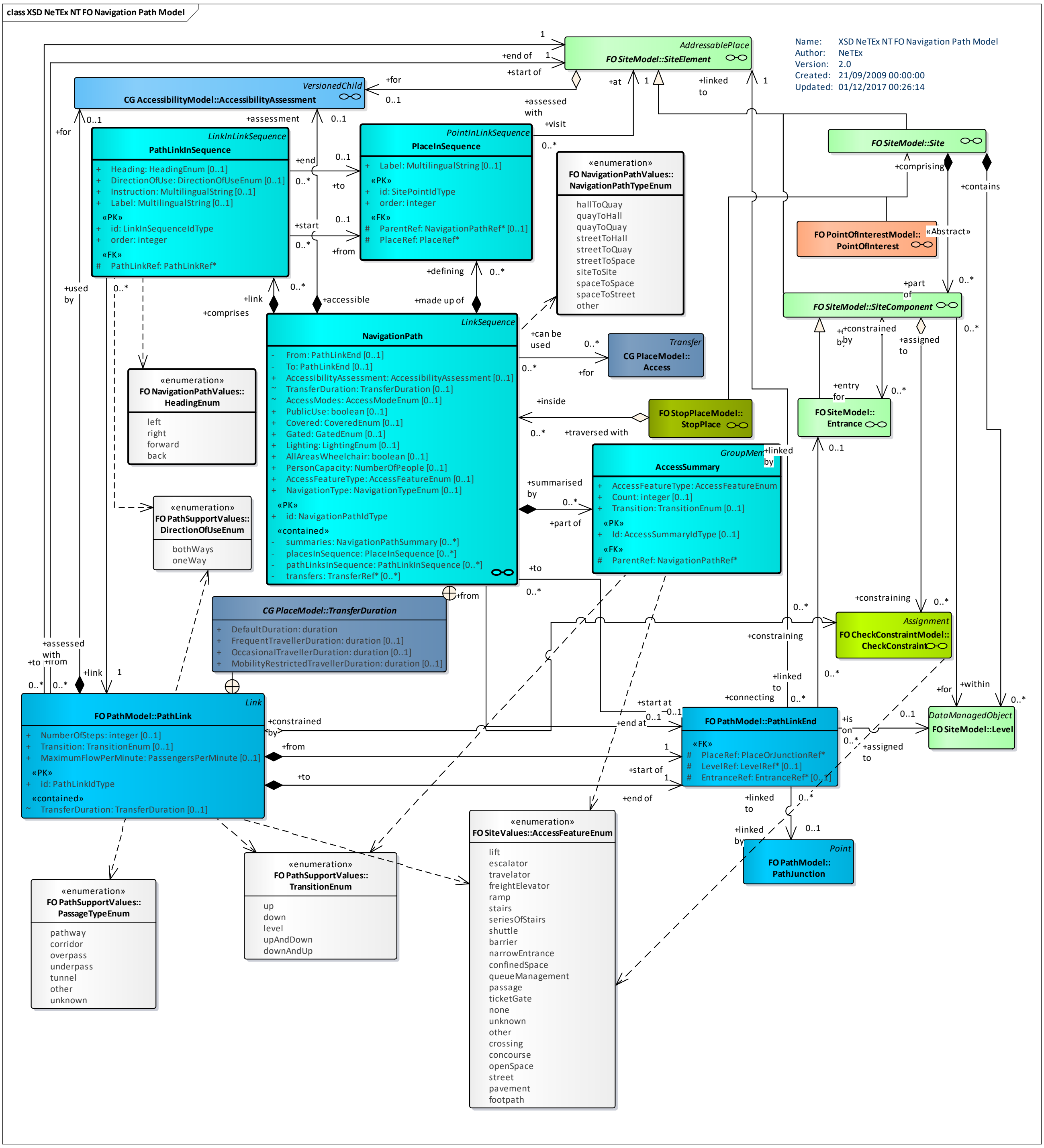


Figure 656 – Navigation Path Details – Physical Model (UML)

8.4.10.4.2 Path Classifications – Physical Model

The following diagram summarises the standardised values that can be used to classify the elements of a NAVIGATION PATH.

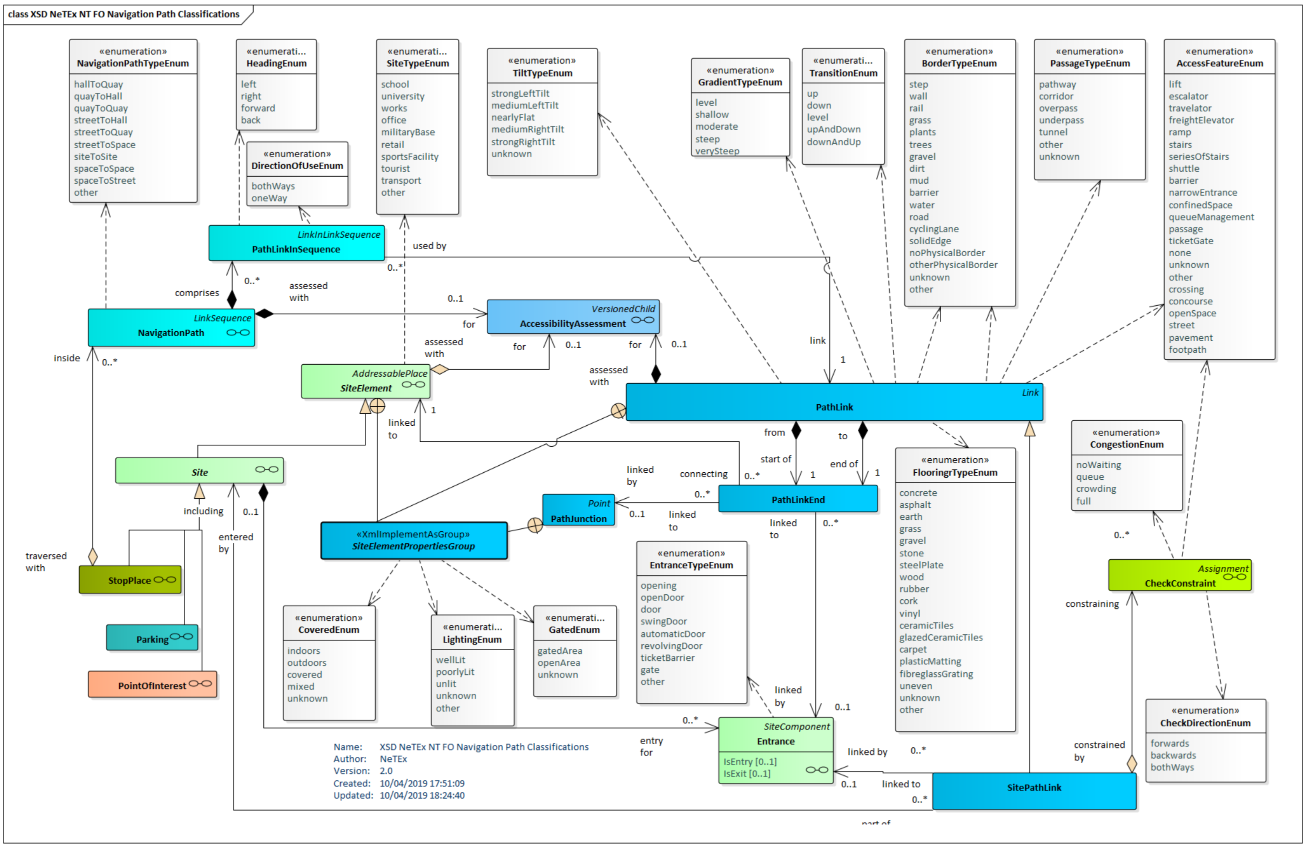


Figure 657 –Path Links Inntroduction – Physical Model (UML)

8.4.10.5 Navigation Path – Attributes and XSD

8.4.10.5.1 NavigationPath – Model Element

A designated path between two places. May include an ordered sequence of PATH LINKs.

Table 560 – *NavigationPath* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LinkSequence*	::>	NAVIGATION PATH inherits from LINK SEQUENCE.
«PK»	***id***	*NavigationPathIdType*	1:1	Identifier of a NAVIGATION PATH.
«cntd»	***From***	*PathLinkEnd*	0:1	Origin end of NAVIGATION PATH. Only needed if detailed PATH LINKs are not given.
«cntd»	***To***	*PathLinkEnd*	0:1	Destination end of NAVIGATION PATH. Only needed if detailed PATH LINKs are not given.
XGRP	***NavigationPath-SummaryGroup***	***xmlGroup***	0:1	SUMMARIEs of NAVIGATION PATH features.
XGRP	***SiteElement-PropertiesGroup***	***xmlGroup***	0:1	Common properties of a SITE ELEMENT. See SITE ELEMENT.

«enum»	***AccessFeature-List***	*AccessFeatureEnum*	0:*	Types of access feature found on path. See ACCESS SPACE for allowed values.
«enum»	***NavigationType***	*NavigationTypeEnum*	0:1	Type of NAVIGATION PATH. See allowed values below.
«cntd»	***placesInSequence***	*PlaceInSequence*	0:*	PLACEs traversed by a NAVIGATION PATH in sequence. Can be derived from the ends of individual PATH LINKs.
«cntd»	***pathLinksIn-Sequence***	*PathLinkInSequence*	0:*	PATH LINKs of a NAVIGATION PATH.
«cntd»	***transfers***	*TransferRef*	0:*	TRANSFERs and ACCESS links associated with a NAVIGATION PATH and for which it describes a detailed path.

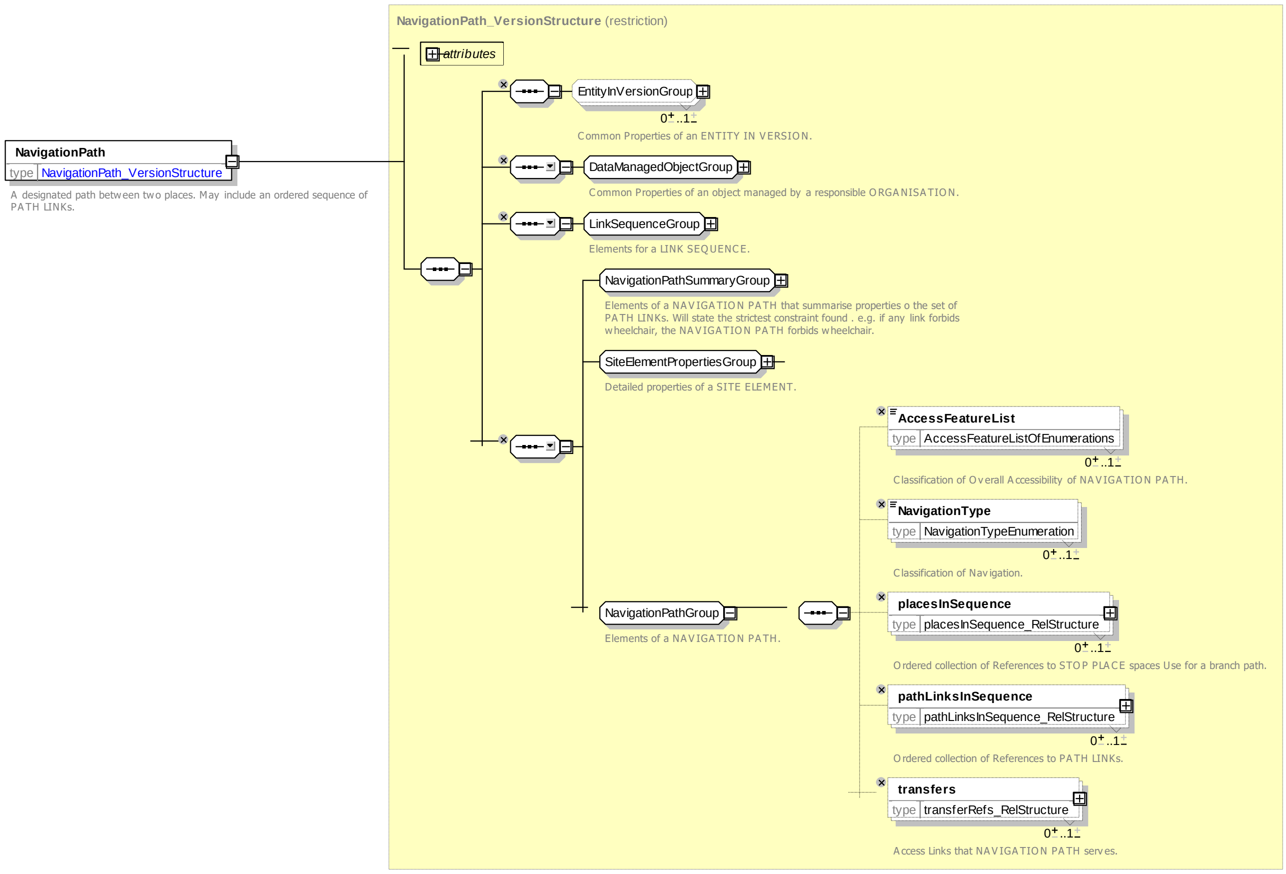


Figure 658 – *NavigationPath – XSD*

8.4.10.5.1.1 NavigationPathType – Allowed values

The following table shows the allowed values for ***NavigationPathType*** ***(****NavigationTypeEnumeration).*

Table 561 – Types of *NavigationPath* – Allowed Values

Navigation Type	Description	Used for

quayToQuay	Quay to Quay	STOP PLACE
streetToQuay	Street to Quay	STOP PLACE
quayToStreet	Quay to Street	STOP PLACE
hallToQuay	Ticket Hall to Quay	STOP PLACE, POI
quayToHall	Quay to Ticket Hall or Entrance Gallery	STOP PLACE, POI
streetToHall	Street to Ticket Hall	STOP PLACE, POI
hallToStreet	Ticket Hall to Street	STOP PLACE, POI
streetToSpace	Street to Spectator Area	POI
spaceToStreet	Spectator Area to Street	POI
spaceToSpace	Spectator Area to Spectator Area	POI
other	other	STOP PLACE, POI

8.4.10.5.1.2 NavigationPathSummaryGroup – XML Group

A designated path between two places. May include an ordered sequence of PATH LINKs.

Table 562 – *NavigationPathSummaryGroup* – Element

Classification	Name	Type	Cardinality	Description
«cntd»	***From***	*PathLinkEnd*	0:1	Origin end of NAVIGATION PATH. Only needed if detailed PATH LINKs are not given.
«cntd»	***To***	*PathLinkEnd*	0:1	Destination end of NAVIGATION PATH. Only needed if detailed PATH LINKs are not given.
«cntd»	***Accessibility-Assessment***	*AccessibilityAssessment*	0:1	ACCESSIBILITY ASSESSMENT associated with NAVIGATION PATH.
«enum»	***AccessModes***	*AccessModeEnum*	0:1	Access modes that can be used to traverse a NAVIGATION PATH. See TRANSPORT MODE for allowed values.
«cntd»	***summaries***	*AccessSummary*	0:*	Summary of a feature used in NAVIGATION PATH, used by journey planners to indicate accessibility, e.g. “Stairs with 25 steps up,”.
«cntd»	***TransferDuration***	*TransferDuration*	0:1	Time it takes to traverse a NAVIGATION PATH. May be derived from summing individual PATH LINK durations.

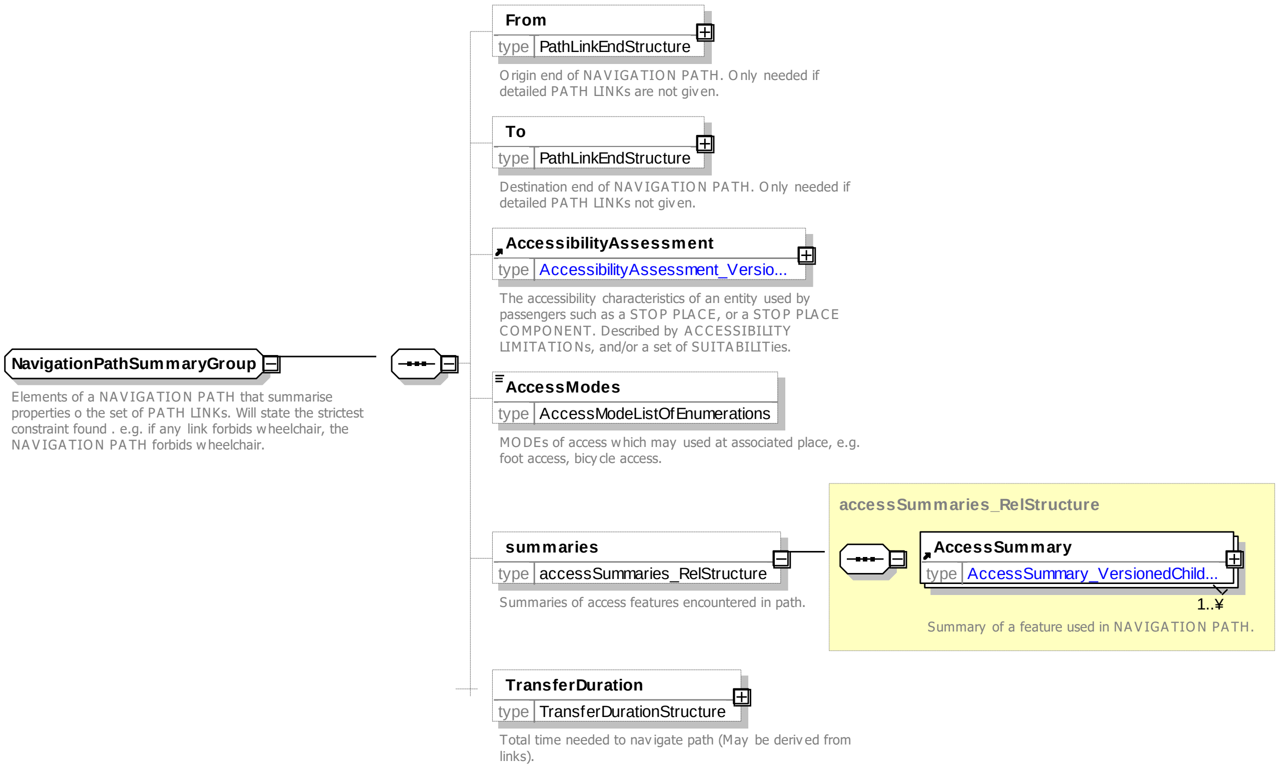


Figure 659 – *NavigationPathSummaryGroup – XSD*

8.4.10.5.2 AccessSummary – Model Element

A summary view of the accessibility properties (Number of lifts etc.) found in a NAVIGATION PATH.

Can be used by Journey planners, etc., to give a summary of the accesibility characteristics. For example "*Stairs, 35 steps up*".

Table 563 – *AccessSummary* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*VersionedChild*	*::>*	ACCESS SUMMARY inherits from VERSIONED CHILD.
«PK»	***id***	*NavigationPathSummary-IdType*	0:1	Identifier of Summary of NAVIGATION PATH.
«enum»	***AccessFeature-Type***	*AccessFeatureEnum*	1:1	Type of an ACCESS FEATURE. See ACCESS SPACE for allowed values.
	Count	*xsd:integer*	1:1	Number of instances of feature.
«enum»	***Transition***	*TransitionEnum*	0:1	Transition made when using ACCESS FEATURE. See PATH LINK for allowed values.
«FK»	***ParentRef***	*NavigationPathRef*	0:1	Parent NAVIGATION PATH of summary.

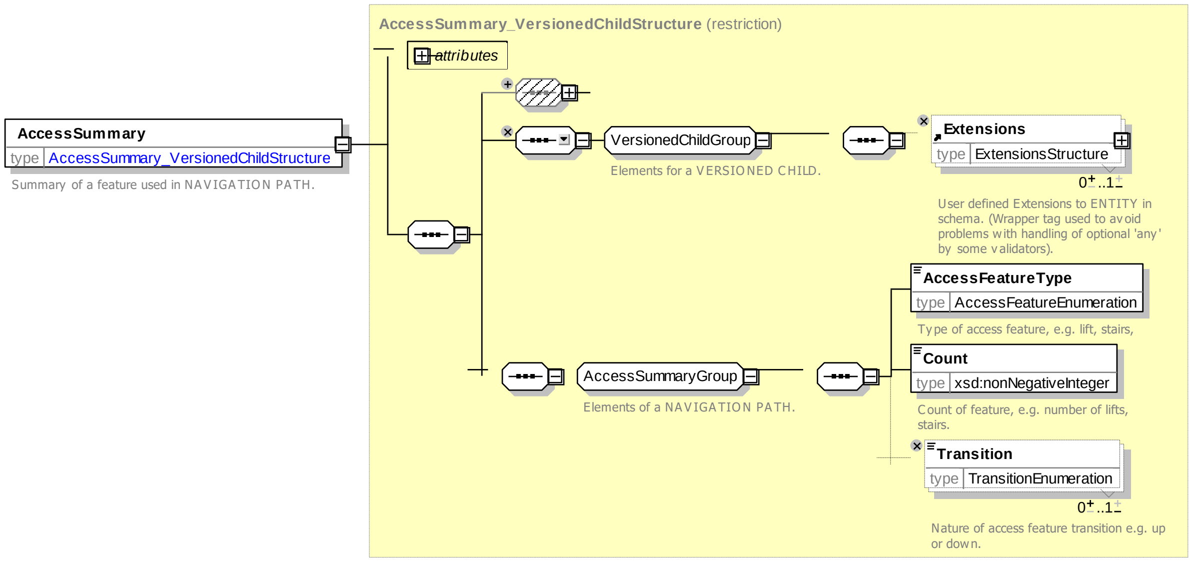


Figure 660 – *AccessSummary – XSD*

8.4.10.5.3 PlaceInSequence – Model Subelement

PLACE, PATH JUNCTION or POINT building up a NAVIGATION PATH within an ordered set of other PLACE IN SEQUENCE. May also be the beginning or end of a PATH LINK IN SEQUENCE.

Table 564 – *PlaceInSequence* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*PointInLinkSequence*	::>	SITE POINT IN SEQUENCE inherits from POINT IN LINK SEQUENCE.
«PK»	***id***	*SitePointIdType*	1:1	Identifier of SITE POINT IN SEQUENCE.
«atr», «PK»	***order***	*xsd:integer*	1:1	Order of SITE POINT IN SEQUENCE within NAVIGATION PATH.
	Label	*MultilingualString*	0:1	Label of SITE POINT IN SEQUENCE.
«FK»	***PlaceRef***	*(PlaceRef)*	0:1	Reference to a PLACE or SITE that this SITE POINT IN SEQUENCE belongs to.
	BranchLevel	*xsd:NMTOKEN*	0:1	Branch nesting number for branched paths.
	Description	*MultilingualString*	0:1	Description of PLACE IN SEQUENCE.
«cntd»	***onwardLinks***	*PathLinkInSequence*	0:*	Next links in path, may be more than one if branch point.

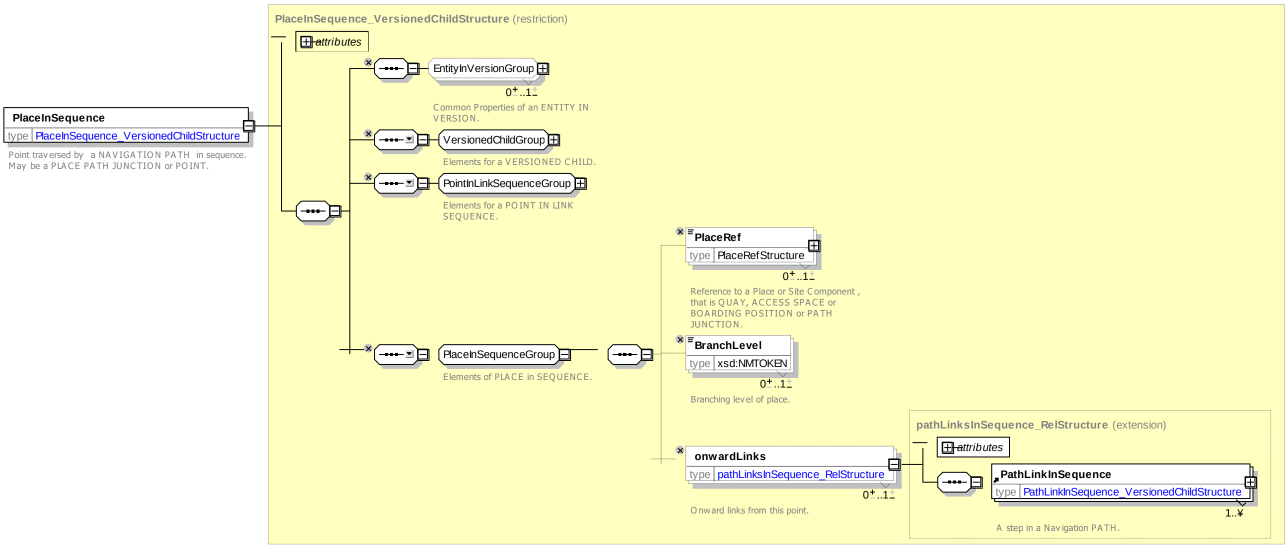


Figure 661 – *PlaceInSequence – XSD*

8.4.10.5.4 PathLinkInSequence – Model Subelement

A step of a NAVIGATION PATH indicating traversal of a particular PATH LINK as part of a recommended route.

The same PATH LINK may occur in different sequences in different NAVIGATION PATHs.

Table 565 – *PathLinkInSequence* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	PATH LINK IN SEQUENCE inherits from versioned child
«PK»	***id***	*LinkInSequenceIdType*	1:1	Identifier of PATH LINK IN SEQUENCE.
«atr», «PK»	***order***	*xsd:integer*	1:1	Order of PATH LINK IN SEQUENCE within LINK SEQUENCE.
«enum»	***Heading***	*HeadingEnum*	0:1	Heading instruction relative to point declared *'left'*, *'right'*, '*onwards'*, etc. See allowed values below.
«enum»	***DirectionOfUse***	*DirectionOfUseEnum*	0:1	Permitted direction of travel. See PATH LINK for allowed values.
	Label	*MultilingualString*	0:1	Label of PATH LINK IN SEQUENCE.
«FK»	***PathLinkRef***	*PathLinkRef*	1:1	Reference to a PATH LINK.

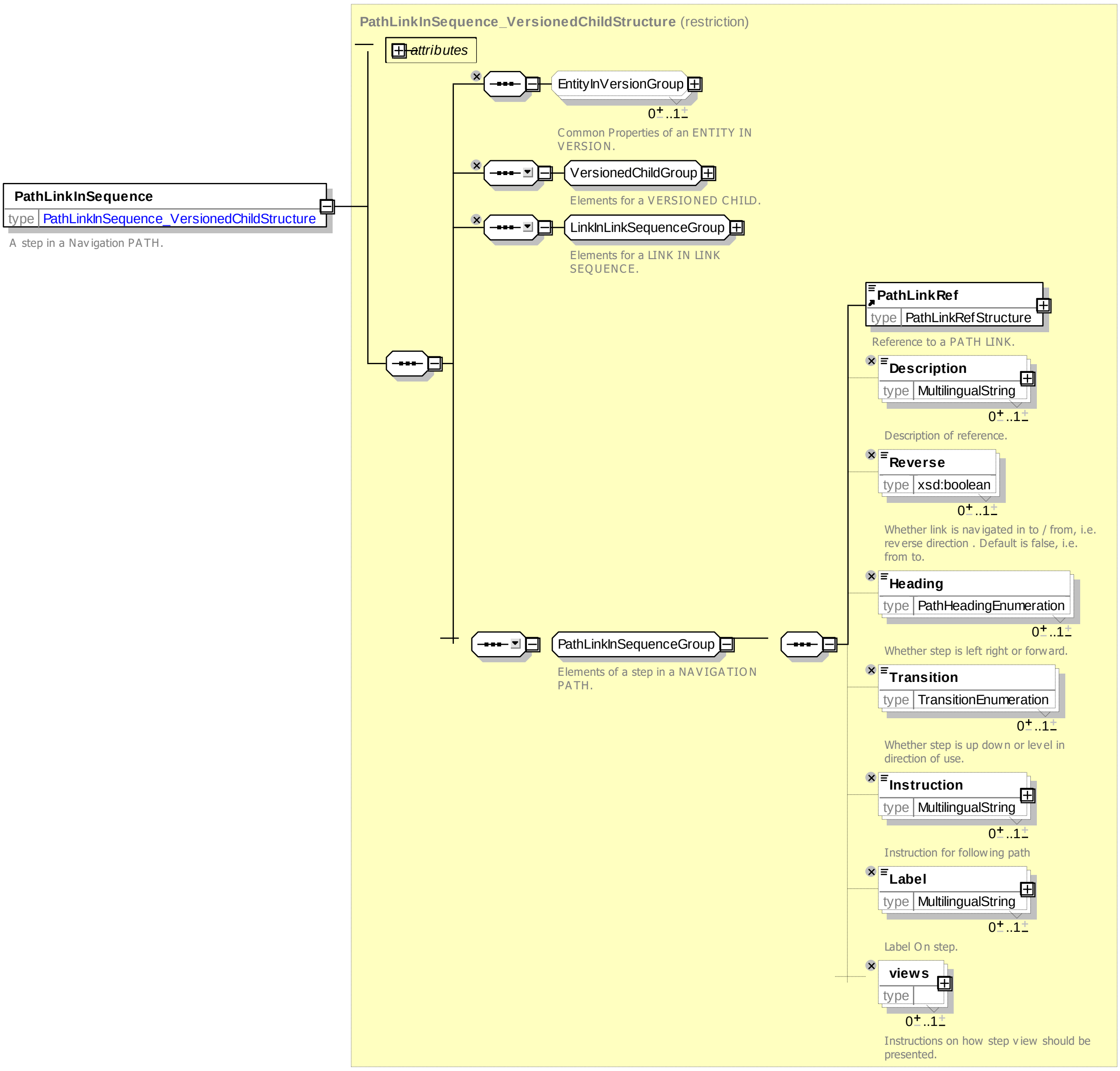


Figure 662 – *PathLinkInSequence – XSD*

8.4.10.5.4.1 HeadingType – Allowed values

The following table shows the allowed values for ***HeadingType (****HeadingTypeEnum****)***

Table 566 – *HeadingType* – Allowed values

Value	Description
left	Left turn
right	Right turn
forward	Straight on
back	Backwards

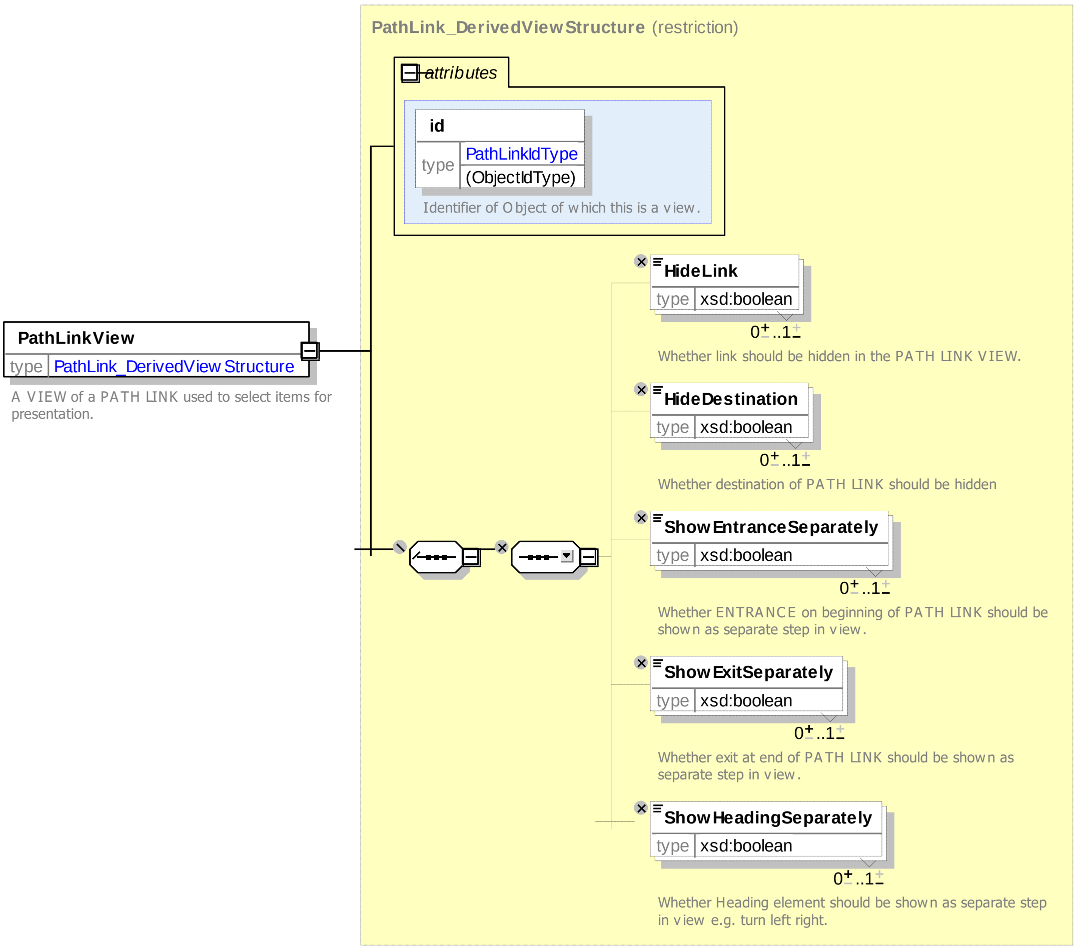


Figure 663 – *PathLinkView – XSD*

8.4.10.6 Navigation Path –XML Examples

8.4.10.6.1 XML example of a Navigation Path

The following XML code fragment shows a NAVIGATION PATH from hall to quay using a lift (Platform 5 to Platform 6 in the Wimbledon example). It references six PATH LINKs (shown in the subsequent XML fragment below). The NAVIGATION PATH has overall accessibility attributes based on the properties of individual links.

8.4.10.6.2 XML fragment for Navigation Path

EXAMPLE XML Example of Navigation Path

```
<NavigationPath created="2010-05-17T09:30:47Z" modification="new" id="tbd:9100WIMBLDN_A1-EE1_to_5n6-acc">
    <AccessibilityAssessment created="2010-05-17T09:30:47Z">
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation>
         id="tbd:9100WIMBLDN_A1-EE1_to_5n6-acc_01">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>false</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
```

```
    </AccessibilityAssessment>
    <features>
        <AccessSummary>
            <AccessFeatureType>lift</AccessFeatureType>
            <Count>1</Count>
            <Transition>down</Transition>
        </AccessSummary>
    </features>
    <Name>Street to Platform 5 and 6 - Accessible</Name>
    <TypeOfNavigation>hallToQuay</TypeOfNavigation>
    <pathLinksInSequence>
        <PathLinkInSequence order="1">
            <PathLinkRef>tbd:9100WIMBLDN_lnk_A2-EI1_A2-J2</PathLinkRef>
            <Description>From Upper Concourse Main Entrance to Path Junction 2</Description>
            <Reverse>false</Reverse>
            <Transition>level</Transition>
        </PathLinkInSequence>
        <PathLinkInSequence order="2">
            <PathLinkRef>tbd:9100WIMBLDN_lnk_A2b-EI5_A2-J2</PathLinkRef>
            <Description>From Upper Concourse Internal Entrance 5 to Path Junction 2 in upper
concourse</Description>
            <Reverse>true</Reverse>
            <Heading>right</Heading>
            <Transition>level</Transition>
        </PathLinkInSequence>
        <PathLinkInSequence order="3">
            <PathLinkRef>tbd:9100WIMBLDN_lnk_ A2b-EI5_A2b-J5</PathLinkRef>
            <Description>From Upper Concourse Lift area Internal Entrance 5 to Path Junction 5 in lift
area </Description>
            <Reverse>false</Reverse>
            <Transition>level</Transition>
        </PathLinkInSequence>
        <PathLinkInSequence order="4">
            <PathLinkRef>tbd:9100WIMBLDN_lnk_A2b-EL2g_A2b-J5</PathLinkRef>
            <Description>From Upper Concourse Lift Entrance 2 to Path Junction 5 in lift
area</Description>
            <Reverse>true</Reverse>
            <Heading>left</Heading>
            <Transition>level</Transition>
        </PathLinkInSequence>
        <PathLinkInSequence order="5">
            <PathLinkRef>tbd:9100WIMBLDN_lnk_A2b-EL2g_5n6-EL1_by-L2</PathLinkRef>
            <Description>From Upper Concourse to platform 5 and 6 by Lift </Description>
            <Reverse>false</Reverse>
            <Transition>down</Transition>
        </PathLinkInSequence>
    </pathLinksInSequence>
</NavigationPath>
```

8.4.10.6.3 XML fragment for Path Links used in Navigation Path

The following XML code fragment shows the single PATH JUNCTION and six PATH LINKs referenced by the NAVIGATION PATH above.

EXAMPLE XML Example of Navigation Path PathLinks

```
<pathJunctions>
    <PathJunction created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_A2_J2">
        <Name>Branch from main entrance to Rail stairs to 5 and 6</Name>
```

```
            <Centroid>
                <Location>
                    <Longitude>-180</Longitude>
                    <Latitude>-90</Latitude>
                </Location>
            </Centroid>
            <ParentZoneRef>tbd:9100WIMBLDN_A2</ParentZoneRef>
            <Covered>indoors</Covered>
        </PathJunction>
    </pathJunctions>
    <pathLinks>

        <SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A1-EI1_A2-J2">
            <Name>From Upper Concourse Internal Entrance 1 to lift area to Path Junction 2 in Upper
    concourse</Name>
            <AccessibilityAssessment>
                <MobilityImpairedAccess>true</MobilityImpairedAccess>
                <limitations>
                    <AccessibilityLimitation
                        <WheelchairAccess>true</WheelchairAccess>
                        <StepFreeAccess>true</StepFreeAccess>
                        <EscalatorFreeAccess>true</EscalatorFreeAccess>
                        <LiftFreeAccess>true</LiftFreeAccess>
                    </AccessibilityLimitation>
                </limitations>
            </AccessibilityAssessment>
            <Covered>indoors</Covered>
            <From>
                <PlaceRef>tbd:9100WIMBLDN_A2</PlaceRef>
                <EntranceRef>tbd:9100WIMBLDN_A2b-EIb2</EntranceRef>
            </From>
            <To>
                <PlaceRef>tbd:9100WIMBLDN_A2_J2</PlaceRef>
            </To>
            <Distance>5.00</Distance>
            <NumberOfSteps>0</NumberOfSteps>
            <AllowedUse>twoWay</AllowedUse>
            <FromToUpDown>level</FromToUpDown>
            <TransferDuration>
                <DefaultDuration>PT30S</DefaultDuration>
            </TransferDuration>
            <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
            <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
        </SitePathLink>
        <SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A2b-EIb1_A2-J2">
            <Name>From Upper Concourse Lift area Entrance 5 to Path Junction 2</Name>
            <AccessibilityAssessment>
                <MobilityImpairedAccess>true</MobilityImpairedAccess>
                <limitations>
                    <AccessibilityLimitation
                        <WheelchairAccess>true</WheelchairAccess>
                        <StepFreeAccess>true</StepFreeAccess>
                        <EscalatorFreeAccess>true</EscalatorFreeAccess>
                        <LiftFreeAccess>true</LiftFreeAccess>
                    </AccessibilityLimitation>
                </limitations>
            </AccessibilityAssessment>
            <Covered>indoors</Covered>
            <From>
```

```xml
        <PlaceRef>tbd:9100WIMBLDN_A2b</PlaceRef>
        <EntranceRef>tbd:9100WIMBLDN_A2b-EI5</EntranceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_A2_J2</PlaceRef>
    </To>
    <Distance>5.00</Distance>
    <NumberOfSteps>0</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>level</FromToUpDown>
    <TransferDuration>
        <DefaultDuration>PT30S</DefaultDuration>
    </TransferDuration>
    <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
    <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
</SitePathLink>
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A2b-EI5_A2b-J5">
    <Name>From Upper Concourse Lift area Entrance 5 to Lift area Path Junction 5</Name>
    <AccessibilityAssessment>
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <Covered>indoors</Covered>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_A2b</PlaceRef>
        <EntranceRef>tbd:9100WIMBLDN_A2b-EIb2</EntranceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_A2b_J5</PlaceRef>
    </To>
    <Distance>5.00</Distance>
    <NumberOfSteps>0</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>level</FromToUpDown>
    <TransferDuration>
        <DefaultDuration>PT30S</DefaultDuration>
    </TransferDuration>
    <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
    <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
</SitePathLink>
<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A2b-EL2g_5n6-EL1_by-L2">
    <Name>From Upper Concourse Lift Area to Platforms 5 and 6 by lift 2</Name>
    <AccessibilityAssessment>
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation created="2010-05-17T09:30:47Z">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>false</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
```

```xml
    </AccessibilityAssessment>
    <From>
        <PlaceRef>tbd:9100WIMBLDN_L2</PlaceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
        <EntranceRef>tbd:9100WIMBLDN_A2b-EL2g</EntranceRef>
    </From>
    <To>
        <PlaceRef>tbd:9100WIMBLDN_L2</PlaceRef>
        <LevelRef>tbd:9100WIMBLDN_Lvl_PL</LevelRef>
        <EntranceRef>tbd:9100WIMBLDN_5n6_EL1pl</EntranceRef>
    </To>
    <Distance>0</Distance>
    <NumberOfSteps>0</NumberOfSteps>
    <AllowedUse>twoWay</AllowedUse>
    <FromToUpDown>down</FromToUpDown>
    <AccessFeatureType>lift</AccessFeatureType>
    <TransferDuration>
        <DefaultDuration>PT3M</DefaultDuration>
        <FrequentTravellerDuration>PT5M</FrequentTravellerDuration>
        <OccasionalTravellerDuration>PT5M</OccasionalTravellerDuration>
        <MobilityRestrictedTravellerDuration>PT10M</MobilityRestrictedTravellerDuration>
    </TransferDuration>
    <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
    <checks>
        <CheckConstraint>
     id="tbd:9100WIMBLDN_lnk_A2b-EL2g_5n6-EL1_by-L2_C1">
            <validityConditions>
                <AvailabilityCondition id="tbd:Av_openingHrs01">
                    <Description>Opening hours for Station</Description>
                </AvailabilityCondition>
            </validityConditions>
            <CheckProcess>none</CheckProcess>
            <CheckService>selfserviceMachine</CheckService>
            <AccessFeatureType>lift</AccessFeatureType>
            <Congestion>queue</Congestion>
            <MinimumLikelyDelay>P1Y2M3DT10H30M</MinimumLikelyDelay>
            <AverageDelay>P1Y2M3DT10H30M</AverageDelay>
            <MaximumLikelyDelay>P1Y2M3DT10H30M</MaximumLikelyDelay>
        </Check>
    </checks>
    <placeEquipments>
        <LiftEquipment id="tbd:9100WIMBLDN_A2b_L2">
            <Name>Lift to Platforms 5 and 6</Name>
            <Width>1.5</Width>
            <WheelchairTurningCircle>1</WheelchairTurningCircle>
            <ThroughLoader>false</ThroughLoader>
            <Automatic>true</Automatic>
        </LiftEquipment>
    </placeEquipments>
</SitePathLink>

<SitePathLink created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_lnk_A2b-EL2g_A2b-J5">
    <Name>From Upper Concourse Lift Entrance 2 to Lift Area Path Junction 5</Name>
    <AccessibilityAssessment>
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
```

```
                    <EscalatorFreeAccess>true</EscalatorFreeAccess>
                    <LiftFreeAccess>true</LiftFreeAccess>
                </AccessibilityLimitation>
            </limitations>
        </AccessibilityAssessment>
        <Covered>indoors</Covered>
        <From>
            <PlaceRef>tbd:9100WIMBLDN_A2b</PlaceRef>
            <EntranceRef>tbd:9100WIMBLDN_A2b-EL2g</EntranceRef>
        </From>
        <To>
            <PlaceRef>tbd:9100WIMBLDN_A2b_J5</PlaceRef>
        </To>
        <Distance>5.00</Distance>
        <NumberOfSteps>0</NumberOfSteps>
        <AllowedUse>twoWay</AllowedUse>
        <FromToUpDown>level</FromToUpDown>
        <AccessFeatureType>confinedSpace</AccessFeatureType>
        <TransferDuration>
            <DefaultDuration>PT30S</DefaultDuration>
        </TransferDuration>
        <MaximumFlowPerMinute>200</MaximumFlowPerMinute>
        <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
    </SitePathLink>
</pathLinks>
```

8.4.10.6.4 XML example of a Navigation Path Summary for Stairs

The following XML code fragment shows a staircase represented just in summary on a NAVIGATION PATH – the total number of stairs is mentioned but no other detail.

EXAMPLE XML Example of Navigation Path Summary

```
<NavigationPath created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_A1-EE1_to_DL-notacc">
    <AccessibilityAssessment created="2010-05-17T09:30:47Z">
        <MobilityImpairedAccess>false</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation created="2010-05-17T09:30:47Z" id="tbd:9100WIMBLDN_A1-EE1_to_DL-
notacc-01">
                <WheelchairAccess>false</WheelchairAccess>
                <StepFreeAccess>false</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <features>
        <AccessSummary>
            <AccessFeatureType>stairs</AccessFeatureType>
            <Count>26</Count>
            <Transition> down</Transition>
        </AccessSummary>
    </features>
    <Name>Street to District Line - By Stairs</Name>
    <TypeOfNavigation>streetToQuay</TypeOfNavigation>
</NavigationPath>
```

8.4.10.6.5 XML example of a Branched Navigation Path

The following XML code fragment shows a branched path that has both Stair and Lift alternatives for the last part.

EXAMPLE XML Example of Branched Path

```
<NavigationPath created="2010-05-17T09:30:47Z" id="tbd:490G00272P_ST-StopD_to_DL">
    <AccessibilityAssessment dataSourceRef="XtraAccessibilityData">
        <MobilityImpairedAccess>true</MobilityImpairedAccess>
        <limitations>
            <AccessibilityLimitation id="tbd:490G00272P_ST-StopD_to_DL_01">
                <WheelchairAccess>true</WheelchairAccess>
                <StepFreeAccess>true</StepFreeAccess>
                <EscalatorFreeAccess>true</EscalatorFreeAccess>
                <LiftFreeAccess>true</LiftFreeAccess>
            </AccessibilityLimitation>
        </limitations>
    </AccessibilityAssessment>
    <Name>Ticket Hall District line- Branched by stairs</Name>
    <TypeOfNavigation>hallToQuay</TypeOfNavigation>
    <placeRefsInSequence>
        <PlaceRefInSequence order="1">
            <PlaceRef>tbd:9100WIMBLDN_A1</PlaceRef>
            <BranchLevel>1</BranchLevel>
            <Description>Ticket Hall </Description>
            <onwardLinks>
                <PathLinkInSequence order="5">
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A1-EI1_A2-j2</PathLinkRef>
                    <Description>From Upper concourse internal entrance to junction 2</Description>
                    <Reverse>false</Reverse>
                    <Heading>forward</Heading>
                </PathLinkInSequence>
            </onwardLinks>
        </PlaceRefInSequence>
        <PlaceRefInSequence order="2">
            <PlaceRef>tbd:9100WIMBLDN_A2-j2</PlaceRef>
            <BranchLevel>1</BranchLevel>
            <Description>Path Junction j2 in Upper Concourse Ticket Hall (BRANCH POINT)</Description>
            <onwardLinks>
                <PathLinkInSequence order="1">
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A2-j2-A2_j1</PathLinkRef>
                    <Description>From Upper concourse junction 2 in path </Description>
                    <Reverse>true</Reverse>
                </PathLinkInSequence>
                <PathLinkInSequence order="1">
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A2b_EI1-A2-j2</PathLinkRef>
                    <Description>From Entrance1 to Lift Area in Upper concourse to junction 2 in Upper
concourse</Description>
                    <Reverse>true</Reverse>
                </PathLinkInSequence>
            </onwardLinks>
        </PlaceRefInSequence>
        <PlaceRefInSequence order="2">
            <PlaceRef>tbd:9100WIMBLDN_A2-j1</PlaceRef>
            <BranchLevel>3.1</BranchLevel>
            <Description>Path Junction j1 in Upper Concourse Ticket Hall </Description>
            <onwardLinks>
                <PathLinkInSequence order="1">
```

```
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A2-j1-A2_ES1g</PathLinkRef>
                    <Description>From Entrance to stairs 1 in Upper concourse to junction 1 in Upper
concourse</Description>
                    <Reverse>true</Reverse>
                </PathLinkInSequence>
            </onwardLinks>
        </PlaceRefInSequence>
        <PlaceRefInSequence order="2">
            <PlaceRef>tbd:9100WIMBLDN_S1</PlaceRef>
            <BranchLevel>3.1</BranchLevel>
            <Description>Stairs 1</Description>
            <onwardLinks>
                <PathLinkInSequence order="2">
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A2-ES2g-5n6_ES1pl</PathLinkRef>
                    <Description>From Upper Concourse Entrance to Stairs 1 to District Line Concourse
Entrance to stairs by stairs</Description>
                    <Reverse>false</Reverse>
                    <Transition>down</Transition>
                </PathLinkInSequence>
            </onwardLinks>
        </PlaceRefInSequence>
        <PlaceRefInSequence order="2">
            <PlaceRef>tbd:9100WIMBLDN_A2b-j1</PlaceRef>
            <BranchLevel>3.1</BranchLevel>
            <Description>Upper Concourse Lift Area path junction 1</Description>
            <onwardLinks>
                <PathLinkInSequence order="3">
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A2b-EL2_A2b-EI2</PathLinkRef>
                    <Description>From Lift Entrance 1 in Lift Area in Upper concourse to path junction
1 in Lift area in Upper concourse </Description>
                    <Reverse>false</Reverse>
                    <Heading>forward</Heading>
                </PathLinkInSequence>
            </onwardLinks>
        </PlaceRefInSequence>
        <PlaceRefInSequence order="3">
            <PlaceRef>tbd:9100WIMBLDN_L2</PlaceRef>
            <BranchLevel>3.1</BranchLevel>
            <Description> Lift 2 to platform 5 and 6</Description>
            <onwardLinks>
                <PathLinkInSequence order="4">
                    <PathLinkRef>tbd:9100WIMBLDN_lnk_A2b-EL1g_A5-EL1pl</PathLinkRef>
                    <Description>From Lift AREA Lift 1 to Platform 5 and 6 Lift1 Entrance</Description>
                    <Reverse>false</Reverse>
                    <Transition>down</Transition>
                </PathLinkInSequence>
            </onwardLinks>
        </PlaceRefInSequence>
        <PlaceRefInSequence order="7">
            <PlaceRef>tbd:9100WIMBLDN_DL</PlaceRef>
            <BranchLevel>1</BranchLevel>
            <Description>Platform 1</Description>
        </PlaceRefInSequence>
    </placeRefsInSequence>
</NavigationPath>
```

8.4.11 Check Constraint

8.4.11.1 CHECK CONSTRAINT – Conceptual MODEL

There may be points in the STOP PLACE or SITE that incur significant delays either always or at certain times of day – for example, to buy a ticket, pass through a ticket barrier or security check, or immigration control. *NeTEx* allows one or more CHECK CONSTRAINTs to be associated with STOP PLACE COMPONENTs, each specifying a process type and a delay. There may be different CHECK CONSTRAINT DELAYs for different times of day.

One can also specify a VALIDITY CONDITION for when it applies (e.g. *ticket machine queue delays* 5-10 minutes, *8:30-9:30 am*). These can be used to give more realistic journey times and to warn users of potential bottlenecks of which they might not be aware, (for example trying to buy a TfL ticket at a major station in rush hour).

If more than one CHECK CONSTRAINT is valid at a given time, an order of precedence can be specified.

A CHECK CONSTRAINT associated with a PATH LINK by default applies in the directions specified for it (i.e. one way or two way). It may be further restricted to apply only in a given sense of the link.

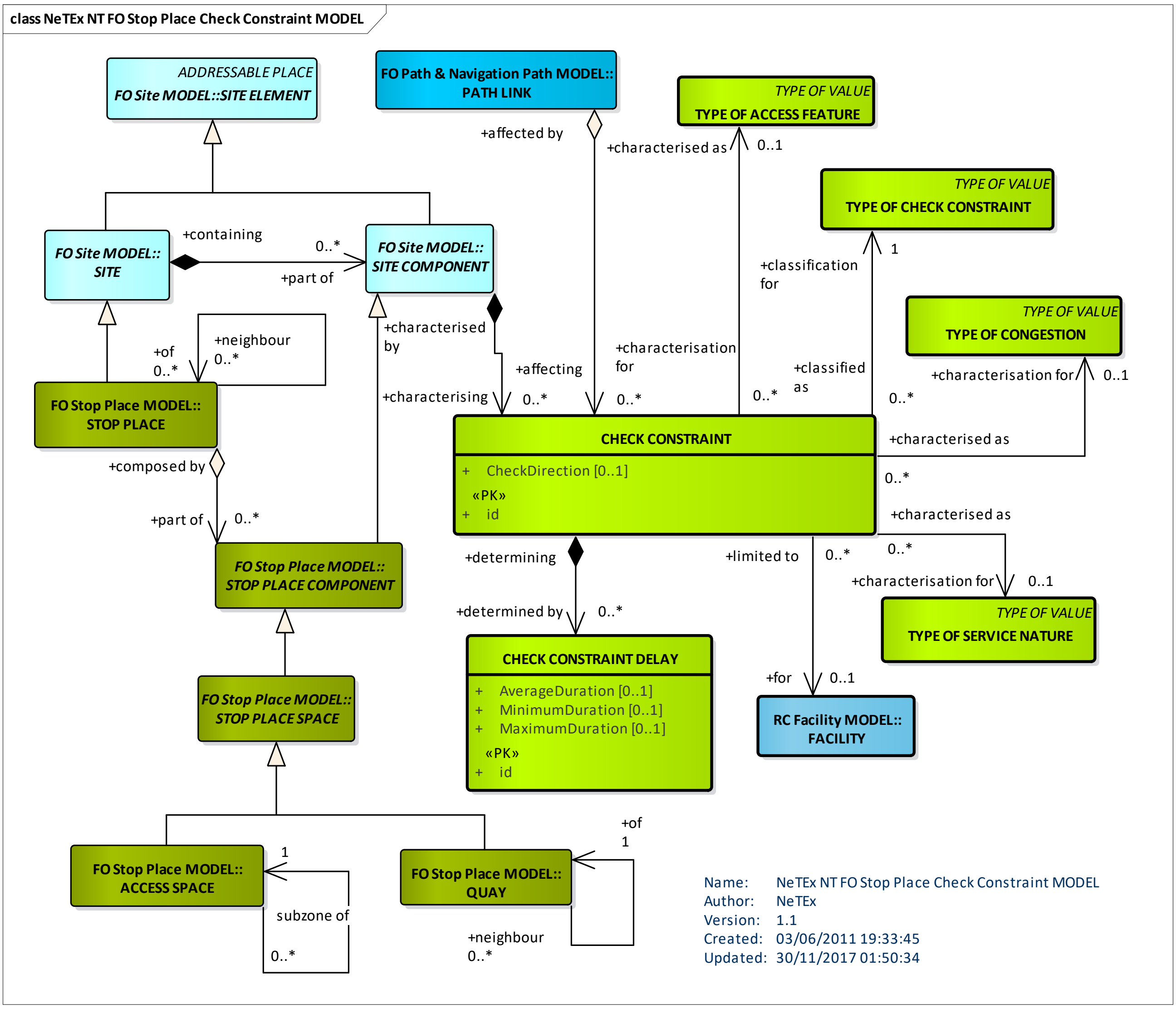


Figure 664 – CHECK CONSTRAINT – Conceptual MODEL – (UML)

8.4.11.2 Check Constraint – Physical Model

The following diagram shows detailed attributes of the CHECK CONSTRAINT model.

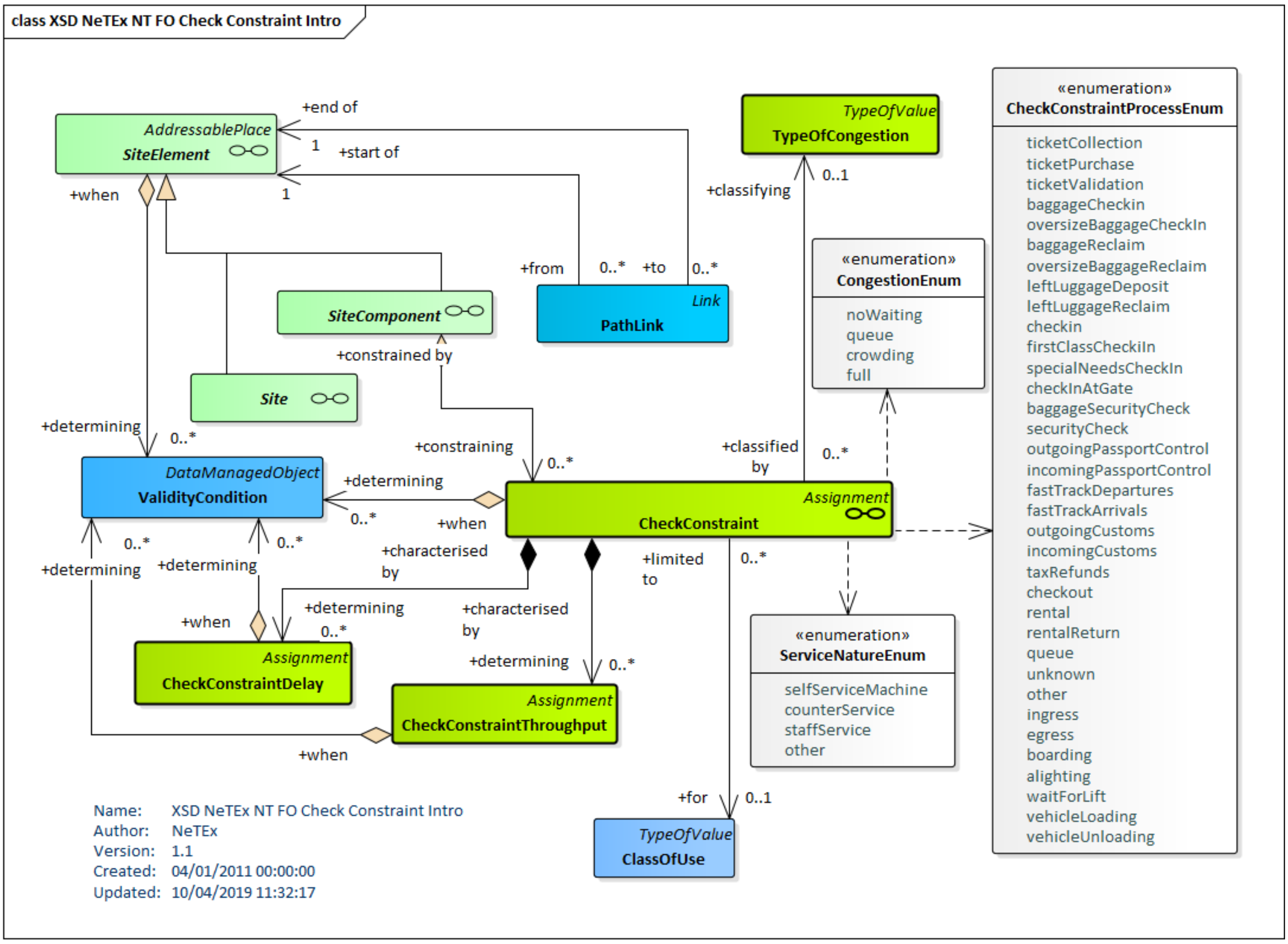


Figure 665 – Check Constraint – Physical Model (UML) – With Values

8.4.11.2.1 CheckConstraint – Physical Model details

The following figure shows detailed attributes of the CHECK CONSTRAINT model.

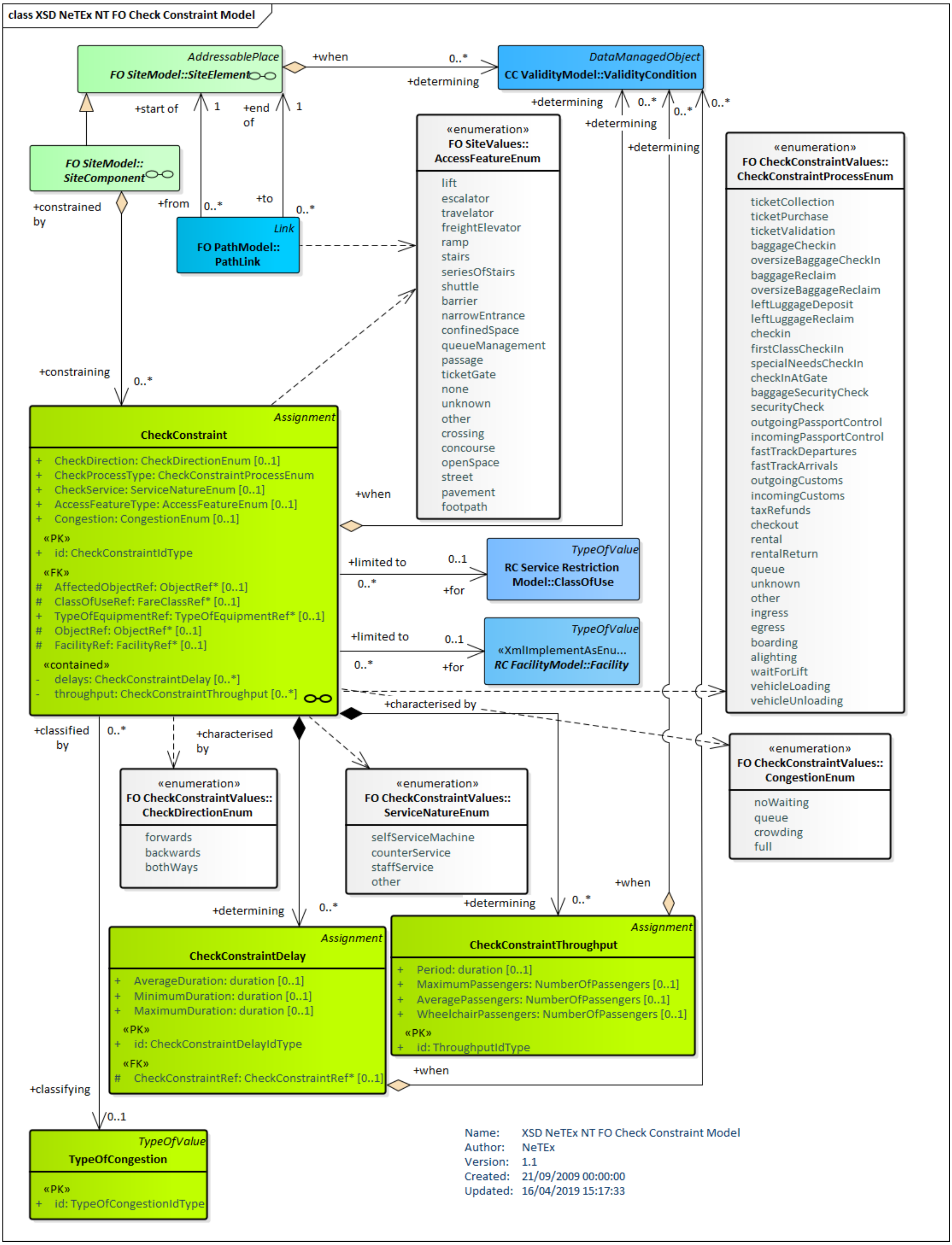


Figure 666 – Check Constraint – Physical Model (UML) – Detailed

8.4.11.3 CheckConstraint Path – Attributes and XSD

8.4.11.3.1 CheckConstraint – Model Element

Characteristics of a process that takes place at a SITE COMPONENT, such as check-in, security screening, ticket control or immigration, that may potentially incur a time penalty that should be allowed for when journey planning.

Table 567 – *CheckConstraint* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Assignment*	::>	CHECK CONSTRAINT inherits from ASSIGNMENT.
«PK»	***id***	*CheckConstraintIdType*	1:1	Identifier of CHECK CONSTRAINT.
«atr»	***order***	*xsd:integer*	0:1	Order of CHECK CONSTRAINT.
	Name	*MultilingualString*	0:1	Name of CHECK CONSTRAINT.
«FK»	***PlaceRef***	*(PlaceRef)*	0:1	Reference to PLACE affected by process defined by CHECK CONSTRAINT.
XGRP	***CheckConstraint-PropertiesGroup***	**xmlGroup**	0:1	General properties of a CHECK CONSTRAINT.
XGRP	***CheckConstraint-RelationsGroup***	**xmlGroup**	0:1	Related elemenst for a CHECK CONSTRAINT.

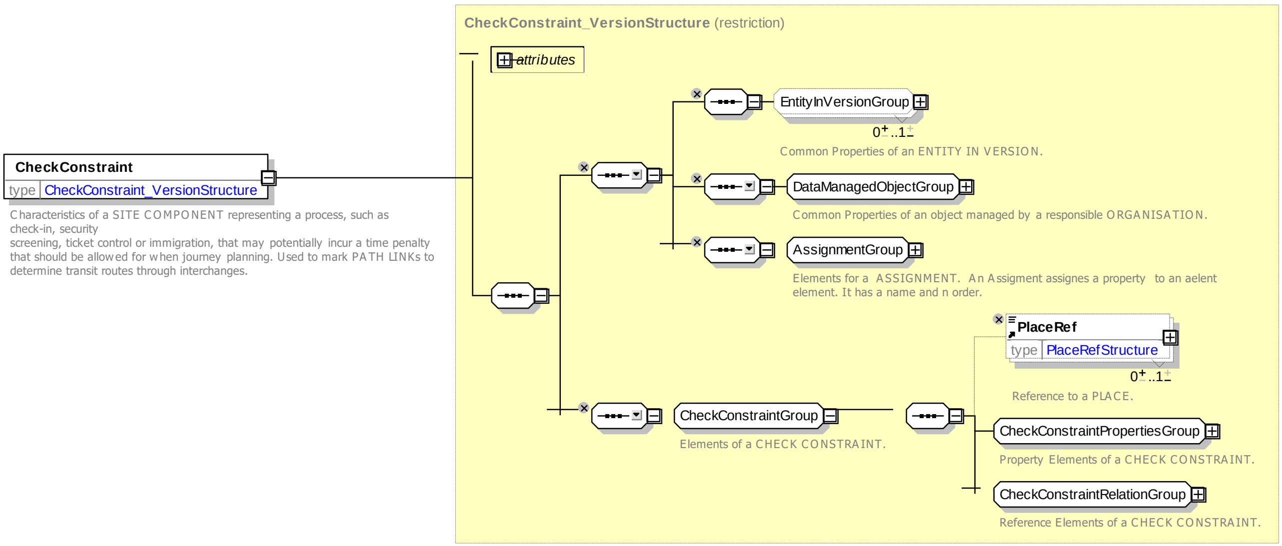


Figure 667 – *CheckConstraint* – *XSD*

8.4.11.3.1.1 CheckConstraintPropertiesGroup – XML Group

Table 568 – *CheckConstraintPropertiesGroup* – Group

Classification	Name	Type	Cardinality	Description
«enum»	***CheckDirection***	*CheckDirectionEnum*	0:1	Direction in which CHECK CONSTRAINT applies: forwards, back or both. For CHECK CONSTRAINT associated with links, it corresponds to direction of link. For CHECK CONSTRAINT associated with ENTRANCEs it corresponds to direction from outside SITE to inside SITE. See allowed values below.
«enum»	***CheckProcess***	*CheckProcessType-Enum*	0:1	Process that gives rise to CHECK CONSTRAINT. See allowed values below.
«enum»	***CheckService***	*ServiceNatureEnum*	0:1	Nature service associated CHECK CONSTRAINT. See allowed values below.
«enum»	***AccessFeatureType***	*AccessFeatureEnum*	0:1	Access feature associated with CHECK CONSTRAINT. See ACCESS SPACE for allwoed values.
«enum»	***Congestion***	*CongestionEnum*	0:1	Nature of congestion arising from CHECK CONSTRAINT. See allowed values below.
«FK»	***TypeOfCongestion-Ref***	*TypeOfCongestionRef*	0:1	Reference to TYPE OF CONGESTION characterising CHECK CONSTRAINT.

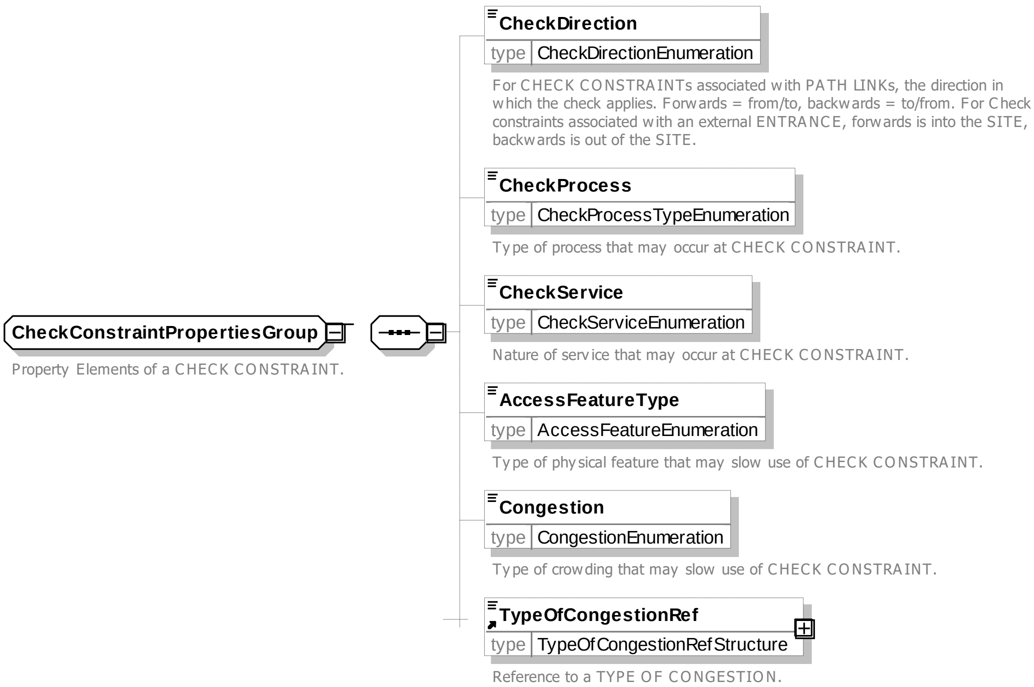


Figure 668 – *CheckConstraintPropertiesGroup – XSD*

CheckDirection – Allowed values

Allowed values for CHECK CONSTRAINT Direction (*CheckDirectionEnum*)

Table 569 – *CheckDirection* – Allowed Values

Value	Description
forwards	Constraint applies in forwards direction of associated link
backwards	Constraint applies in backwards direction of associated link
bothWays	Constraint applies in both directions of associated link

CheckProcessType – Allowed values

Allowed values for CHECK CONSTRAINT Chechkrocess (*CheckProcessEnum*).

Table 570 – *CheckProcessType* – Allowed Values

Value	Description
ticketCollection	Process is ticket collection
ticketPurchase	Process is ticket purchase
ticketValidation	Process is ticket validation
baggageCheckin	Process is baggage check in
oversizeBaggageCheckIn	Process is oversize baggage check in
baggageReclaim	Process is baggage reclaim
oversizeBaggageReclaim	Process is oversize baggage reclaim
leftLuggageDeposit	Process is left luggage deposit
leftLuggageReclaim	Process is left luggage reclaim
checkin	Process is check in
firstClassCheckinIn	Process is first class check in
specialNeedsCheckIn	Process is special needs check in
checkInAtGate	Process is check in at gate
baggageSecurityCheck	Process is baggage security check
securityCheck	Process is security check
outgoingPassportControl	Process is outgoing passport control
incomingPassportControl	Process is incoming passport control
fastTrackDepartures	Process is fast track departures
fastTrackArrivals	Process is fast track arrivals
outgoingCustoms	Process is outgoing customs
incomingCustoms	Process is incoming customs
taxRefunds	Process is tax refunds
checkout	Process is checkout
rental	Process is rental
rentalReturn	Process is rental return
queue	Process is a queue
unknown	Process is unknown
other	Other process
ingress	Process is ingress
egress	Process is egress
boarding	Process is boarding
alighting	Process is alighting
waitForLift	Process is wait for lift
vehicleLoading	Process is vehicle loading
vehicleUnloading	Process is vehicle unloading

ServiceNature – Allowed values

Allowed values for ***ServiceNature*** (*ServcieNatureEnumeration*).

Table 571 – *ServiceNature* – Allowed Values

Value	Description
selfServiceMachine	Process supported by Self-service machine
counterService	Process supported by counter service
staffService	Process supported by staff service
other	Process supported by other means

Congestion – Allowed values

Allowed values for ***Congestion*** (*CongestionEnumeration*).

Table 572 – *Congestion* – Allowed Values

Value	Description
noWaiting	No congestion
queue	Congestion causing queue
crowding	Crowding
full	Congestion at capacity

8.4.11.3.1.2 CheckConstraint – XML Group

Elements describing elements associated with a CHECK CONSTRAINT.

Table 573 – *CheckConstraintRelationsGroup* – Group

Classification	Name	Type	Cardinality	Description
«FK»	***ClassOfUseRef***	*FareClassRef*	0:1	Reference to CLASS OF USE for which CHECK CONSTRAINT applies. E.g. There may be different delays for First class than for second class.
«FK»	***TypeOfEquipment-Ref***	*TypeOfEquipmentRef*	0:1	Reference to a TYPE OF EQUIPMENT.
«FK»	***FacilityRef***	(*FacilityRef*)	0:1	Reference to FACILITY affected by CHECK CONSTRAINT.
«FK»	***EquipmentRef***	(*EquipmentRef*)	0:1	Reference to other object affected by CHECK CONSTRAINT, e.g. SERVICE JOURNEY.
«cntd»	***delays***	*CheckConstraintDelay*	0:*	CHECK CONSTRAINT DELAYs at CHECK CONSTRAINT.
«cntd»	***throughput***	*CheckConstraint-Throughput*	0:*	CHECK CONSTRAINT THROUGHPUTs at CHECK CONSTRAINT.

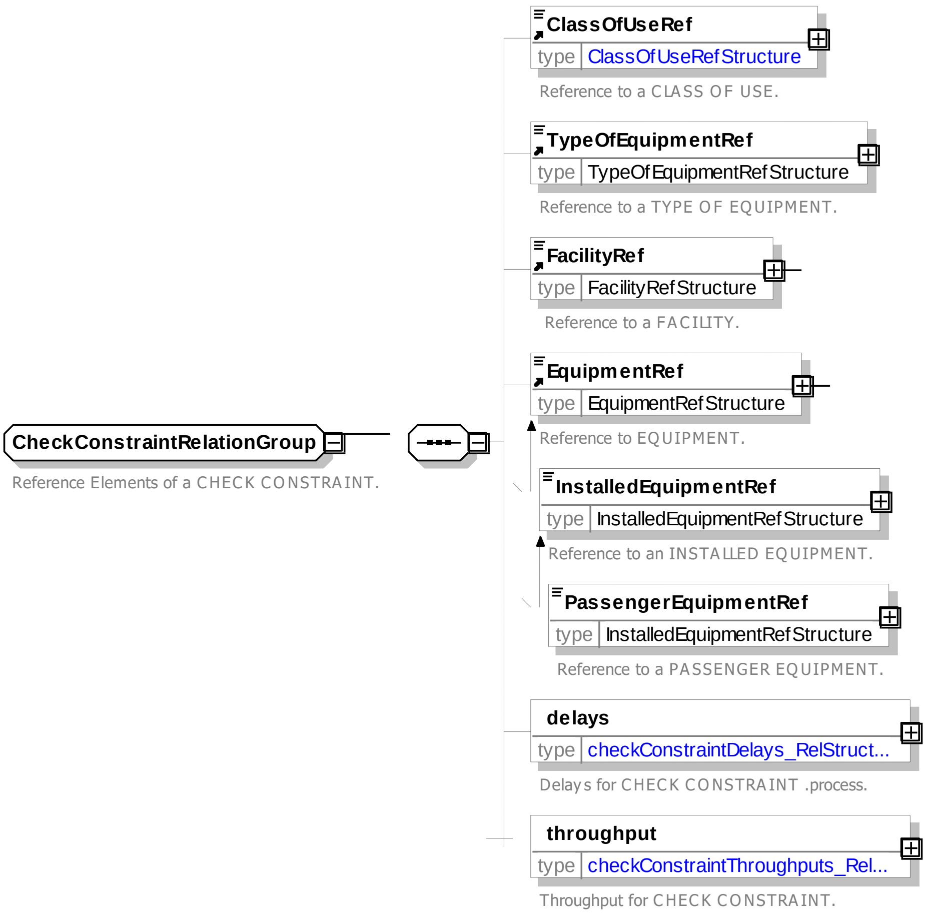


Figure 669 – *CheckConstraint – XSD*

8.4.11.3.2 CheckConstraintDelay – Model Element

Time penalty associated with a CHECK CONSTRAINT.

Table 574 – *CheckConstraintDelay* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Assignment*	::>	CHECK CONSTRAINT DELAY inherits from ASSIGNMENT.
«PK»	***id***	*CheckConstraintDelay-IdType*	1:1	Identifier of CHECK CONSTRAINT DELAY.
«atr»	***order***	*xsd:integer*	0:1	Order of CHECK CONSTRAINT DELAY.
«FK»	***CheckConstraint-Ref***	*CheckConstraintRef*	0:1	Parent CHECK CONSTRAINT of CHECK CONSTRAINT DELAY.
«FK»	***ClassOfUseRef***	*ClassOfUseRef*	0:1	Reference to CLASS OF USE affected by delay.
	MinimumLikely-Delay	*xsd:duration*	0:1	Minimum expected duration of delay of CHECK CONSTRAINT DELAY.

	AverageDelay	*xsd:duration*	0:1	Average duration of delay of CHECK CONSTRAINT DELAY.
	MaximumLikely-Delay	*xsd:duration*	0:1	Maximum expected duration of delay of CHECK CONSTRAINT DELAY.

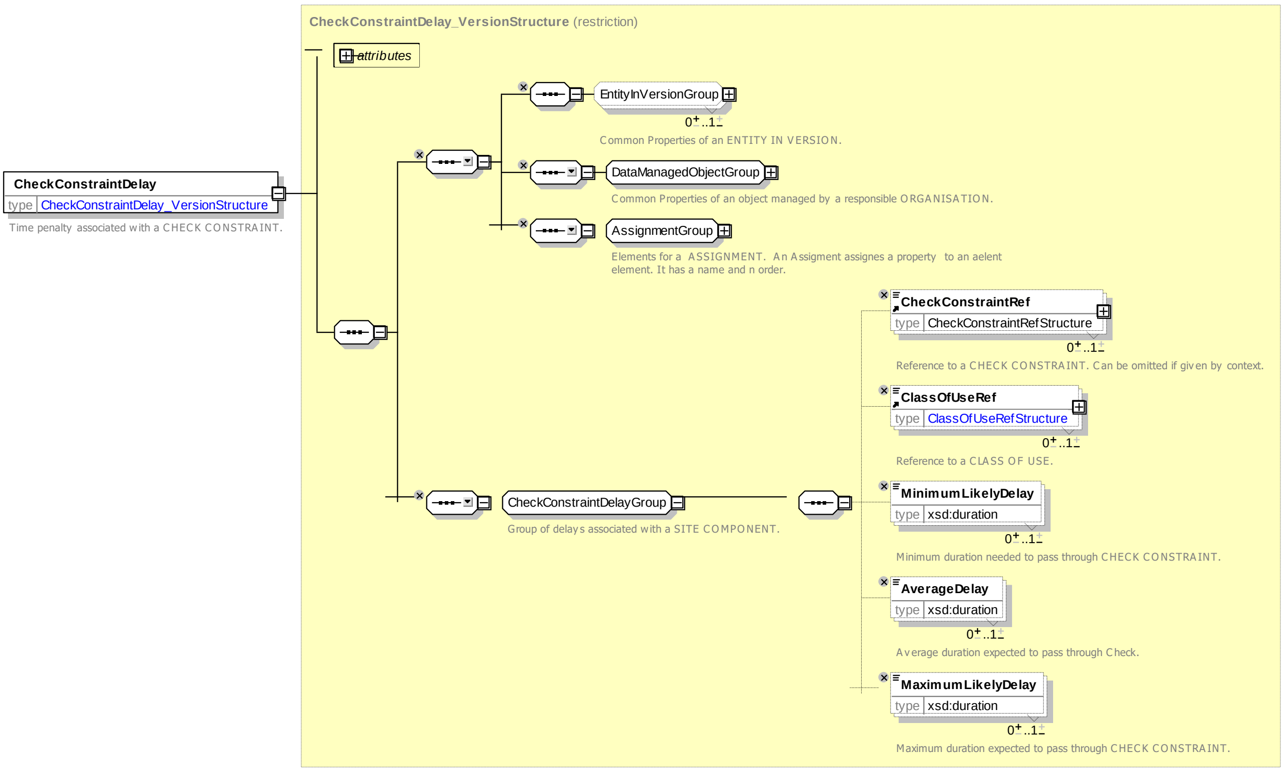


Figure 670 – *CheckConstraintDelay – XSD*

8.4.11.3.3 CheckConstraintThroughput – Model Element

Throughput of a CHECK CONSTRAINT: the number of passengers who can pass through it in a specified interval.

Table 575 – *CheckConstraintThroughput* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Assignment*	*::>*	CHECK CONSTRAINT THROUGHPUT inherits from ASSIGNMENT.
«PK»	***id***	*CheckConstraint-ThroughputIdType*	0:1	Identifier of CHECK CONSTRAINT THROUGHPUT.
«FK»	***CheckConstraintRef***	*CheckConstraintRef*	0:1	Parent CHECK CONSTRAINT of CHECK CONSTRAINT THROUGHPUT.
	Period	*xsd:duration*	0:1	Identifier of PERIOD for CHECK CONSTRAINT THROUGHPUT.

	MaximumPassengers	*NumberOfPassengers*	0:1	Maximum number of passengers for CHECK CONSTRAINT THROUGHPUT.
	AveragePassengers	*NumberOfPassengers*	0:1	Average number of passengers for CHECK CONSTRAINT THROUGHPUT.
	Wheelchair-Passengers	*NumberOfPassengers*	0:1	Maximum number of wheelchair passengers for CHECK CONSTRAINT THROUGHPUT.

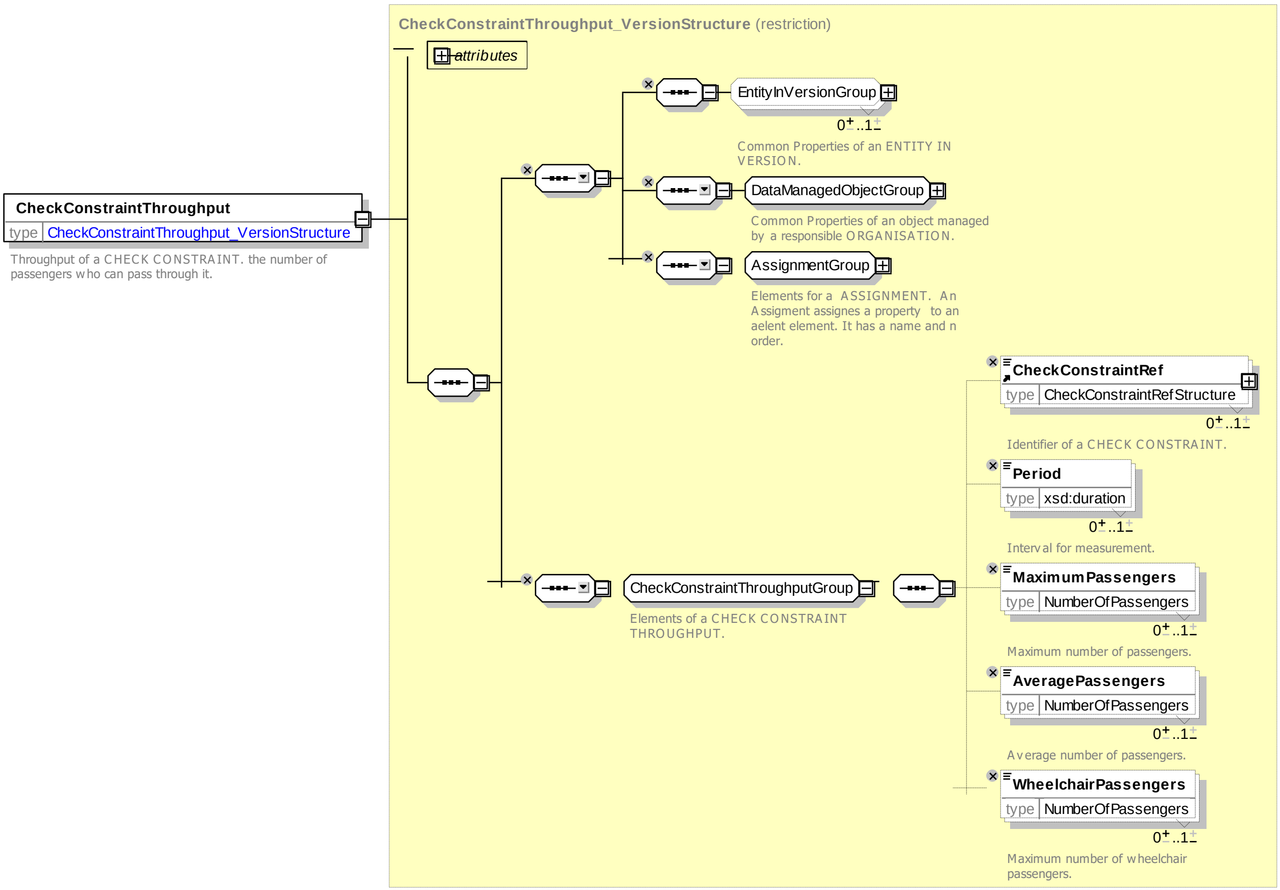


Figure 671 – *CheckConstraintThroughput – XSD*

8.4.12 Parking

8.4.12.1 PARKING – Conceptual MODEL

NeTEx includes a model to describe parking elements. A PARKING is a type of SITE that describes the availability of parking for different types of vehicles, and its relation to other SITEs such as stations. Concerning such road related issues, it is reminded that the NeTEx model is intended to be interoperable with the Datex2 representation.

A PARKING may be described in summary – for example, a car park of 50 places, or be further broken down into PARKING AREAs (each on a LEVEL), each containing individual PARKING BAYs of a designated size.

— A PARKING may have designated PARKING VEHICLE ENTRANCEs as well as PARKING PASSENGER ENTRANCEs.

— Charges may be specified using a PARKING TARIFF and PARKING TARIFF CHARGE BAND elements – See NeTEx Part3.

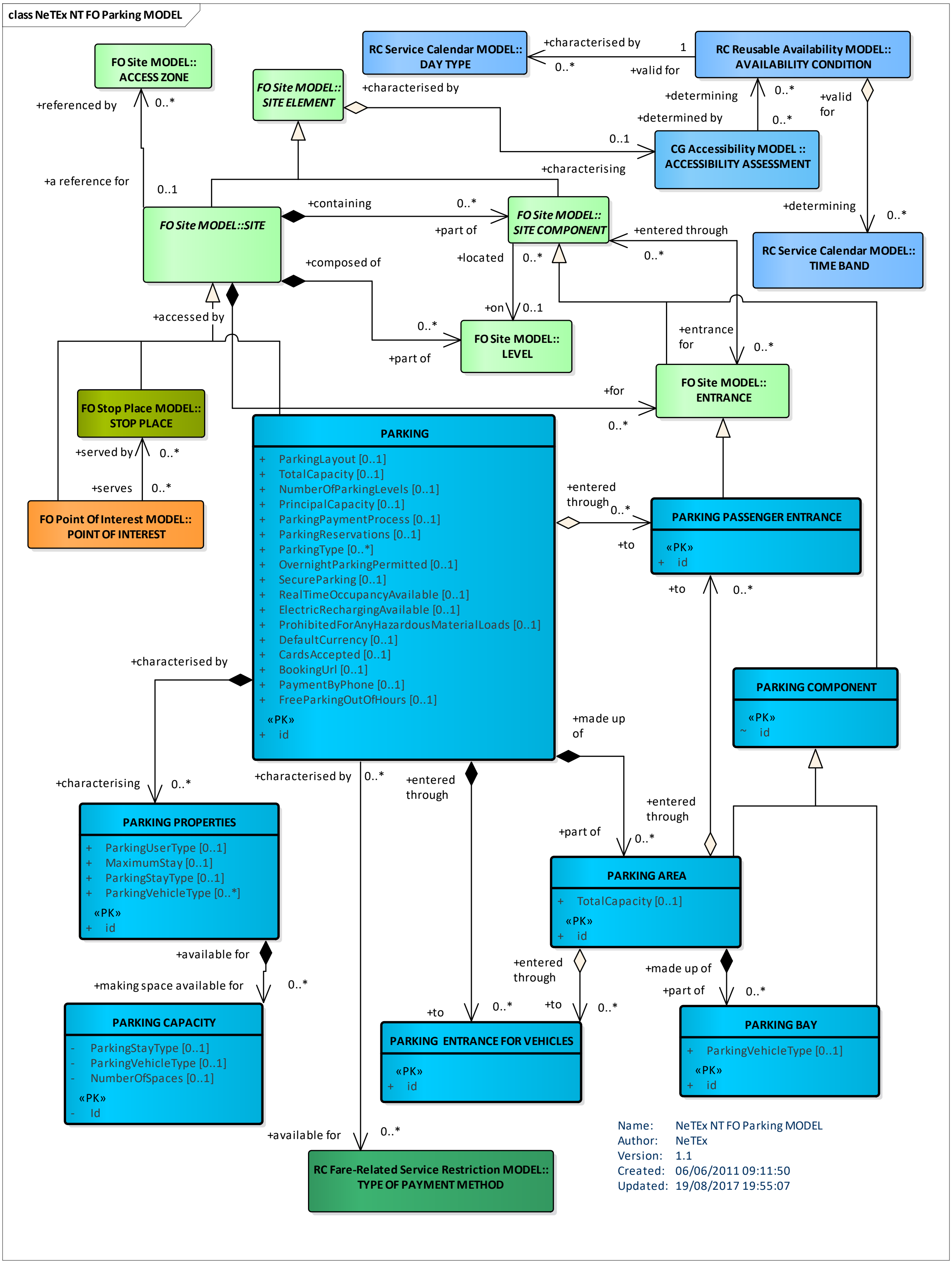


Figure 672 – PARKING – Conceptual MODEL (UML)

8.4.12.2 Parking – Physical Model

8.4.12.2.1 Detailed properties of a Parking – Physical Model

The following figure shows detailed attributes of the PARKING model.

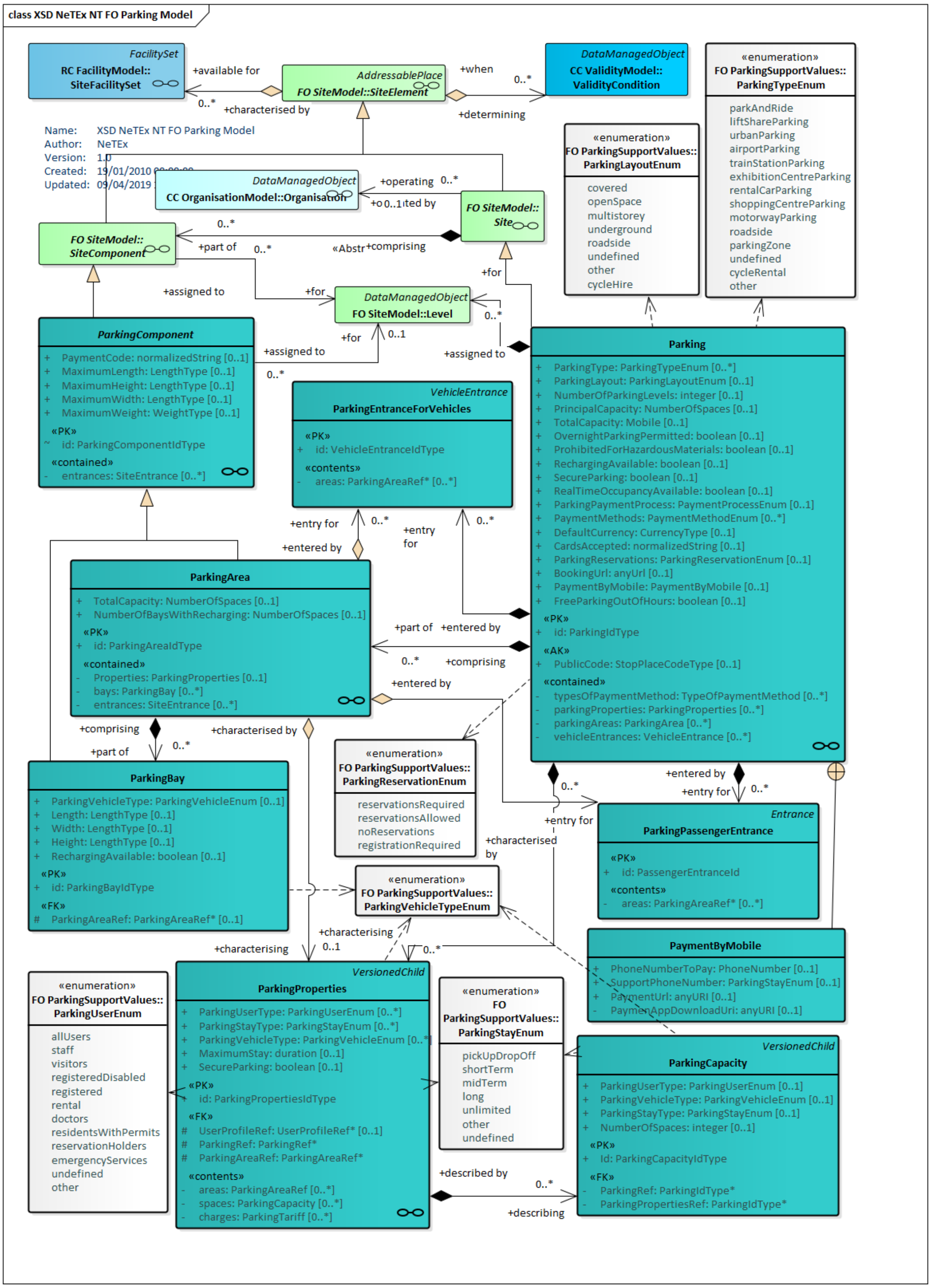


Figure 673 –Parking Model – Physical Model (UML) – Detail

8.4.12.2.2 Parking Payment Means – Physical Model

The process and method of payment isa useful part of passenger infromation for parking. The following figure shows attributes of the PARKING model related to the means of payment.

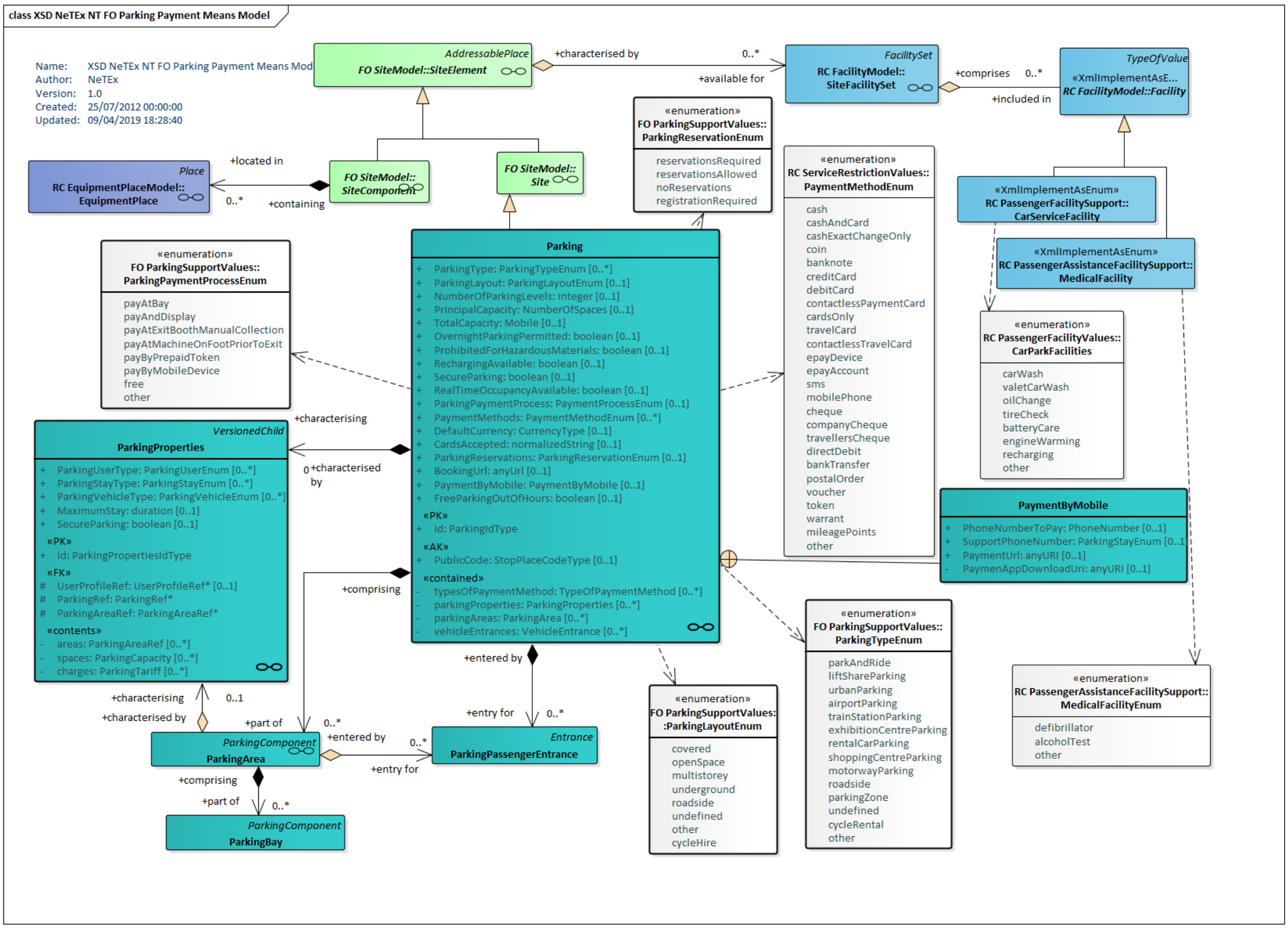


Figure 674 –Parking Payment Means – Physical Model (UML) – Detail

8.4.12.3 Parking – Attributes and XSD

8.4.12.3.1 Parking – Model Element

Designated locations for leaving vehicles such as cars, motorcycles and bicycles.

Table 576 – *Parking* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Site*</u>	::>	PARKING inherits from SITE.
«PK»	***id***	*ParkingIdType*	1:1	Identifier of PARKING.
XGRP	*SiteAccessGroup*	<u>***xmlGroup***</u>	0:1	Elemetns describing relationship of PARKINGto other SITEs. See SITE MODEL.
«AK»	***PublicCode***	*StopPlaceCodeType*	0:1	Public Code of PARKING.

XGRP	***ParkingDetailsGroup***	***xmlGroup***	0:1	Elements describing PARKING.
XGRP	***ParkingChargeGroup***	***xmlGroup***	0:1	Elements relating to PARKING charges.
«cntd»	***parkingProperties***	*ParkingProperties*	0:*	PARKING PROPERTies of PARKING.
«cntd»	***parkingAreas***	*ParkingArea*	0:*	PARKING AREAs within PARKING.
«cntd»	***vehicleEntrances***	*ParkingEntranceForVehicles*	0:*	Vehicle Entrances for PARKING. (for pedestrian entrance see supertype SITE.)

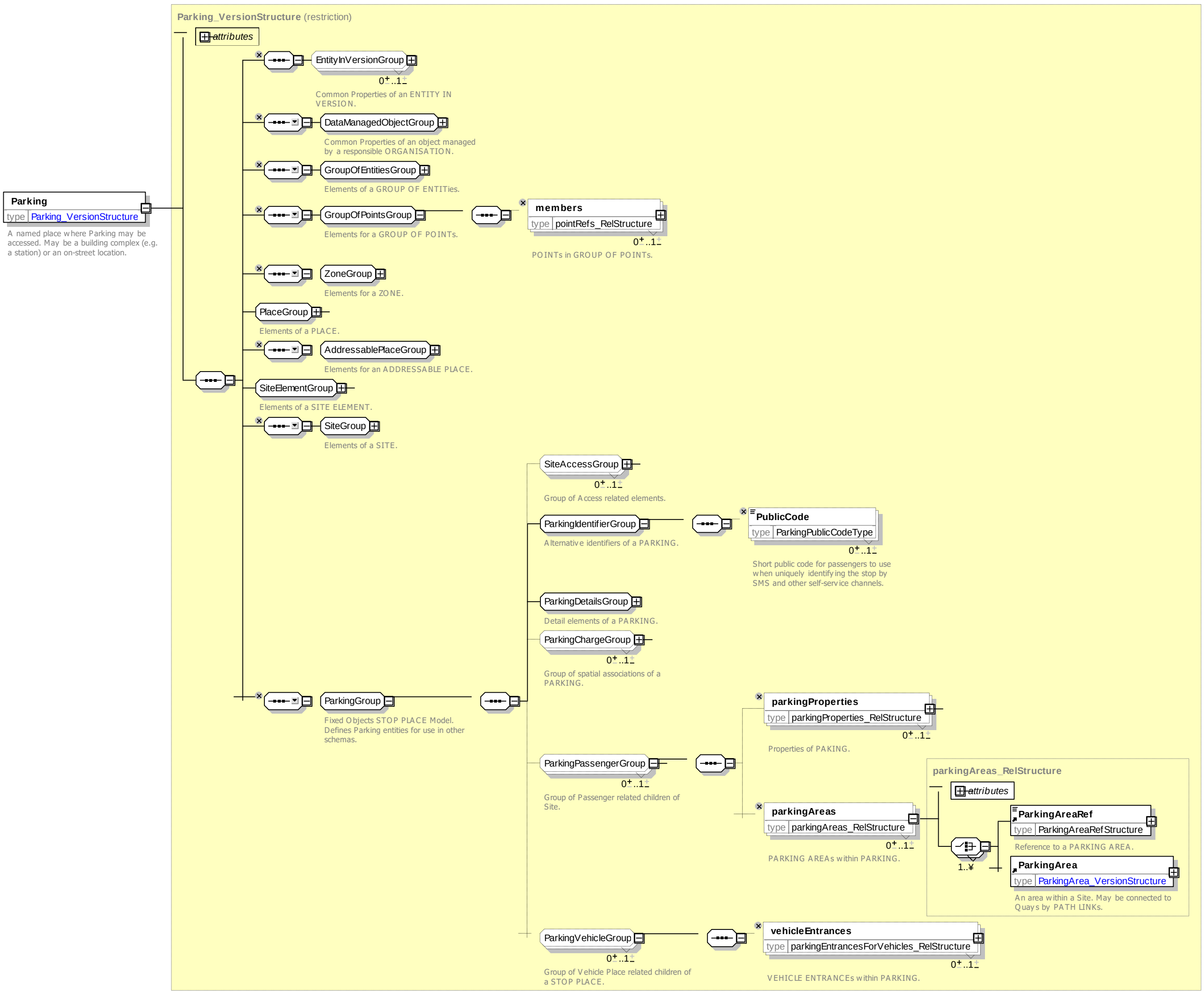


Figure 675 – *Parking – XSD*

8.4.12.3.1.1 ParkingDetailsGroup – XML Group

Table 577 – *ParkingDetailsGroup* – Group

Classification	Name	Type	Cardinality	Description

	Label	MultilingualString		Additional Label of PARKING.
«enum»	***ParkingType***	*ParkingTypeEnum*	0:*	Nature of PARKING. See allowed values below.
«enum»	***ParkingVehicleTypes***	*ParkingVehicleEnum*	0:*	Types of Vehicle allowed in PARKING. See allowed values below.
«enum»	***ParkingLayout***	*ParkingLayoutEnum*	0:1	Layout type of PARKING. See allowed values below.
	NumberOfParkingLevels	*xsd:nonNegative-Integer*	0:1	Total number of levels.
	PrincipalCapacity	*NumberOfSpaces*	0:1	Principal Capacity of PARKING.
	TotalCapacity	*NumberOfSpaces*	0:1	Total Capacity of PARKING.
	OvernightParkingPermitted	*xsd:boolean*	0:1	Whether overnight PARKING is allowed.
	ProhibitedForHazardous-Materials	*xsd:boolean*	0:1	Whether PARKING is prohibited for any Hazardous material.
	RechargingAvailable	*xsd:boolean*	0:1	Whether car park has recharging points.
	Secure	*xsd:boolean*	0:1	Whether Parking is offered as secure.
	RealTimeOccupancy-Available	*xsd:boolean*	0:1	Whether there is real-time occupancy data for PARKING.

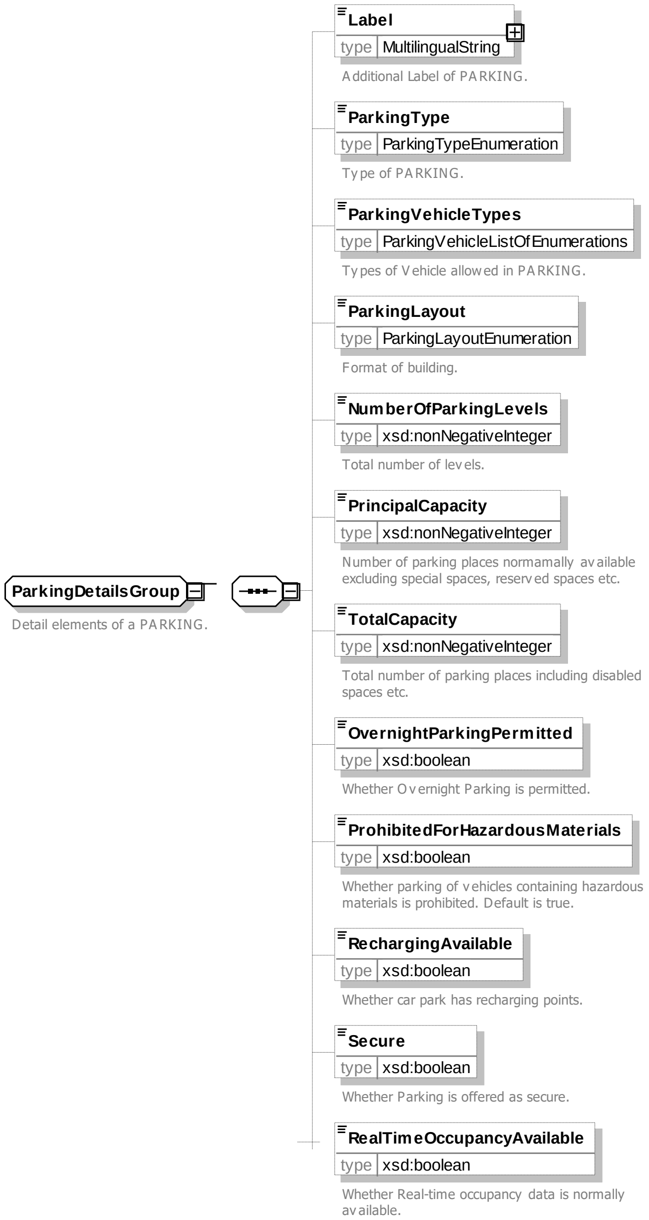


Figure 676 – *ParkingDetailsGroup – XSD*

ParkingType – Allowed values

Allowed values for ***ParkingType*** (*ParkingTypeEnumeration*).

Table 578 – *Parking* – Allowed Values

Value	Description
parkAndRide	Park and ride parking.
liftShareParking	Lift share parking.
urbanParking	Urban parking.
airportParking	Airport parking.
trainStationParking	Train station parking.
exhibitionCentreParking	Exhibition centre parking.
rentalCarParking	Rental car parking.
shoppingCentreParking	Shopping centre parking.
motorwayParking	Motorway parking.

roadside	Roadside parking.
parkingZone	Parking zone parking.
undefined	Undefined parking.
cycleRental	Cycle rental stand.
other	Other parking.

ParkingVehicleType – Allowed values

Allowed values for ***ParkingVehicleType*** (*ParkingVehicleTypeEnumeration*)

Table 579 – *ParkingVehicleType* – Allowed Values

Value	Description
pedalCycle	Pedal Cycle
moped	Moped
motorcycle	Motorcycle.
motorcycleWithSidecar	Motorcycle with sidecar.
motorScooter	Motor scooter.
twoWheeledVehicle	Two wheeled vehicle.
threeWheeledVehicle	Three wheeled vehicle.
car	Car.
smallCar	Small Car.
passengerCar	Passenger Car.
largeCar	Large Car.
taxi	Taxi.
camperCar	Camper Car.
carWithCaravan	Car with Caravan.
carWithTrailer	Car with Trailer.
caravan	Caravan.
minibus	Minibus.
bus	Bus
van	Van
fourWheelDrive	Four wheel drive vehicles
highSidedVehicle	High sided vehicle.
lightGoodsVehicle	Light goods vehicle.
lightGoodsVehicleWithTrailer	Light goods vehicle with trailer.
heavyGoodsVehicleWithTrailer	Heavy goods vehicle.
heavyGoodsVehicle	Heavy goods vehicle with trailer.
truck	Truck.
agriculturalVehicle	Agricultural vehicle.
tanker	Tanker.
tram	Tram.
articulatedVehicle	Articulated vehicle.
vehicleWithTrailer	Vehicle with trailer.
other	Other vehicle type.
undefined	Undefined vehicle type.
all	All vehicle types.

ParkingLayout – Allowed values

Allowed values for ***ParkingLayout*** (*ParkingLayoutEnumeration*).

Table 580 – *ParkingLayout* – Allowed Values

Value	Description
covered	Covered car park.
openSpace	Open space car park.
multistorey	Multi-storey car park.
underground	Underground car park.
roadside	Roadside car park.
undefined	Undefined car park.
other	Other car park.
cycleHire	A cycle hire location with locked racks.

8.4.12.3.1.2 **ParkingChargeGroup – XML Group**

PARKING elements relating to charges and payment

Table 581 – *ParkingChargeGroup* – Group

Classifi cation	Name	Type	Cardinal ity	Description
«enum»	***ParkingPaymentProcess***	*PaymentProcessEnum*	0:1	How to pay for PARKING.
«enum»	***PaymentMethods***	*PaymentMethodEnum*	0:*	Method of Payment for use of PARKING. See RC Service Restrictions Model for allowed values.
«cntd»	***typesOfPaymentMethod***	*TypeOfPaymentMethod*	0:*	Open TYPEs OF PAYMENT METHOD accepted for parking.
	DefaultCurrency	*CurrencyType*	0:1	Default Currency for payment.
«enum»	***CurrenciesAccepted***	*CurrencyList*	0:1	Currencies accepted.
	CardsAccepted	xsd:NMTOKENS	0:1	Payment Cards that are accepted
«enum»	***ParkingReservations***	*ParkingReservationEnum*	0:1	How to reserve for PARKING. See allowed values below.
	BookingUrl	*xsd:anUri*		URL to make booking.
ntd»	***PaymentByMobile***	*PaymentByMobile*	0:1	How to make payment by phone.
« »	***FreeParkingOutOfHours***	*xsd:boolean*	0:1	hether there is free parking out of hours.
«cntd»	***parkingProperties***	*ParkingProperties*	0:*	PARKING PROPERTies of PARKING.
«cntd»	***parkingAreas***	*ParkingArea*	0:*	PARKING AREAs within PARKING.
«cntd»	***entrances***	*StopPlaceEntrance*	0:*	Pedestrian Entrances for PARKING.

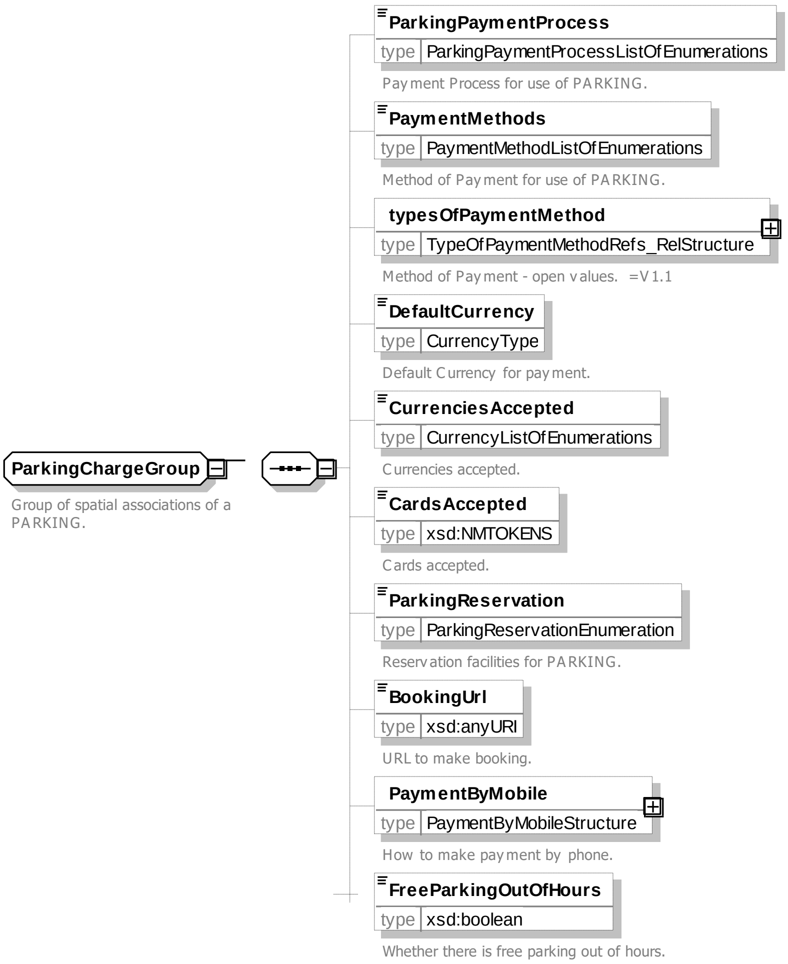


Figure 677 – *ParkingChargeGroup – XSD*

ParkingPaymentProcess – Allowed values

Allowed values for Parking Payment Process (*ParkingPaymentProcessEnumeration*).

Table 582 – *ParkingPaymentProcess* – Allowed Values

Value	Description
payAtBay	Pay at bay.
payAndDisplay	Buy ticket and display on vehicle.
payAtExitBoothManual-Collection	Pay at exit booth.
payAtMachineOnFoot-PriorToExit	Pay machine on foot prior to exit
payByPrepaidToken	Pay by prepaid token.
payByMobileDevice	Pay by mobile device.
other	Other means of payment for parking.

ParkingReservationType – Allowed values

Allowed values for Parking Reservation (*ParkingReservationEnumeration*).

Table 583 – *ParkingReservationType* – Allowed Values

Value	Description
reservationsRequired	Reservations required.
reservationsAllowed	Reservations allowed.
noReservations	No reservations.
registrationRequired	Whether preregistration is needed to use scheme.
other	Other condition.

8.4.12.3.2 PaymentByMobile – Model Element

Generic COMPONENT of a PARKING (e.g. PARKING AREA or PARKING BAY) +v1.1

Table 584 – *PaymentByMobile* – Element

Classification	Name	Type	Cardinality	Description
	PhoneNumberToPay	*PhoneType*	*0:1*	Phone number to call or text to pay.
	SupportPhoneNumber	*PhoneType*	*0:1*	Phone number for support.
	PaymentUrl	*xsd:anyUri*	*0:1*	URL to make payment.
	PaymentApp-DownloadUrl	*xsd:anyUri*	*0:1*	URL to downlaod app to pay.

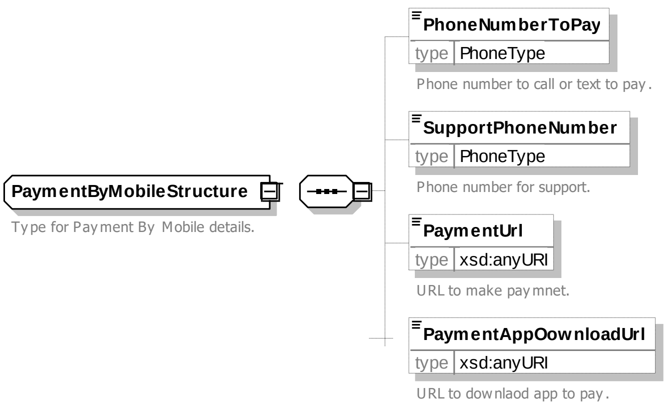


Figure 678 – *PaymentByMobile Structure – XSD*

8.4.12.3.3 ParkingComponent – Model Element

Generic COMPONENT of a PARKING (e.g. PARKING AREA or PARKING BAY)

Table 585 – *ParkingComponent* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SiteComponent*	*::>*	PARKING COMPONENT inherits from SITE COMPONENT.
«PK»	***id***	*ParkingComponentIdType*	1:1	Identifier of PARKING COMPONENT.

	ParkingPayment-Code	*xsd:normalizedString*	0:1	PaymentCode assoicated with PARKING COMPONENT
	Label	*MultilingualString*	0:1	Additional Label of PARKING COMPONENT.
	MaximumLength	*LengthType*	0:1	Maximum length of VEHICLE to use PARKING AREA.
	MaximumWidth	*LengthType*	0:1	Maximum width of VEHICLE to use PARKING AREA.
	MaximumHeight	*LengthType*	0:1	Maximum height of VEHICLE to use PARKING AREA.
	MaximumWeight	*WeightType*	0:1	Maximum weight of VEHICLE to use PARKING AREA.

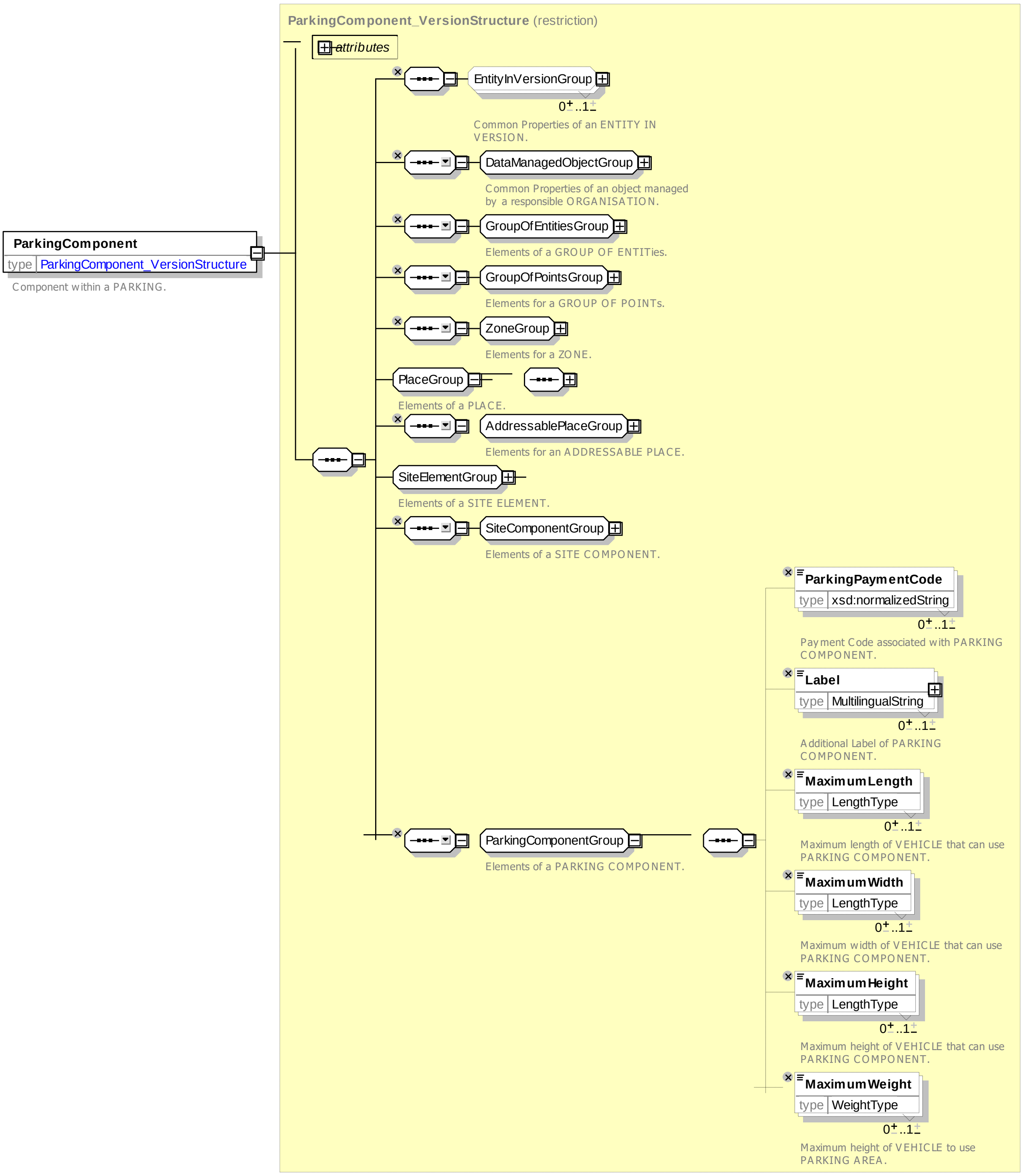


Figure 679 – *ParkingComponent – XSD*

8.4.12.3.4 ParkingArea – Model Element

A marked zone within a PARKING containing PARKING BAYs.

Table 586 – *ParkingArea* – Element

Classification	Name	Type	Cardinality	Description

::>	*::>*	*SiteComponentGroup*	*::>*	PARKING AREA inherits from SITE COMPONENT.
«PK»	***id***	*ParkingAreaIdType*	1:1	Identifier of PARKING AREA.
	TotalCapacity	*NumberOfVehicles*	0:1	Total Capacity of PARKING AREA.
	NumberOf-BaysWithWIth Recharging	*NumberOfSpaces*	0:1	Total number of bays with electric charging points in PARKING AREA.
«cntd»	***Parking-Properties***	*ParkingProperties*	0:1	Properties of PARKING AREA.
«cntd»	***bays***	*ParkingBay*	0:*	Bays within PARKING AREA.
«cntd»	***entrances***	(*SiteEntrance*) ParkingPAssenger-Entrance | ParkingEntranceFor-Passengers	0:*	ENTRANCEs of PARKING COMPONENT.

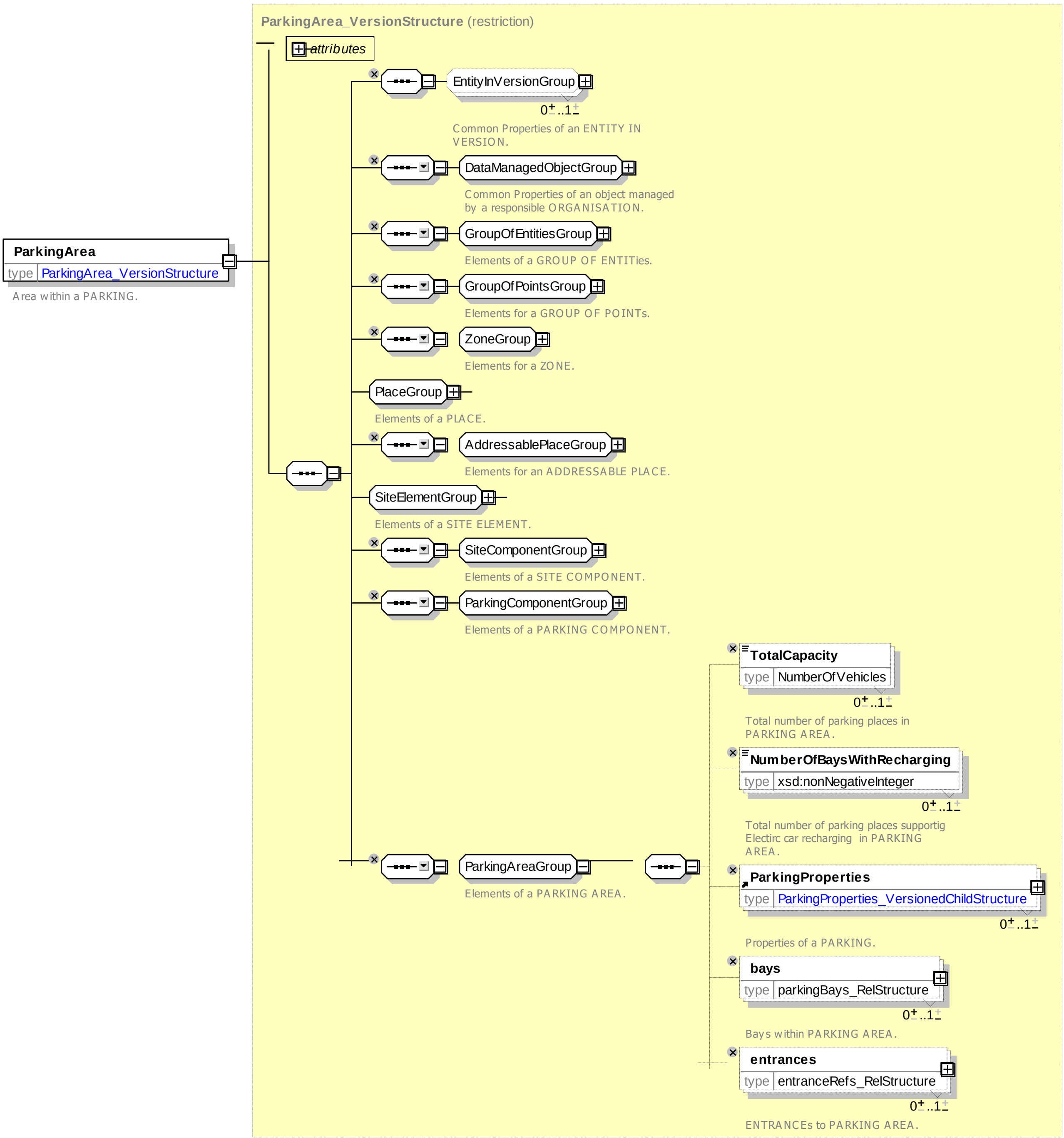


Figure 680 – *ParkingArea – XSD*

8.4.12.3.5 *ParkingBay* – Model Element

A place to park an individual vehicle.

Table 587 – *ParkingBay* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*SiteComponent*	*::>*	PARKING BAY inherits from SITE COMPONENT.

«PK»	***id***	*ParkingBayIdType*	1:1	Identifier of PARKING BAY.
«FK»	***ParkingAreaRef***	*ParkingAreaRef*	0:1	PARKING AREA within which PARKING BAY is found.
«enum»	***ParkingVehicle-Type***	*ParkingVehicleEnum*	0:1	TYPEs of VEHICLE that may use PARKING BAY.
	Length	*LengthType*	0:1	Usable length of PARKING BAY.
	Width	*LengthType*	0:1	Usable width PARKING BAY.
	Height	*LengthType*	0:1	Usable height of PARKING BAY.
	Weight	*WeightType*	0:1	Maximum weight of VEHICLE to use PARKING BAY.
	Recharging-Available	*xsd:boolean*	*0:1*	Whether power for recharging. See Equipment for details.

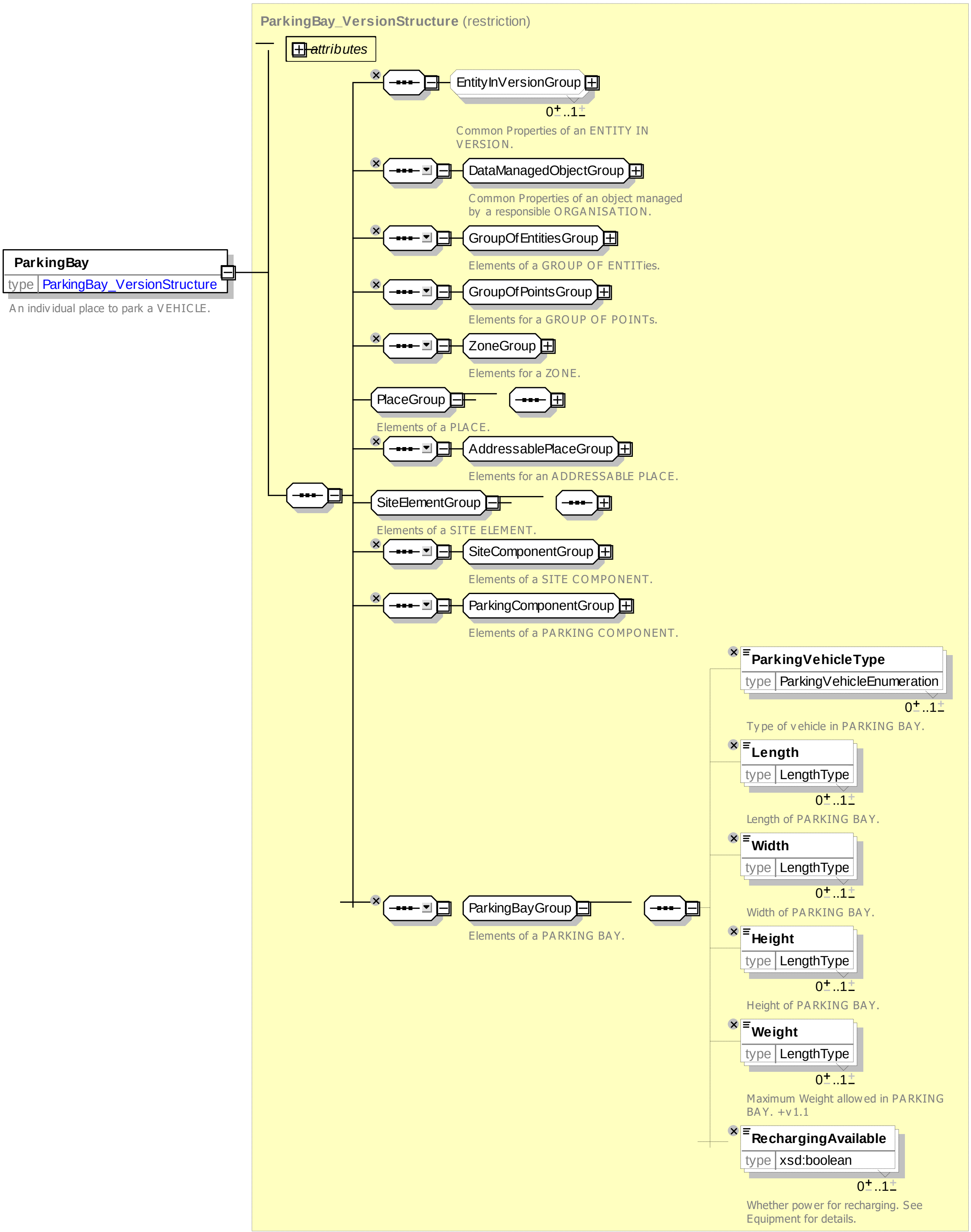


Figure 681 – *ParkingBay – XSD*

8.4.12.3.6 ParkingEntranceForVehicles – Model Element

An entrance for vehicles to the PARKING from the road.

Table 588 – *ParkingEntranceForVehicle* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*SiteEntrance*	::>	PARKING VEHICLE ENTRANCE inherits from SITE ENTRANCE.
«PK»	***id***	*ParkingEntranceFor-VehiclesIdType*	1:1	Identifier of PARKING VEHICLE ENTRANCE.
«cntd»	**areas**	*ParkingArea*Ref	0:*	PARKING AREAs for which this is an ENTRANCE.

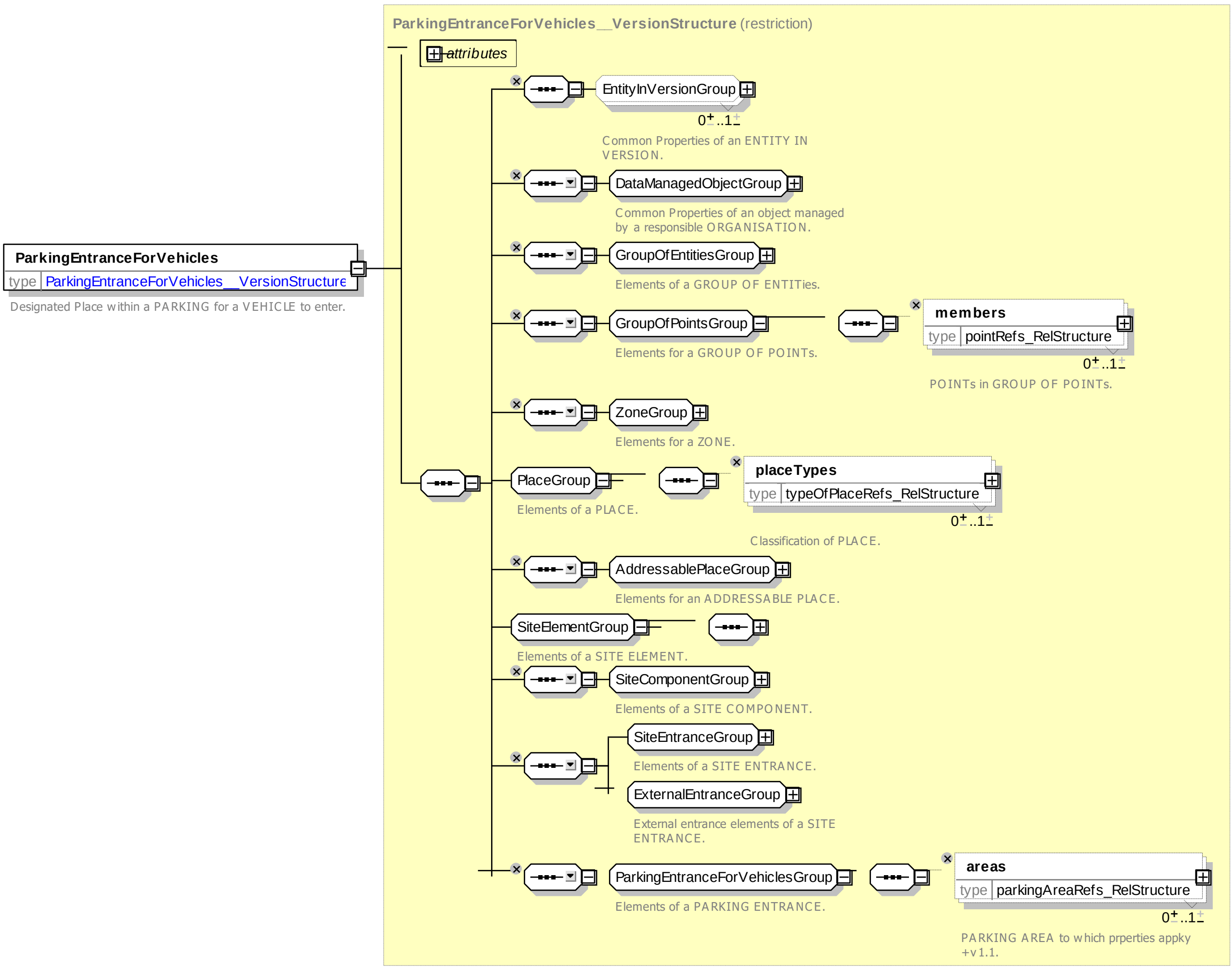


Figure 682 – *ParkingEntranceForVehicles* – *XSD*

8.4.12.3.7 *ParkingPassengerEntrance* – Model Element

An entrance to the PARKING for passengers on foot or other out-of-vehicle mode, such as wheelchair.

Table 589 – *ParkingPassengerEntrance* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*Entrance*	::>	PARKING PASSENGER ENTRANCE inherits from SITE ENTRANCE.
«PK»	***id***	*PassengerEntrance-IdType*	1:1	Identifier of PARKING PASSENGER ENTRANCE.
«cntd»	**areas**	*ParkingArea*Ref	0:*	PARKING AREAs for which this is an ENTRANCE. +v1.1.

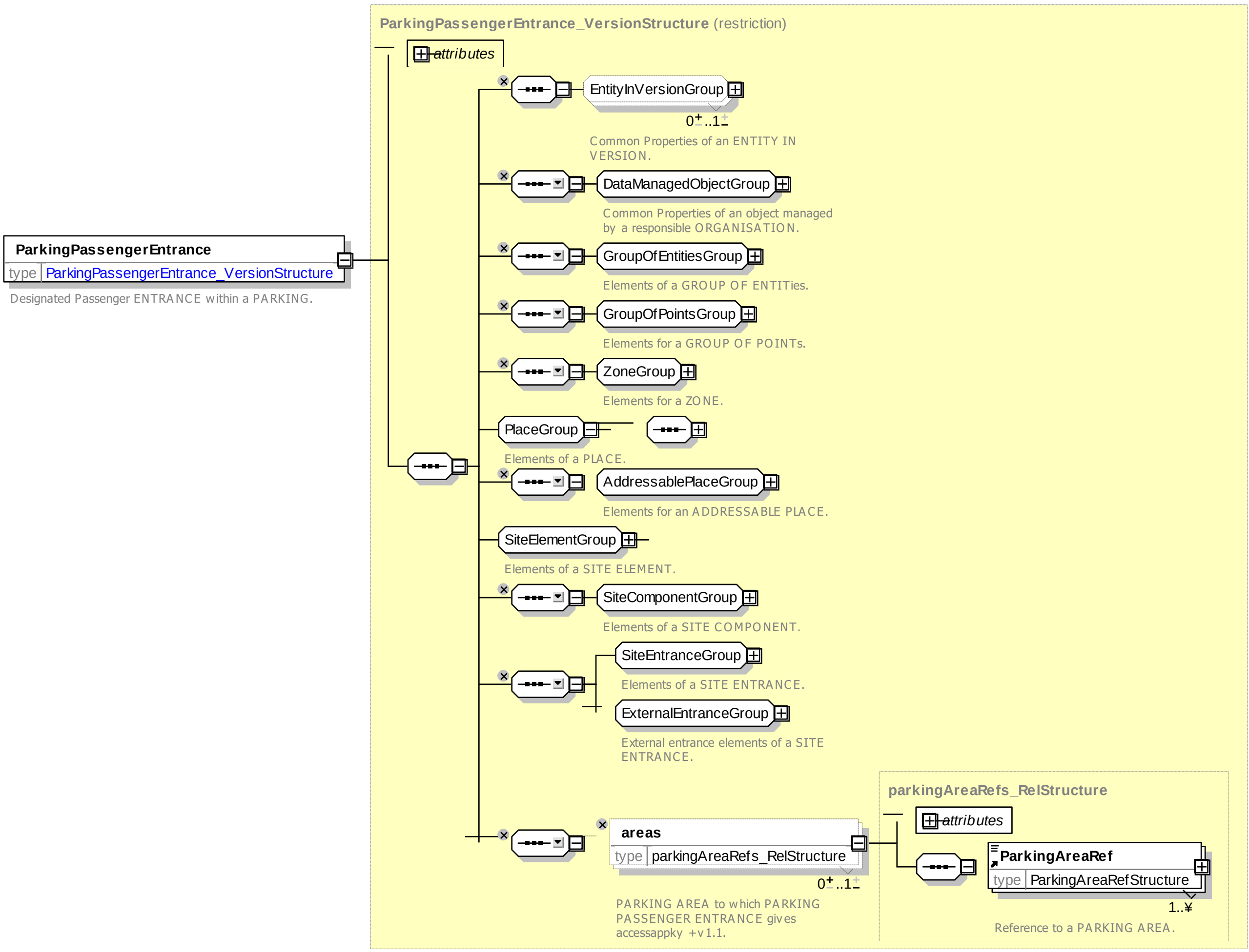


Figure 683 – *ParkingPassengerEntrance – XSD*

8.4.12.3.8 ParkingProperties – Model Element

PARKING specific properties other than PARKING CAPACITY.

Table 590 – *ParkingProperties* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	PARKING PROPERTies inherits from VERSIONED CHILD.

«PK»	***id***	*ParkingPropertiesIdType*	1:1	Identifier of PARKING PROPERTies.
«FK»	***SiteElementRef***	*(SiteElementRef)*	0:1	Parent PARKING or PARKING component associated with PARKING PROPERTies.
«enum»	***ParkingUserType***s	*ParkingUserEnum*	0:*	Types of users of PARKING PROPERTies. See allowed values below.
«enum»	***ParkingVehicle-Type***s	*ParkingVehicleTypeEnum*	0:*	Types of Vehicle for PARKING PROPERTies. See allowed values earlier above.
«enum»	***ParkingStayListe***	*ParkingStayEnum*	0:*	Types of stay specified by this PARKING PROPERTies. See allowed values below.
	MaximumStay	*xsd:duration*	0:1	Maximum stay specified by this PARKING PROPERTies.
«cntd»	**areas**	*ParkingArea*	0:*	PARKING AREAs associated with this PARKING PROPERTies. +v1.1
«cntd»	***spaces***	*ParkingCapacity*	0:*	Number of spaces specified by this PARKING PROPERTies.
«cntd»	***charges***	*ParkingTariff*	0:*	PARKING TARIFF specified by this PARKING PROPERTies.

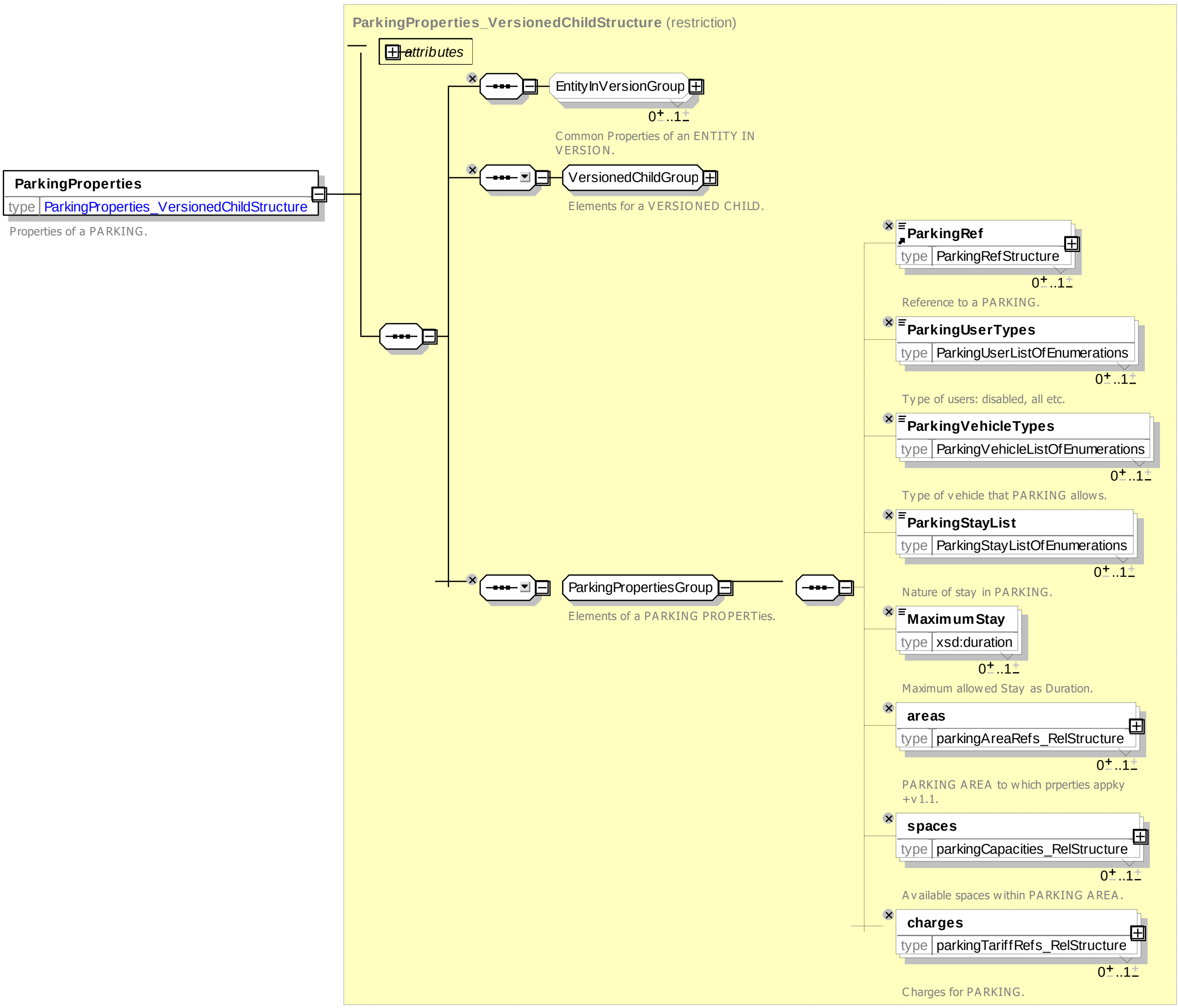


Figure 684 – *ParkingProperties – XSD*

8.4.12.3.8.1 ParkingUserType – Allowed values

Allowed values for ***ParkingUser*** (*ParkingUserEnumeration*).

Table 591 – *ParkingUserType* – Allowed Values

Value	Description
allUsers	Parking for all users.
staff	Parking for staff.
visitors	Parking for visitors.
registeredDisabled	Parking for registered disabled only.
registered	Parking for registered users only.
rental	Parking is for rented vehicles.
doctors	Parking for doctors only.
residentsWithPermits	Parking for residents with permits.
reservationHolders	Parking for reservation holders only.
emergencyServices	Parking for emergency services.
all	All user type.

undefined	Parking use undefined	*other*	Other type of parking

8.4.12.3.8.2 ParkingStay – Allowed values

Allowed values for ***ParkingStay*** (*ParkingStayEnumeration*).

Table 592 – *ParkingTerm* – Allowed Values

Value	Description
dropOff	Pick up and drop off parking.
shortSay	Short term parking.
midTerm	Mid-term parking.
*long*Term	Long term parking.
unlimited	Unlimited term parking.
all	All types available.
other	Other parking.
undefined	Undefined parking.

8.4.12.3.9 *ParkingCapacity* – Model Element

*P*ARKING properties providing information about its CAPACITY.

Table 593 – *ParkingCapacity* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	<u>*VersionedChild*</u>	*::>*	PARKING CAPACITY inherits from VERSIONED CHILD.
«PK»	***id***	*ParkingCapacityIdType*	1:1	Identifier of PARKING CAPACITY.
«FK»	***SiteElementRef***	*(SiteElementRef)*	1:1	PARKING element to which this PARKING CAPACITY belongs.
«enum»	***ParkingUserType***	*ParkingUserEnum*	0:1	TYPE OF VEHICLE specified by this PARKING CAPACITY.
«enum»	***ParkingVehicle-Type***	*ParkingVehicleEnum*	0:1	TYPE OF VEHICLE specified by this PARKING CAPACITY.
«enum»	***ParkingStayType***	*ParkingStayEnum*	0:1	Type of stay of PARKING CAPACITY.
	NumberOfSpaces	*NumberOfVehicles*	0:1	Number of parking spaces specified by this PARKING CAPACITY.
	NumberOfSpacesWithRecharge-Point	*NumberOfVehicles*	0:1	Number of parking spaces with electrical recharge points specified by this PARKING CAPACITY.

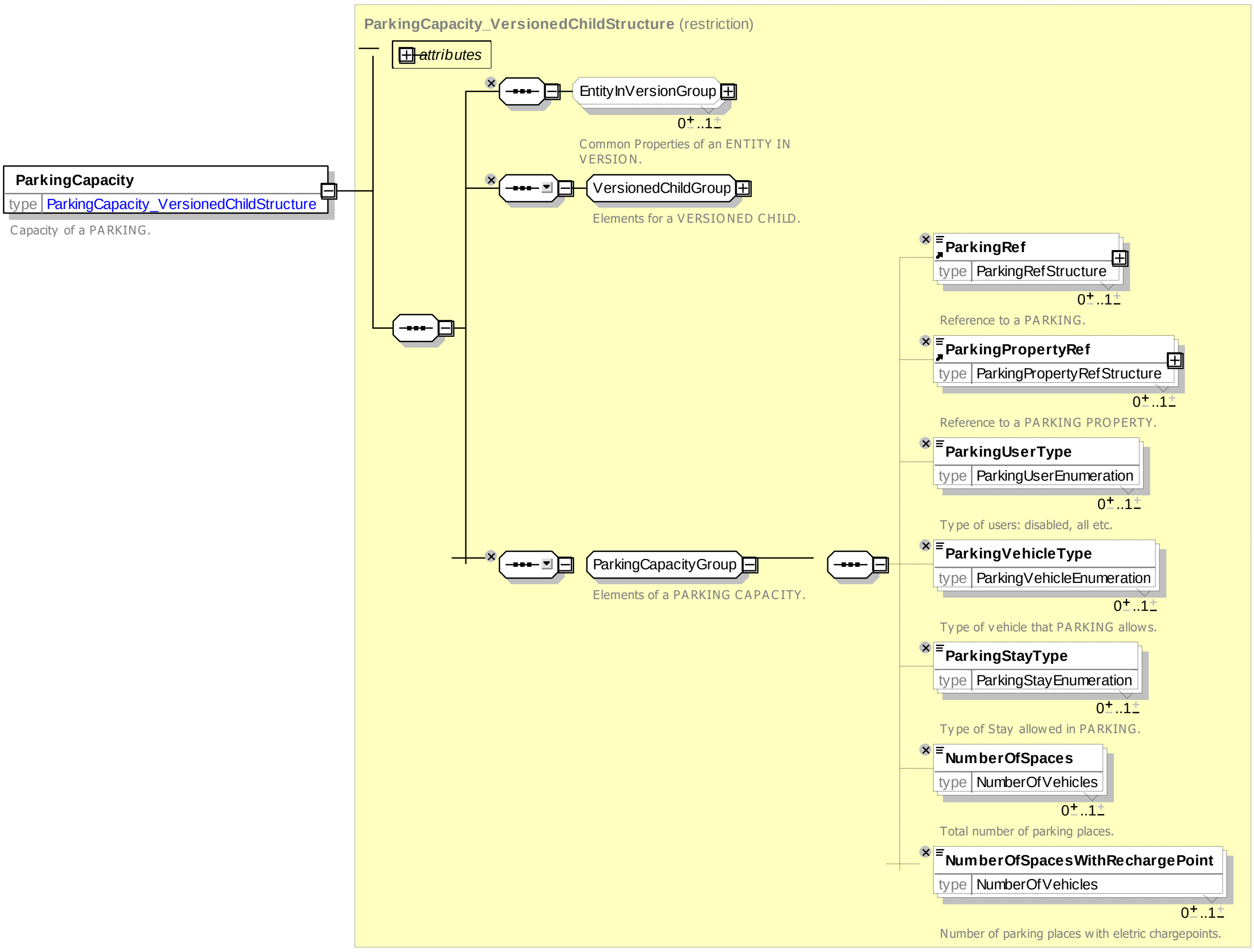


Figure 685 – *ParkingCapacity – XSD*

8.4.12.4 XML Example of Parking

The following XML code fragment shows part of a PARKING linked to a Stop Place by a ramp

EXAMPLE

```
<Parking>
    <ResponsibilitySetRef>napt:RS_80</ResponsibilitySetRef id="tbd:9100WIMBLDN_A2b_P1”>
    <Name>Station Parking</Name>
    <Description>Forecourt parking</Description>
    <Centroid>
        <Location>
            <Longitude>-180</Longitude>
            <Latitude>-90</Latitude>
        </Location>
    </Centroid>
    <PublicUse>true</PublicUse>
    <Covered>outdoors</Covered>
    <Lighting>wellLit</Lighting>
    <pathLinks>
        <SitePathLink>
            <Name>Link from parking to forecourt</Name>
            <From><PlaceRef>tbd:9100WIMBLDN_P1</PlaceRef></From>
            <To><PlaceRef>tbd:9100WIMBLDN_FC-J1</PlaceRef></To>
```

```xml
            <Description>From car park to forecourt path junction 1</Description>
            <Distance>10 </Distance>
            <NumberOfSteps>0</NumberOfSteps>
            <FromToUpDown>up</FromToUpDown>
            <AccessFeatureType>ramp</AccessFeatureType>
            <TransferDuration>
                <DefaultDuration>PT1M</DefaultDuration>
            </TransferDuration>
            <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
            <placeEquipments>
                <RampEquipment id="tbd:9100WIMBLDN_P1_Eq-1">
                    <Name>Ramp from car park</Name>
                    <Width>2</Width>
                    <DirectionOfUse>both</DirectionOfUse>
                    <Length>40</Length>
                    <GradientType>gentle</GradientType>
                    <HandrailType>none</HandrailType>
                </RampEquipment>
            </placeEquipments>
        </SitePathLink>
    </pathLinks>
    <ParkingType>trainStationParking</ParkingType>
    <ParkingLayout>openSpace</ParkingLayout>
    <TotalCapacity>4</TotalCapacity>
    <parkingProperties>
        <ParkingProperties id="tbd:9100WIMBLDN_P1_pp2">
            <ParkingUserType>allUsers</ParkingUserType>
            <ParkingStayType>shortStay</ParkingStayType>
            <MaximumStay>PT2H</MaximumStay>
            <spaces>
                <ParkingCapacity id="tbd:9100WIMBLDN_A2b_P1-p1">
                    <ParkingVehicleType>largeCar</ParkingVehicleType>
                    <ParkingStayType>shortStay</ParkingStayType>
                    <NumberOfSpaces>4</NumberOfSpaces>
                </ParkingCapacity>
            </spaces>
        </ParkingProperties>
    </parkingProperties>
</Parking>
```

8.4.13 Vehicle Stopping

8.4.13.1 VEHICLE STOPPING – Conceptual MODEL

The VEHICLE STOPPING MODEL describes the designated stopping positions for public transport VEHICLEs so that the VEHICLE doors are correctly aligned with the QUAYs and BOARDING POSITIONs. These are relevant for some real-time control systems.

A VEHICLE STOPPING PLACE such as a track in a station, may potentially align with one or more QUAY's either side of it, as rerpesented by a VEHICLE QUAY ALIGNMENT. It may also have different VEHICLE STOPPING POSITIONs for trains of different length and make-up.; each VEHICLE STOPPING POSITION can have a set of VEHICLE POSITION ALIGNMENTs to indicate the BOARDING POSITIONs that would be relevant for a VEHICLE TYPE (i.e. TRAIN) of a given kind.

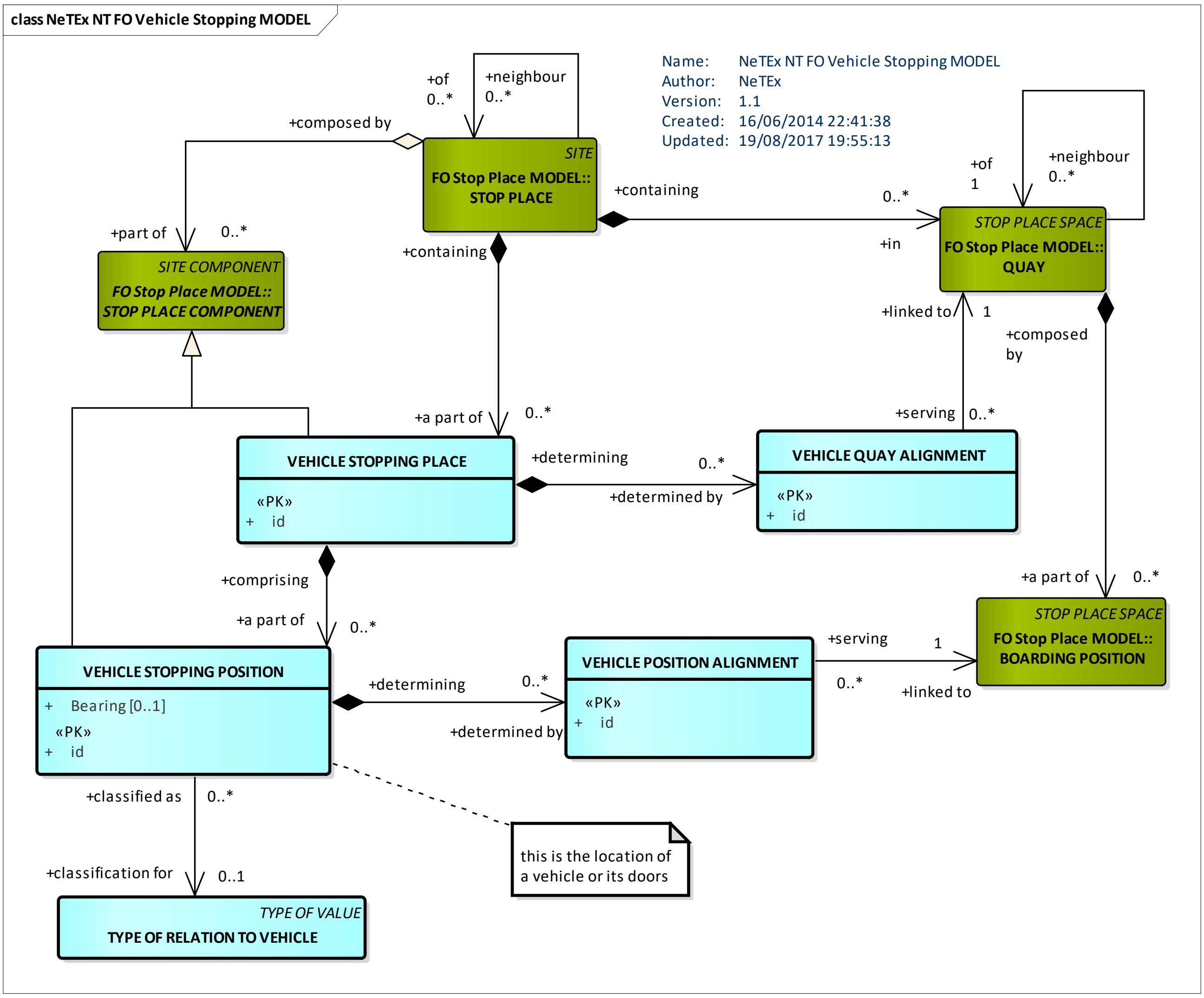


Figure 686 – VEHICLE STOPPING – Conceptual MODEL (UML)

8.4.13.2 Vehicle Stopping – Physical model

The following figure shows detailed attributes of the VEHICLE STOPPING model.

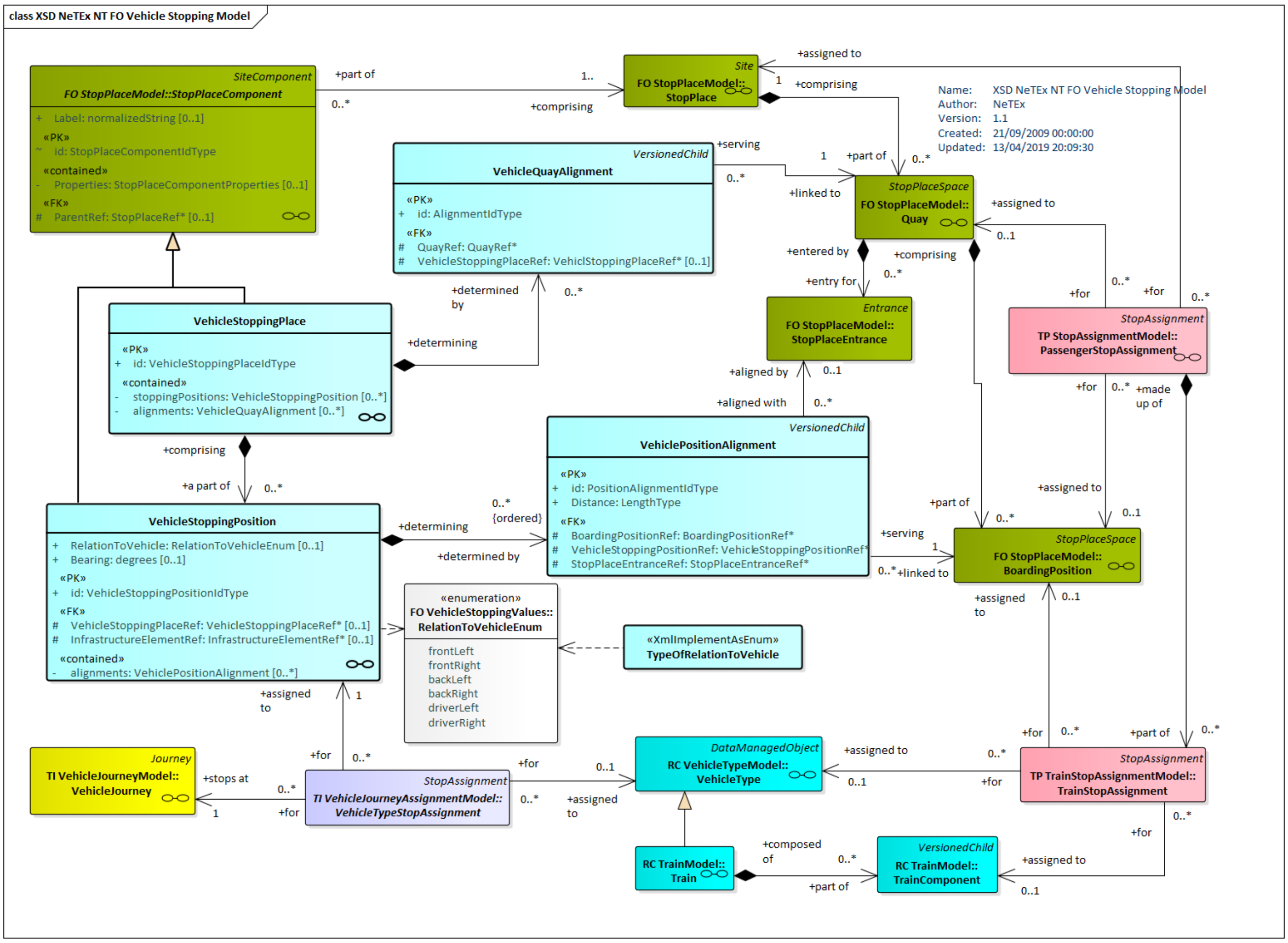


Figure 687 – Vehicle Stopping – Physical Model (UML)

8.4.13.3 Vehicle Stopping – Attributes and XSD

8.4.13.3.1 VehicleStoppingPlace – Model Element

A place on the vehicle trackway where vehicles stop in order for passengers to board or alight from a vehicle.

A vehicle trackway is located on the respective INFRASTRUCTURE LINK for the MODE (RAILWAY ELEMENT of rail network, ROAD ELEMENT of road network, etc.). A VEHICLE STOPPING PLACE may be served by one or more QUAYs.

Table 594 – *VehicleStoppingPlace* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*StopPlaceComponent*</u>	::>	VEHICLE STOPPING PLACE inherits from STOP PLACE COMPONENT.
«PK»	***id***	*VehicleStoppingPlace-IdType*	1:1	Identifier of VEHICLE STOPPING PLACE.
«FK»	***Infrastructure-LinkRef***	*(InfrastructureLinkRef) WireElementRef*	1:1	INFRASTRUCTURE LINK to which position relates.

		RoadElementRef | RailwayElementRef		
«FK»	***Infrastructure-PointRef***	*(InfrastructurePointRef) WirePoint | RoadPoint | RailwayPoint*	1:1	INFRASTRUCTURE POINT to which position relates.
«cntd»	***vehicleStopping-Positions***	*VehicleStoppingPosition*	0:*	VEHICLE STOPPING POSITIONs for VEHICLE STOPPING PLACE.
«cntd»	***quayAlignments***	*VehicleQuayAlignment*	0:*	Alignments for VEHICLE STOPPING PLACE.

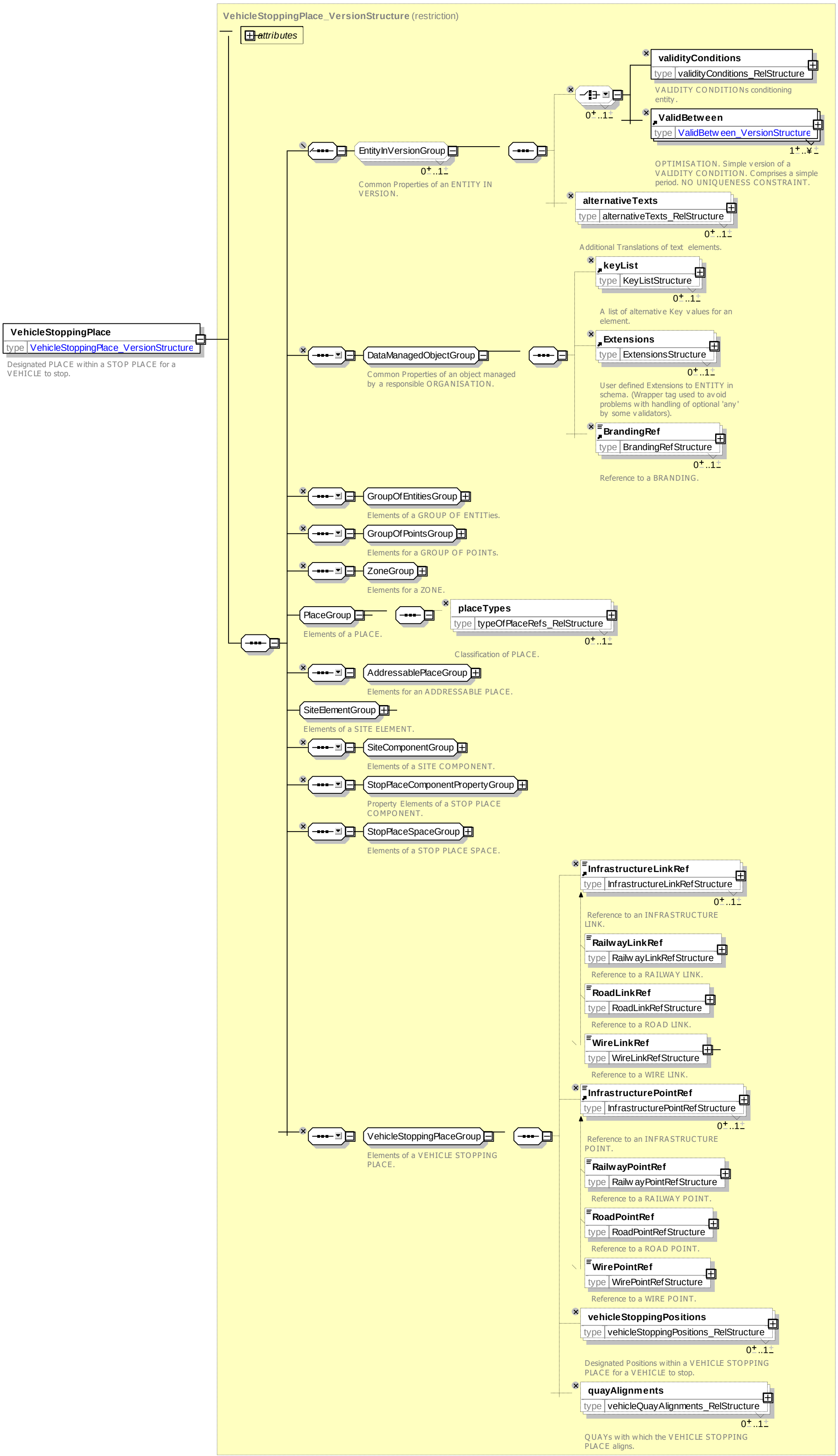

VehicleStoppingPlace_VersionStructure (restriction)
attributes
VehicleStoppingPlace
type VehicleStoppingPlace_VersionStructure
Designated PLACE within a STOP PLACE for a VEHICLE to stop.
EntityInVersionGroup
Common Properties of an ENTITY IN VERSION.
validityConditions
type validityConditions_RelStructure
VALIDITY CONDITIONs conditioning entity.
ValidBetween
type ValidBetween_VersionStructure
OPTIMISATION. Simple version of a VALIDITY CONDITION. Comprises a simple period. NO UNIQUENESS CONSTRAINT.
alternativeTexts
type alternativeTexts_RelStructure
Additional Translations of text elements.
DataManagedObjectGroup
Common Properties of an object managed by a responsible ORGANISATION.
keyList
type KeyListStructure
A list of alternative Key values for an element.
Extensions
type ExtensionsStructure
User defined Extensions to ENTITY in schema. (Wrapper tag used to avoid problems with handling of optional 'any' by some validators).
BrandingRef
type BrandingRefStructure
Reference to a BRANDING.
GroupOfEntitiesGroup
Elements of a GROUP OF ENTITies.
GroupOfPointsGroup
Elements for a GROUP OF POINTs.
ZoneGroup
Elements for a ZONE.
PlaceGroup
Elements of a PLACE.
placeTypes
type typeOfPlaceRefs_RelStructure
Classification of PLACE.
AddressablePlaceGroup
Elements for an ADDRESSABLE PLACE.
SiteElementGroup
Elements of a SITE ELEMENT.
SiteComponentGroup
Elements of a SITE COMPONENT.
StopPlaceComponentPropertyGroup
Property Elements of a STOP PLACE COMPONENT.
StopPlaceSpaceGroup
Elements of a STOP PLACE SPACE.
VehicleStoppingPlaceGroup
Elements of a VEHICLE STOPPING PLACE.
InfrastructureLinkRef
type InfrastructureLinkRefStructure
Reference to an INFRASTRUCTURE LINK.
RailwayLinkRef
type RailwayLinkRefStructure
Reference to a RAILWAY LINK.
RoadLinkRef
type RoadLinkRefStructure
Reference to a ROAD LINK.
WireLinkRef
type WireLinkRefStructure
Reference to a WIRE LINK.
InfrastructurePointRef
type InfrastructurePointRefStructure
Reference to an INFRASTRUCTURE POINT.
RailwayPointRef
type RailwayPointRefStructure
Reference to a RAILWAY POINT.
RoadPointRef
type RoadPointRefStructure
Reference to a ROAD POINT.
WirePointRef
type WirePointRefStructure
Reference to a WIRE POINT.
vehicleStoppingPositions
type vehicleStoppingPositions_RelStructure
Designated Positions within a VEHICLE STOPPING PLACE for a VEHICLE to stop.
quayAlignments
type vehicleQuayAlignments_RelStructure
QUAYs with which the VEHICLE STOPPING PLACE aligns.

Figure 688 – *VehicleStoppingPlace* – *XSD*

8.4.13.3.2 *VehicleStoppingPosition* – Model Element

The stopping position of a VEHICLE or one of its components as a location. May be specified as a ZONE corresponding to the bounding polygon of the VEHICLE, or one or more POINTs corresponding to parts of the vehicle such as a door.

If given as a single point, indicates the position for the door relative to an indicated side of the VEHICLE.

Table 595 – *VehicleStoppingPosition* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*StopPlaceComponent*	*::>*	VEHICLE STOPPING POSITION inherits from STOP PLACE COMPONENT.
«PK»	***id***	*VehicleStoppingPositionId-Type*	1:1	Identifier of VEHICLE STOPPING POSITION.
«FK»	***VehicleStopping-PlaceRef***	*VehicleStoppingPlaceRef*	1:1	VEHICLE STOPPING PLACE to which position relates.
	Label	*MultilingualString*	0:1	Label for VEHICLE STOPPING POSITION.
«cntd»	***RelationToVehicle***	*RelationToVehicleEnum*	0:1	Relation to VEHICLE of stopping position. See allowed values below.
	Bearing	Xsd:integer	0:1	Bearing of position relative to North.
«cntd»	***alignments***	*VehiclePositionAlignment*	0:*	Alignments for VEHICLE STOPPING POSITION.

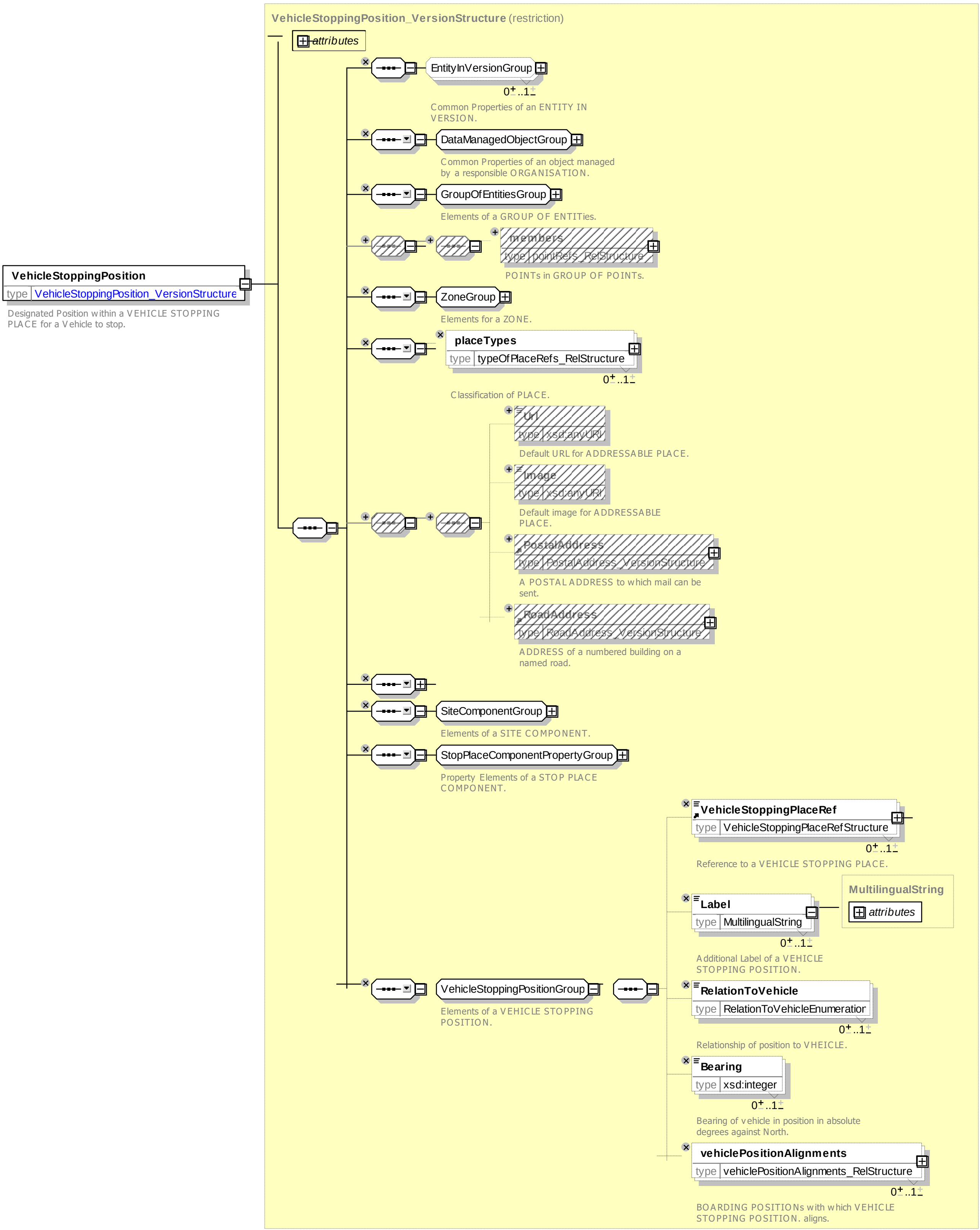


Figure 689 – *VehicleStoppingPosition – XSD*

8.4.13.3.2.1 RelationToVehicle – Allowed values

Allowed values for ***RelationToVehicle*** (*RelationToVehicleEnum*).

Table 596 – *RelationToVehicle* – Allowed Values

Value	Description
frontLeft	Front left of vehicle.
frontRight	Front right of vehicle.
backLeft	Back left of vehicle.
backRight	Back right of vehicle.
driverLeft	Driver left of vehicle.
driverRight	Driver right of vehicle.

8.4.13.3.3 VehicleQuayAlignment – Model Element

The alignment of a particular QUAY with a vehicle as the result of positioning a VEHICLE at a particular VEHICLE STOPPING PLACE.

NOTE A simple ***QuayRef*** is used to implement this.

Table 597 – *VehicleQuayAlignment*– Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*VersionedChild*</u>	::>	VEHICLE QUAY ALIGNMENT inherits from Versioned Child
«PK»	***id***	*AlignmentIdType*	1:1	Identifier of VEHICLE QUAY ALIGNMENT.
«FK»	***VehicleStopping-PositionRef***	*VehicleStopping-PositionRef*	0:1	VEHICLE STOPPING POSITION with which VEHICLE POSITION ALIGNMENT is aligned.
«FK»	***QuayRef***	*QuayRef*	1:1	Reference to a QUAY that is aligned with VEHICLE STOPPING POSITION.

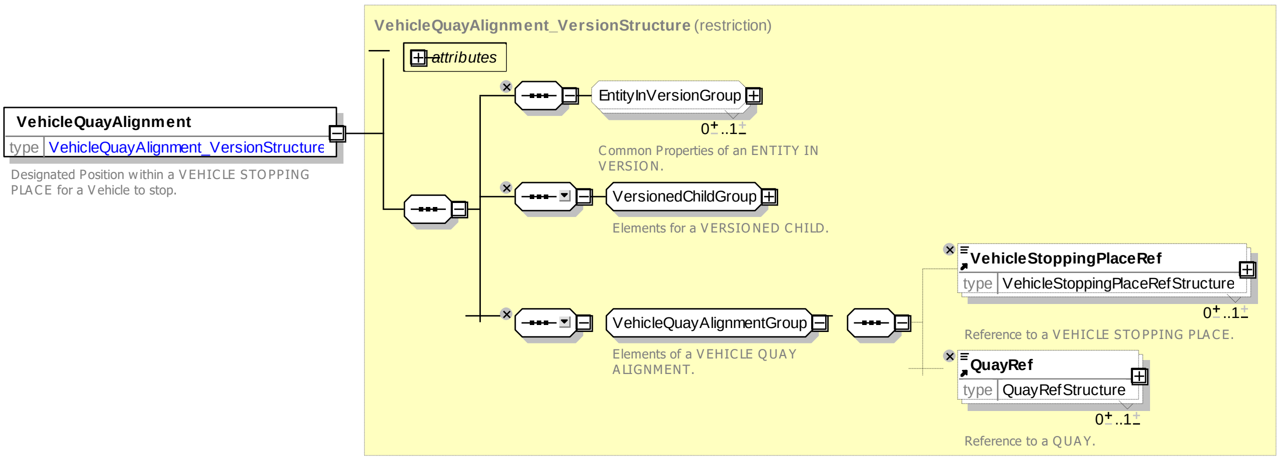


Figure 690 – Vehicle Quay Alignment – XSD

8.4.13.3.4 VehiclePositionAlignment – Model Element

The alignment of a particular BOARDING POSITION with the entrance of a VEHICLE as the result of positioning the VEHICLE at a particular VEHICLE STOPPING PLACE.

NOTE A simple ***BoardingPositionRef*** is used to implement this.

Table 598 – *VehiclePositionAlignment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*VersionedChild*	::>	VEHICLE POSITION ALIGNMENT inherits from Versioned Child.
«PK»	***id***	*PositionAlignmentIdType*	*1:1*	Identifier of VEHICLE POSITION ALIGNMENT.
	Distance	*LengthType*	*0:1*	Distance along track from VEHICLE STOPPING POSITION.
«FK»	***VehicleStopping-PositionRef***	*VehicleStopping-PositionRef*	*0:1*	VEHICLE STOPPING POSITION with which VEHICLE POSITION ALIGNMENT is aligned.
«FK»	***BoardingPosition Ref***	*BoardingPositionRef*	*0:1*	BOARDING POSITION with which VEHICLE POSITION ALIGNMENT is aligned.
«FK»	***StopPlace-EntranceRef***	*StopPlace-EntranceRef*	*01:1*	ENTRANCE to BOARDING POSITION with which VEHICLE POSITION ALIGNMENT is aligned.

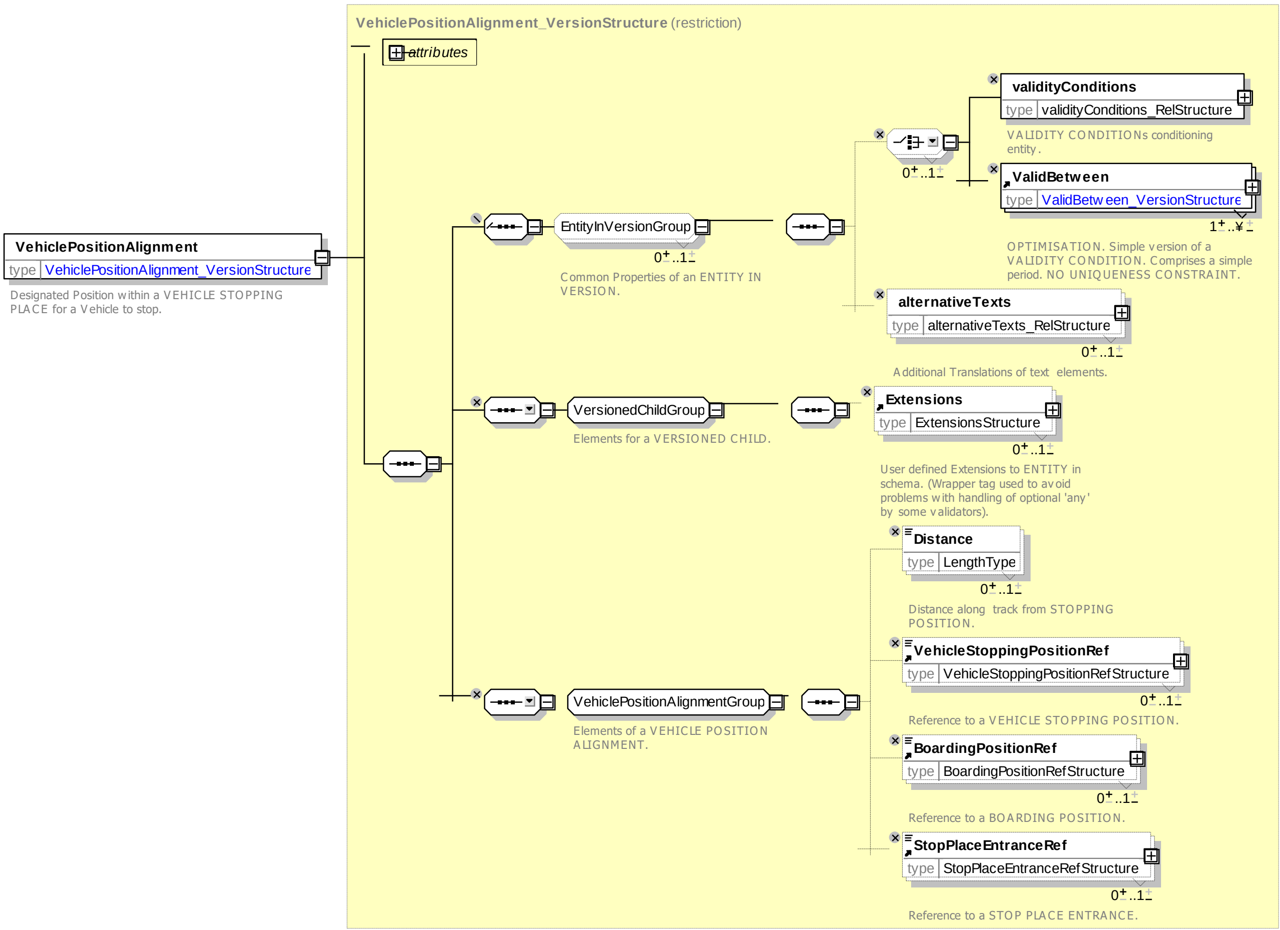


Figure 691 – Vehicle Position Alignment – XSD

8.4.14 Accessibility Coverage

All SITE ELEMENTs, such as ENTRANCEs, QUAYs, ACCESS SPACEs and STOP PLACEs should be given basic ACCESSIBILITY LIMITATION attributes for each of the five standard criteria (*Wheelchair, LiftFree, StepFree, EscalatorFree, TravelatorFree* as described in the section on accessibility.

Each of these take one of three values '*true', 'false',* or *'unknown'.*

It is important to distinguish between absence of data and absence of accessibility, so if no data is available an element should nonetheless be tagged as '*unknown'*.

8.4.14.1 Accessibility Coverage of Site Elements

8.4.14.1.1 Site Accessibility Coverage

A STOP PLACE should be classified as one of the three values.

— A STOP PLACE is accessible (*true*) for a given criterion if **all** of its QUAYs can be reached from an external entrance by at least one NAVIGATION PATH that fulfils that criterion.

— A STOP PLACE is **not** accessible (*false*) STOP PLACE for a given criterion if at least one of its QUAYs **cannot** be reached from an external entrance by at least one NAVIGATION PATH that fulfils that criterion.

— SITEs and other STOP PLACEs should be stated as accessible '*unknown'* unless explicitly known otherwise.

8.4.14.1.2 Quay and Access Space Accessibility Coverage

SITE COMPONENTs within a STOP PLACE should be classified as one of the three values:

— A QUAY or ACCESS SPACE (or other SITE COMPONENT) is accessible (true) for a given criterion if it can be reached from an external entrance by at least one NAVIGATION PATH that fulfils that criterion.

— A QUAY or ACCESS SPACE (or other SITE COMPONENT) is not accessible for a given criterion (false) if it cannot be reached from an external entrance by at least one NAVIGATION PATH that fulfils that criterion.

8.4.14.1.3 Defaulting Accessibility Values for Site Components

- On street QUAYs should be stated as accessible '*true'* unless known otherwise.
- Off street QUAYs (e.g. stations), should be stated as accessible '*unknown'* unless explicitly known otherwise.

Table 599 – Accessibility Attributes for level 1

		Rail / Metro			**On Street Bus**
		STOP PLACE	**QUAY**		**QUAY**
Wheelchair		*unknown*	*unknown*		*true*
LiftFree		*unknown*	*unknown*		*true*
StepFree		*unknown*	*unknown*		*true*
EscalatorFree		*unknown*	*unknown*		*true*
TravelatorFree		*true*	*true*		*true*

8.4.15 Accessibility Coverage of Paths

8.4.15.1.1 Path Link Accessibility Coverage

PATH LINKs should be classified as one of the three values:

— A PATH LINK is accessible (true) for a given criterion if it can be traversed according to that criterion.

— A PATH LINK is not accessible (false) for a given criterion if it cannot be traversed according to that criterion.

— A PATH LINK is should be stated as accessible true unless known otherwise.

8.4.15.1.2 Navigation Path Accessibility Coverage

NAVIGATION PATHs should be classified as one of the two accessibility values:

— A NAVIGATION PATH is accessible (true) for a given criterion if it can be traversed along at least one branch according to that criterion.

— A NAVIGATION PATH is not accessible (false) for a given criterion if it cannot be traversed along any branch according to that criterion.

The accessibility of a NAVIGATION PATH can be derived from its PATH LINKs. The following figure shows an example of summarizing the individual links of an access path to derive the overall accessibility of a path. The second link involves steps – this sets the minimum accessibility of the whole path :

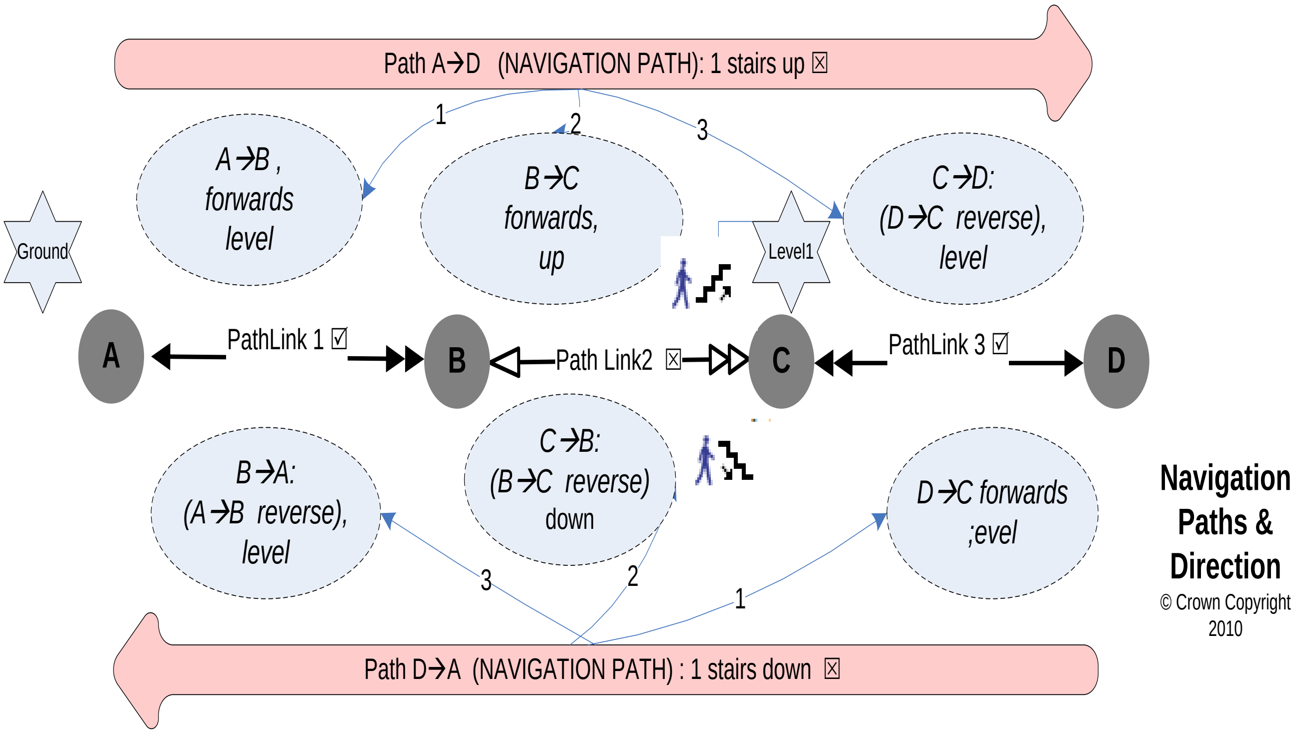


Figure 692– Example of simple NAVIGATION PATH accessibility

The following figure shows an example of the derivation of the accessibility of a branched NAVIGATION PATH. One branch requires the use of steps, the other of an escalator. The NAVIGATION PATH may thus be described overall as having Lift Free and Escalator Free (If lift is used) access but is not wheelchair accessible.

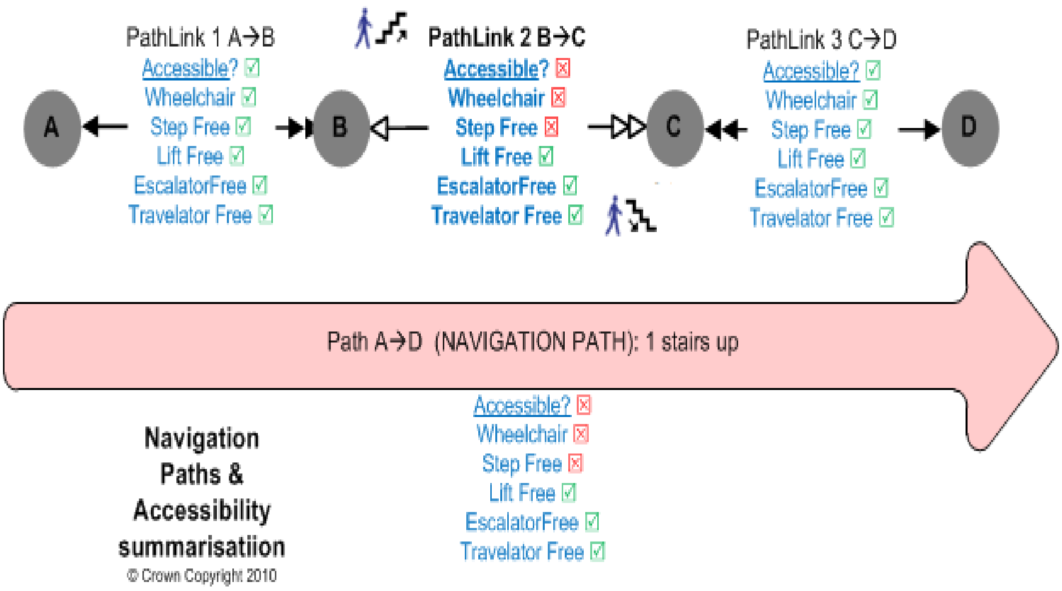


Figure 693– Example Accessibility Criteria for a Navigation path

8.4.15.1.3 Default Accessibility values for a Navigation Path

NAVIGATION PATHs should be stated as accessible ***unknown*** unless explicitly known otherwise.

8.5 Tactical Planning Components – Subsystem

8.5.1 Tactical Planning – Model Dependencies

The TACTICAL PLANNING model provides reusable information about transport planning, such as spatial description of journey patterns and service patterns. Reusable journey patterns and service patterns are independent of actual operating times in scheduled journeys.

In most transport networks the scheduled journeys follow the same patterns of movement and the TACTICAL PLANNING model allows these to be described as reusable components in their own right. The elements defined in the TACTICAL PLANNING model are subsequently used in the JOURNEY TIMES model to specify actual VEHICLE JOURNEYs at particular times.

The submodels themselves depend on a number of general NeTEx framework models (for example, GENERIC POINT AND LINK, NOTICE, etc.,) described elsewhere – See GENERIC FRAMEWORK in 7.5.

Figure 694 shows the dependencies between the TACTICAL PLANNING physical submodels. The payload elements are contained in the following packages, which may be exchanged using a SERVICE FRAME (see earlier):

— JOURNEY PATTERN: models the pattern of POINTs and LINKs of a Transport NETWORK that its services follow including both stops and other points of operational interest.

— SERVICE CONNECTION: models the possible connections allowed at points in the network.

— SERVICE PATTERN: Models the subset of the JOURNEY PATTERN that is relevant to passengers.

— ROUTING CONSTRAINT: models restrictions on connections between journeys such as ‘CannotBoardAndAlightInSameZone’, ‘MustAlightInZone’, ‘MustBoardInZone’.

— TIME DEMAND TYPE: characterises the different temporal contexts in which services may run such as ‘*peak’, ‘off peak’*, etc.

— STOP ASSIGNMENT, TRAIN STOP ASSIGNMENT, and NAVIGATION PATH ASSIGNMENT: model the relationship between stops in the timetable and the physical platforms of an actual station or other stop.

— NOTICE ASSIGNMENT: models the association of footnotes and passenger information content such as stop announcements and the network.

— PASSENGER INFORMATION EQUIPMENT: models the relationship between passenger display equipment and the services to be shown on them.

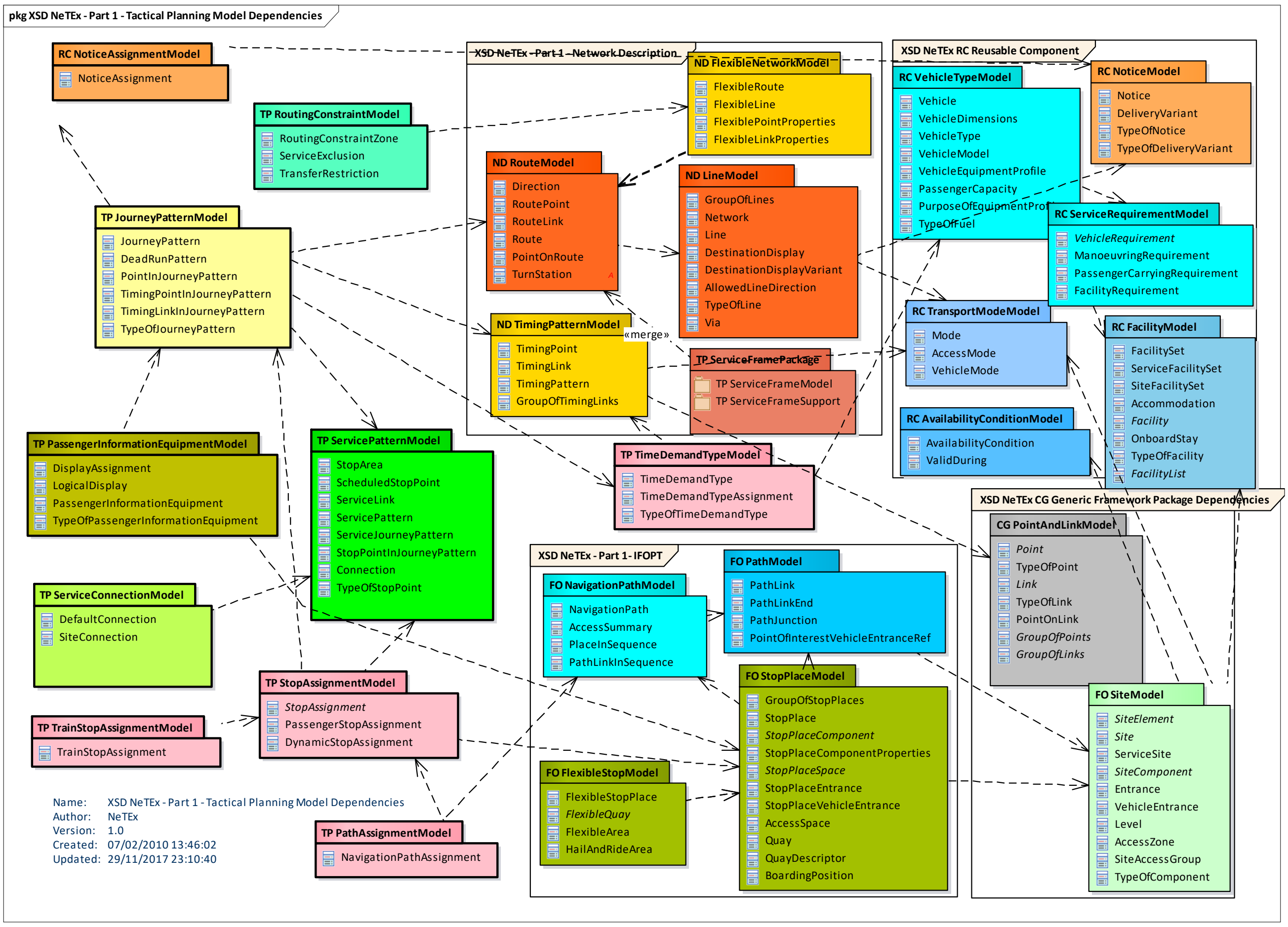


Figure 694 – Tactical Planning – Model dependencies (UML)

8.5.2 Connections & Transfer times

Most journey planners allow transfer times for an interchange to be specified to some degree – usually as default exchange times to use either all interchanges or at a specific station. The *NeTEx* model allows a set of TRANSFER DURATIONs for journey planning to be exchanged with successive levels of precision, for example:

1. DEFAULT CONNECTION – a default time for a transfer between modes on any SITE or given OPERATOR, to be used if there is no more specific value for a site.

2. ACCESS link – the possibility of a transfer between any two points or places. This can be used to state the best STOP PLACE to use to reach a particular a POINT of INTEREST or other distinct SITE.

   — An ACCESS allows a default time for making a transfer between two sites to be specified – this will be regardless of the time needed to reach a particular point within a large site. This can be used to state the average time needed to reach a POINT of INTEREST or other distinct SITE.

3. CONNECTION – The possibility of making a connection between two SCHEDULED STOP POINTs or STOP AREAs. It allows a default time for a transfer between two SCHEDULED STOP POINTs or STOP AREAs. This allows logical connections in the timetable to be computed independently of a STOP PLACE model, for example, '*King's Cross to St Pancras International'*.

   — A CONNECTION can also be used to state an average contingency time to change at a given interchange regardless of the actual point to point transition – by making the 'from' and the 'to' SCHEDULED STOP POINTs the same. (Some journey planners support only this level of precision).

   — A CONNECTION can also be used to state an average contingency time to change at a given interchange between any two modes regardless of the actual point to point transition – by using the 'from' and the 'to' SCHEDULED STOP POINTs for the respective modes.

4. SITE CONNECTION – The possibility of making a connection between two SITE / SITE COMPONENTs and / or SCHEDULED STOP POINTs and STOP AREAs. Used to define points of connection between areas of a SITE for reaching public transport. It allows a default time for a transfer between a part of a SITE (which may also correspond to a SCHEDULED STOP POINTs or STOP AREAs).

In addition a third level of precision is possible:

5. A NAVIGATION PATH may state a transfer time for using a specific path to make a transfer between two physical points within the context of a SITE. For example '*District Line Platform 1 to Tramlink Platform 10'* via lift, allowing a very detailed calculation of journey times for a specific accessibility constraint, if desired.

   — Each PATH LINK may have a TRANSFER DURATION specified on it.

   — A NAVIGATION PATH may have a total TRANSFER DURATION – this should be the sum of the individual links, if present.

   — There can be more than one NAVIGATION PATH between the same points with different times.

   — A NAVIGATION PATH may reference an ACCESS or CONNECTION for which it provides more detailed information. Several different NAVIGATION PATHs may be associated with the same CONNECTION, representing alternative paths.

CONNECTION times are typically created as part of tactical planning of routes and timetables. NAVIGATION PATH times are derived from a bottom up assessment of the Physical STOP PLACE.

The following should be emphasized: CONNECTION transfer times relate to the timetabled connection times (and can be used without reference to actual platforms). NAVIGATION PATH transfer times relate to the known times to traverse between the physical stop. Whilst these may be the same, they are not necessarily so.

8.5.2.1 SERVICE CONNECTION – Conceptual MODEL

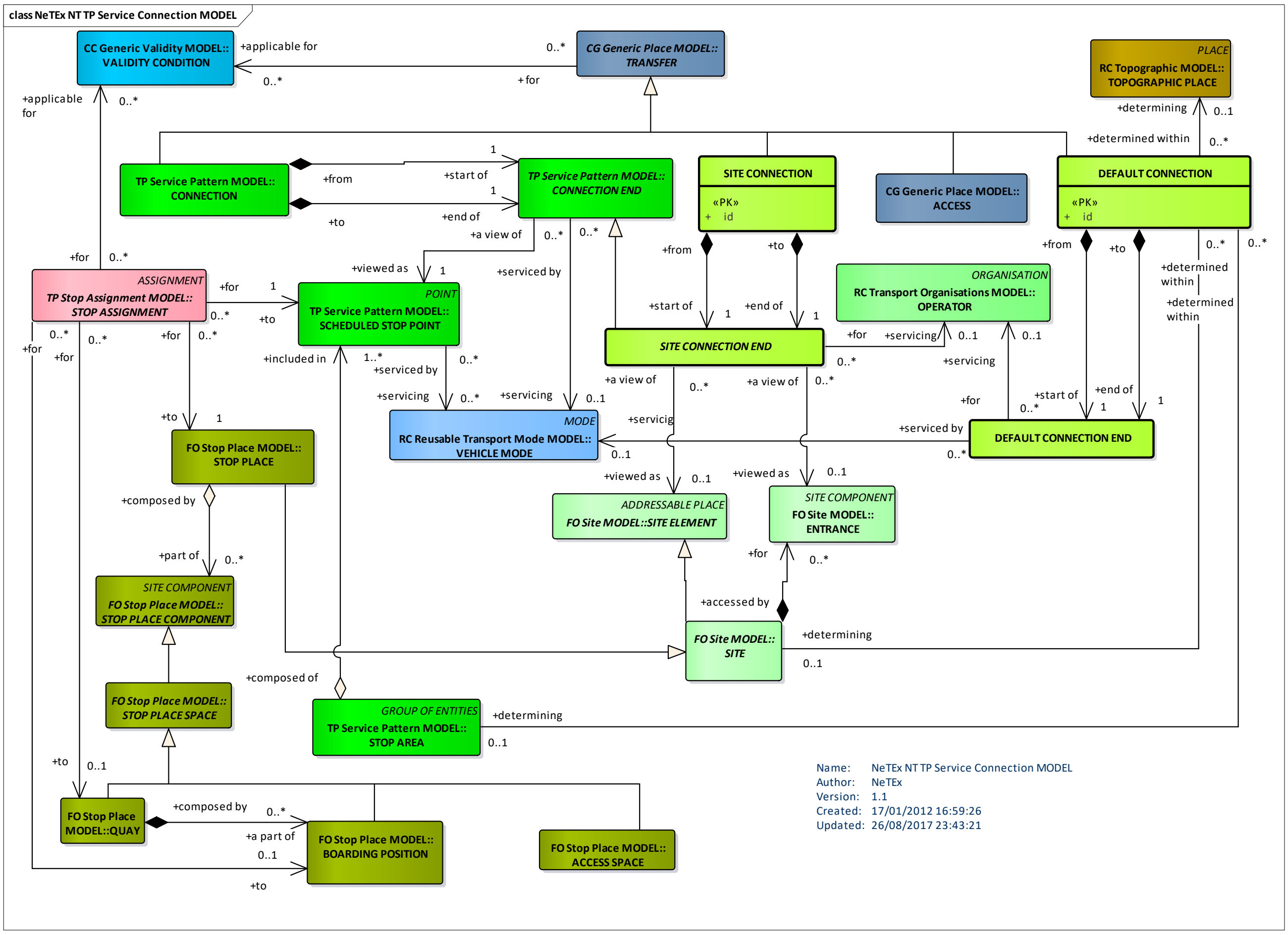


Figure 695 – SERVICE CONNECTION – Conceptual MODEL (UML)

8.5.2.2 Service Connection – Physical MODEL

8.5.2.2.1 Service Connection Overview– Physical MODEL

The following figure gives an overview of the SERVICE CONNECTION model elements.

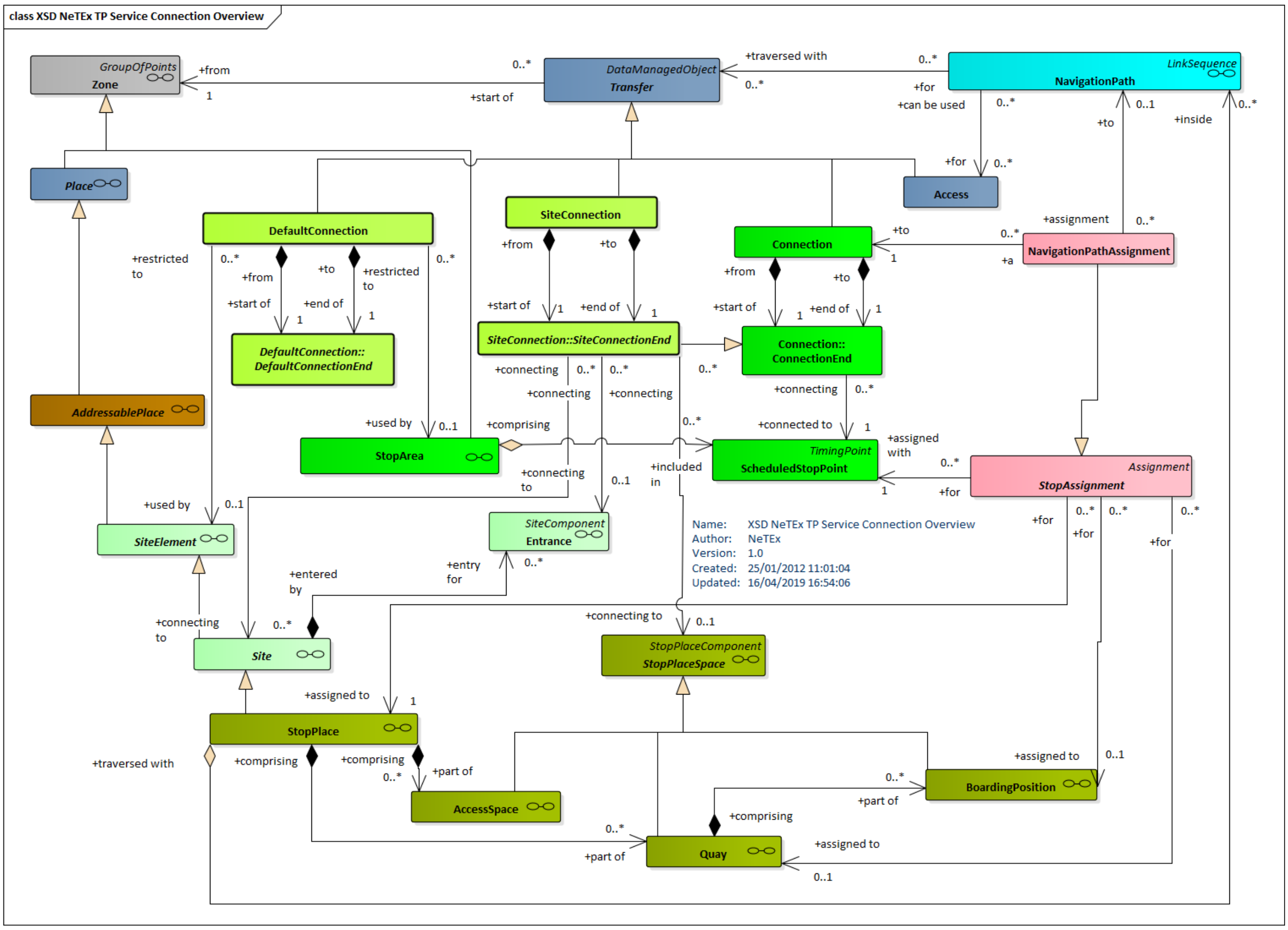


Figure 696 – Service Connection– Physical Model (UML) – Overview

8.5.2.2.2 Default Connection – Physical MODEL

The following figure gives an overview of the DEFAULT CONNECTION model elements.

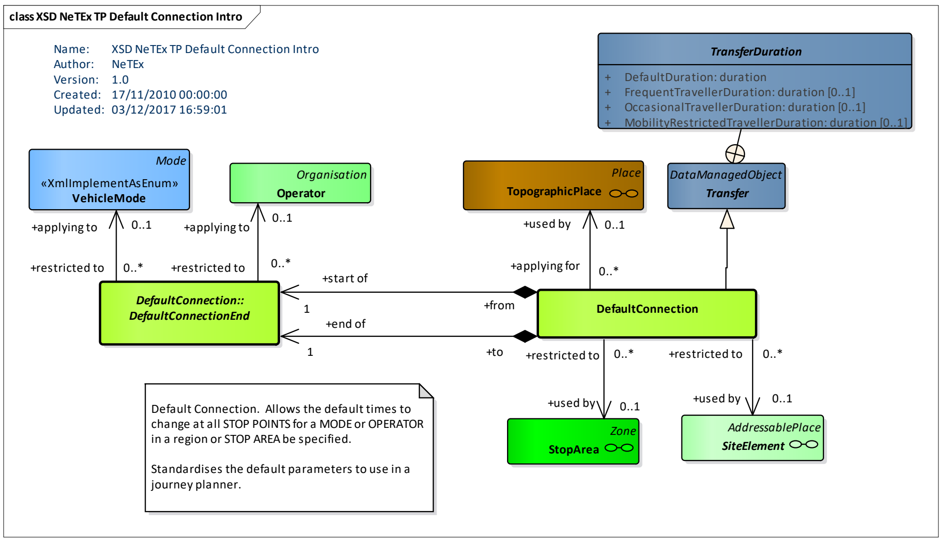


Figure 697 – Default Connection – Physical Model (UML) – Introduction

8.5.2.2.3 Service Connection – Physical MODEL

The following figure gives an overview of the SERVICE CONNECTION model elements.

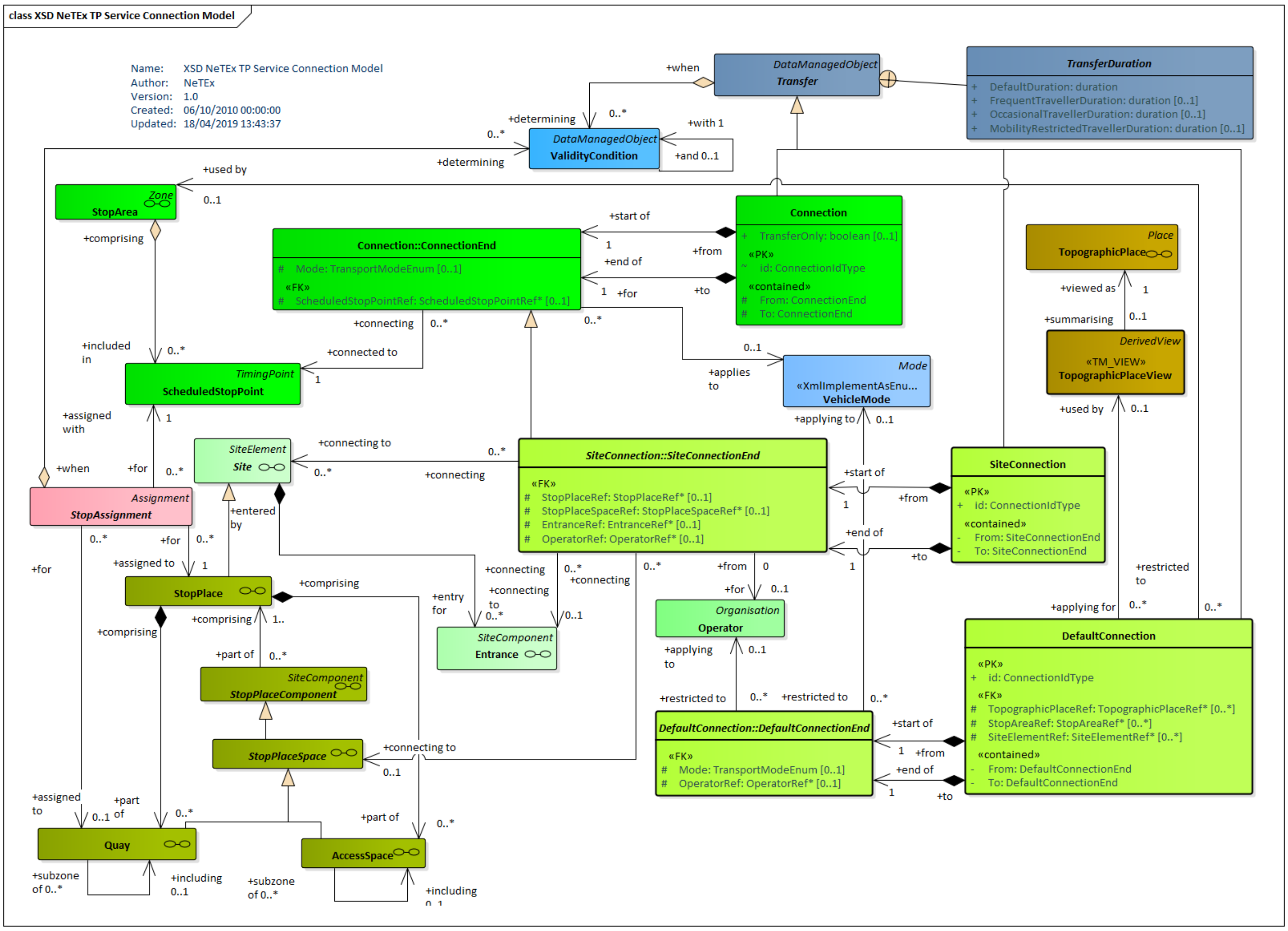


Figure 698 – Service Connection – Physical Model (UML) – Details

8.5.2.3 Example of Transfer Times

The following figure attempts to show the use of the different Transfer Times with an example based on Wimbledon Station. In the top half of the diagram are SCHEDULED STOP POINTs and CONNECTION LINKs. There are in fact two separate rail SCHEDULED STOP POINTs for Wimbledon. In the bottom half of the model some of the QUAYs, PATH LINKS (simplified) and a few NAVIGATION paths.

It can be seen that different Transfer times can be specified for timetabling between the SCHEDULED STOP POINTs and for navigation between the STOP PLACE COMPONENTs.

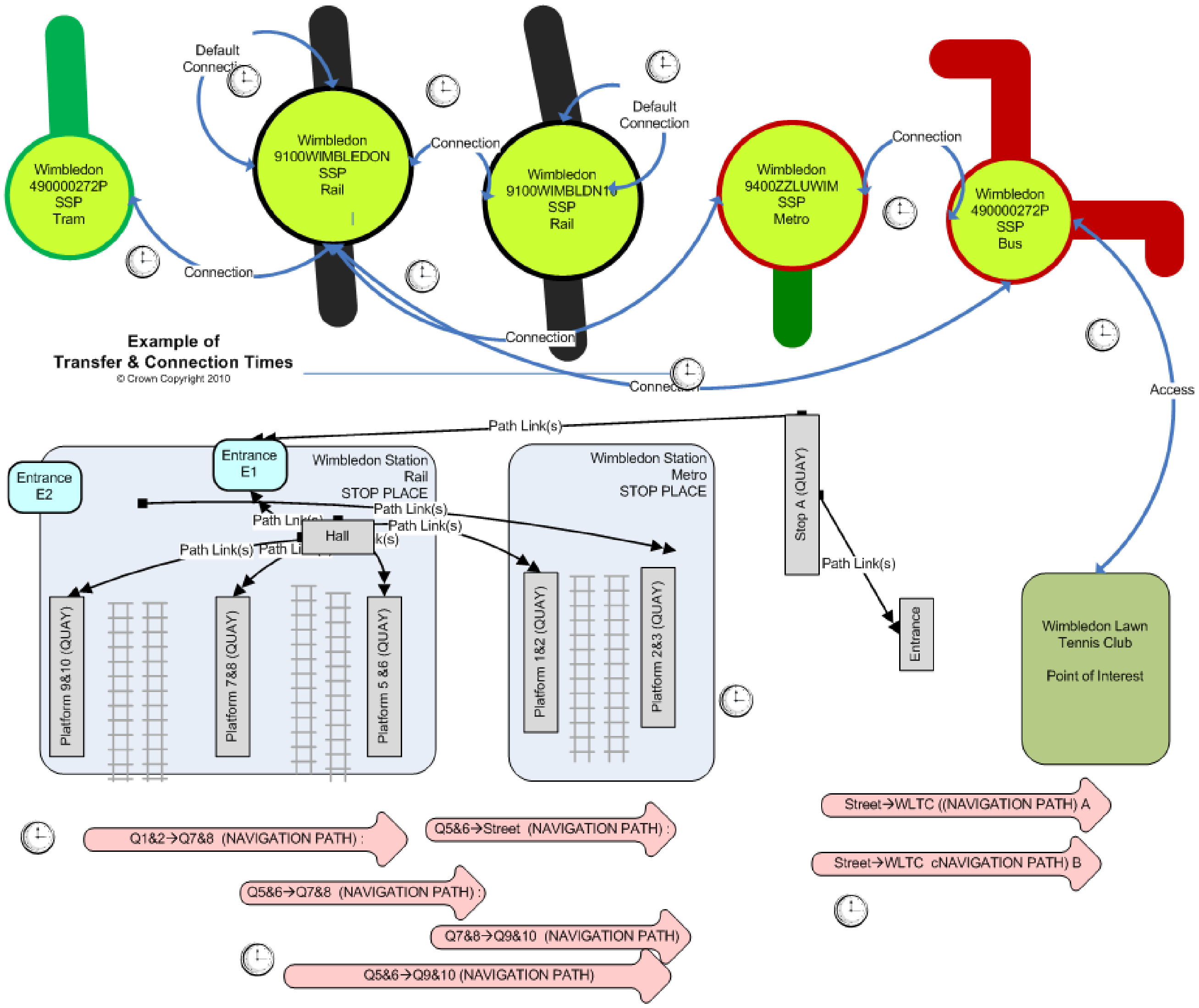


Figure 699 – Example of Transfer and connection times

8.5.2.4 Types of Transfer times

Up to four different transfer times may be specified in a given TRANSFER DURATION, though it is usually sufficient to have a single time and to use weighting factors to derive the others.

Table 600 – Transfer times in a TRANSFER DURATION

Value	Definition
DefaultDuration	Default average transfer time.
FrequentTravellerDuration	Transfer time for a traveller familiar with the interchange.
OccasionalTravellerDuration	Transfer time for a traveller unfamiliar with the interchange.
MobilityRestrictedTravellerDuration	Transfer time for a mobility impaired traveller.

Two sets of times of travel times are available

— A default set to use for journey planning.

— An additional set to use for walk times, if different from the journey planning set.

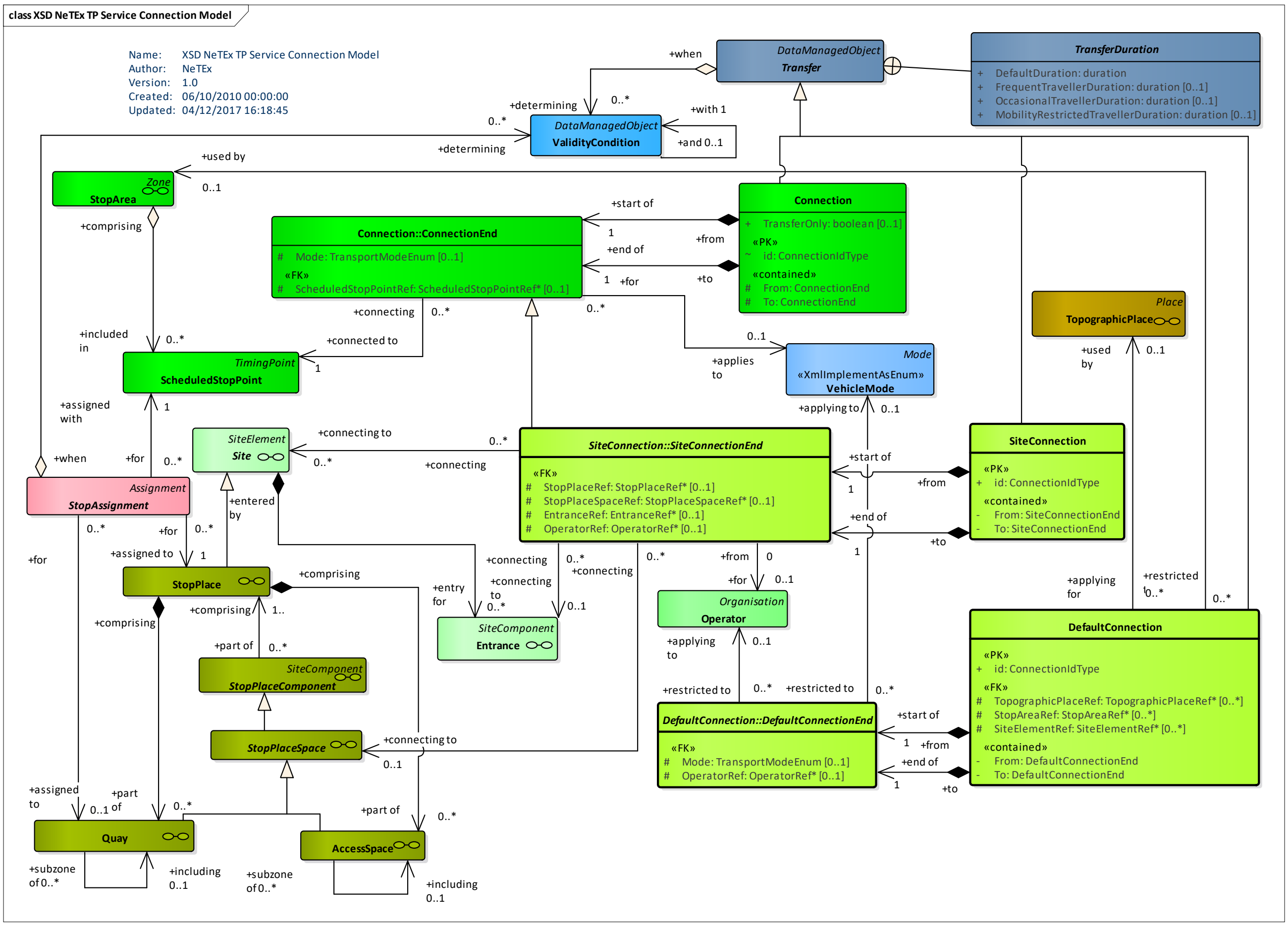


Figure 700 – Service Connection – Physical Model (UML) – Detail

8.5.2.5 Transfer times between Places

A set of transfer times between any two places can be specified on a TRANSFERs and so on its specialisation.

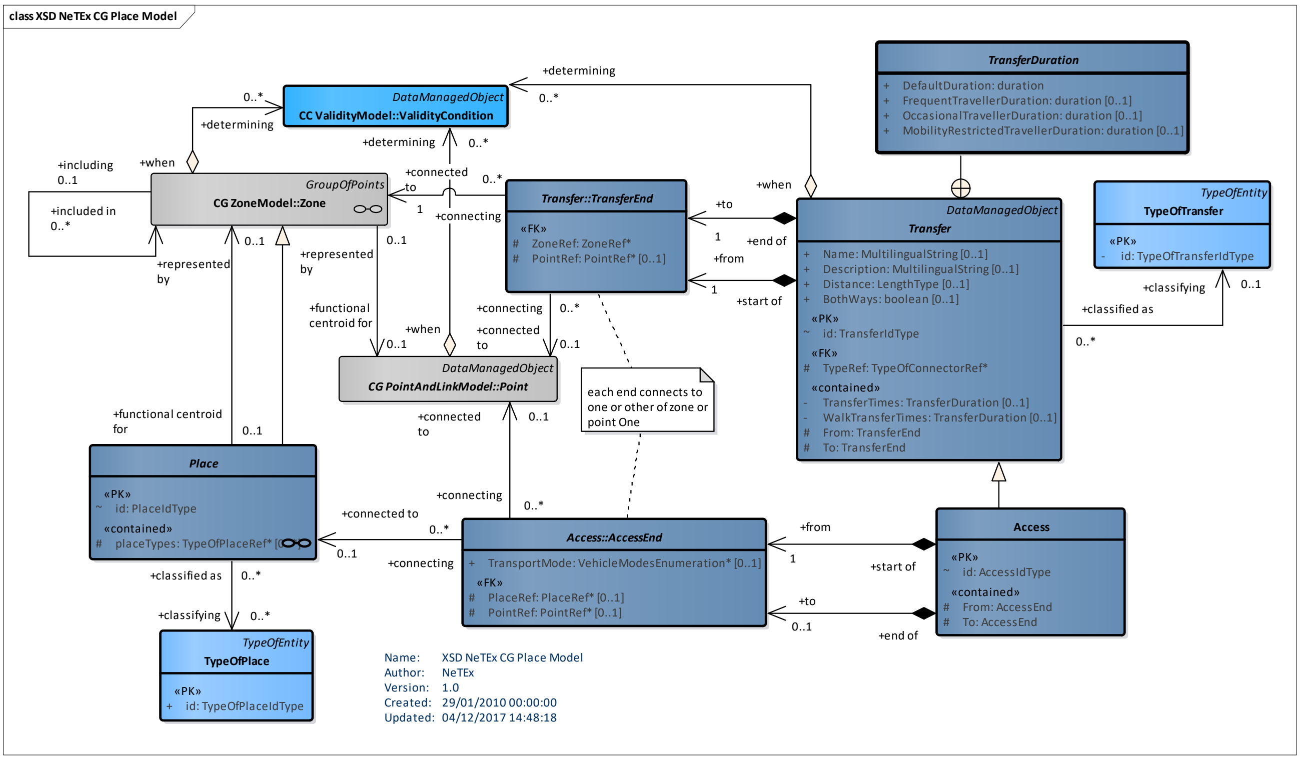


Figure 701 – Transfer times – Physical Model (UML)

8.5.2.6 Transfer times between Site components

A set of transfer times between any two parts of a SITE can be specified on a NAVIGATION PATH. The times for a single link can also be specified on an individual PATH LINK.

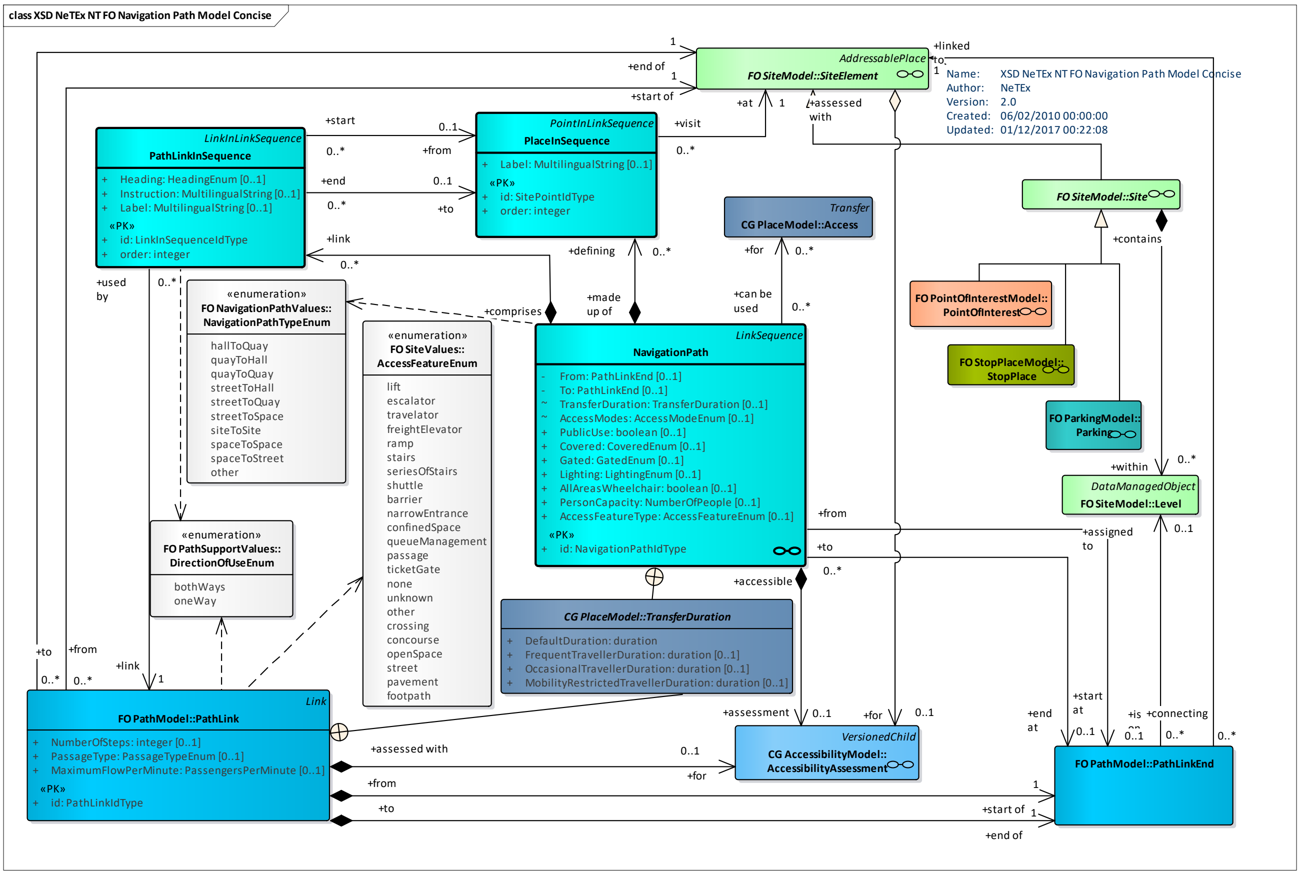


Figure 702 – Site components with Access times – Physical Model (UML) – Detail

8.5.2.7 Connections & Transfer Times – Attributes and XSD

8.5.2.7.1 *DefaultConnection* – Model Element

Specifies default times to be used to change from one mode of transport or another at an area or national level as specified by a TOPOGRAPHIC PLACE or SITE. May be restricted to a specific MODE or OPERATOR or only apply in a particular direction of transfer, e.g. bus to rail may have a different time for rail to bus.

Table 601 – *DefaultConnection* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Transfer*	::>	DEFAULT CONNECTION inherits from TRANSFER.
«PK»	***id***	*DefaultConnection-IdType*	1:1	Identifier of DEFAULT CONNECTION.
«cntd»	***From***	*DefaultConnectionEnd*	1:1	Start end of DEFAULT CONNECTION.
«cntd»	***To***	*DefaultConnectionEnd*	1:1	Destination end of DEFAULT CONNECTION.
«FK»	***TopographicPlace-View***	*TopographicPlaceRef*	0:*	TOPOGRAPHIC PLACE to which DEFAULT CONNECTION applies.
«FK»	***StopAreaRef***	*StopAreaRef*	0:*	STOP AREA to which DEFAULT CONNECTION applies.

«FK»	***SiteElementRef***	*(SiteElementRef)*	0:*	SITE ELEMENT to which DEFAULT TRANSFER applies.

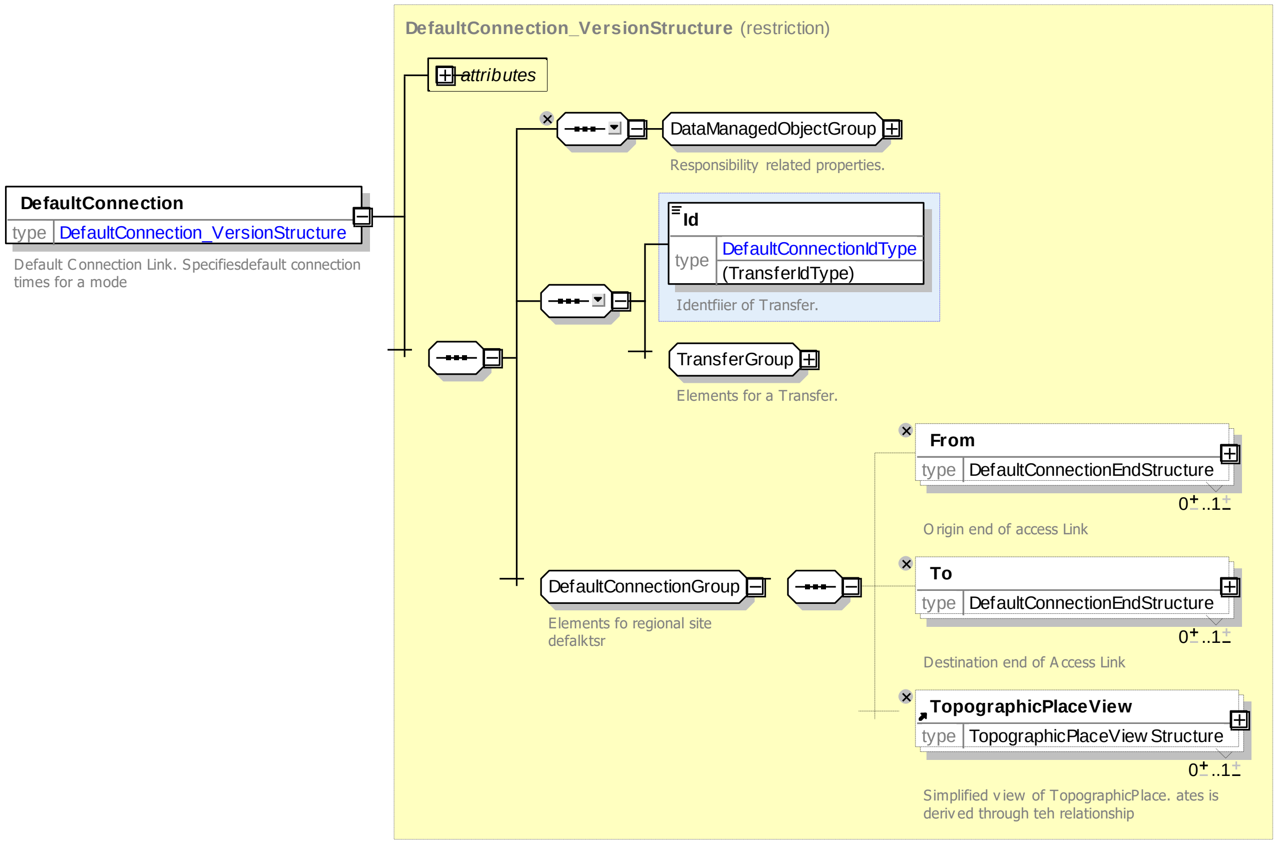


Figure 703 – *DefaultConnection – XSD*

8.5.2.7.2 DefaultConnectionEnd – Model Subelement

One end of a DEFAULT TRANSFER.

Table 602 – *DefaultConnectionEnd* – Structure

Classification	Name	Type	Cardinality	Description
«enum»	***Mode***	*TransportModeEnum*	0:1	TRANSPORT MODE for DEFAULT TRANSFER. See TRANSPORT MODE for alloeed values
«FK»	***OperatorRef***	*OperatorRef*	0:1	Reference to OPERATOR of DEFAULT TRANSFER

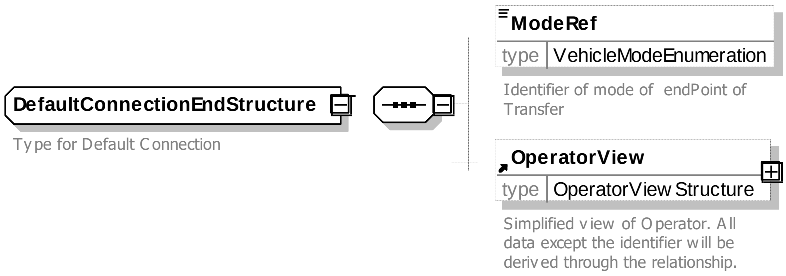


Figure 704 – *DefaultConnectionEnd – XSD*

8.5.2.7.3 SiteConnection – Model Element

The physical (spatial) possibility for a passenger to change from one public transport vehicle to another to continue the trip. The ends of connection can be specified as SCHEDULED STOP POINT or STOP AREA. Optionally this may be additionally qualified with physical STOP PLACE. Different times may be necessary to cover this link, depending on the kind of passenger.

Table 603 – *SiteConnection* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Transfer*	*::>*	SITE CONNECTION inherits from TRANSFER.
«PK»	***id***	*ConnectionIdType*	1:1	Identifier of SITE CONNECTION link.
«cntd»	***From***	*SiteConnectionEnd*	1:1	Start end of SITE CONNECTION link.
«cntd»	***To***	*SiteConnectionEnd*	1:1	Destination end of SITE CONNECTION link.
«cntd»	***navigationPaths***	*NavigationPath*	0:1	NAVIGATION PATHs between SITEs.

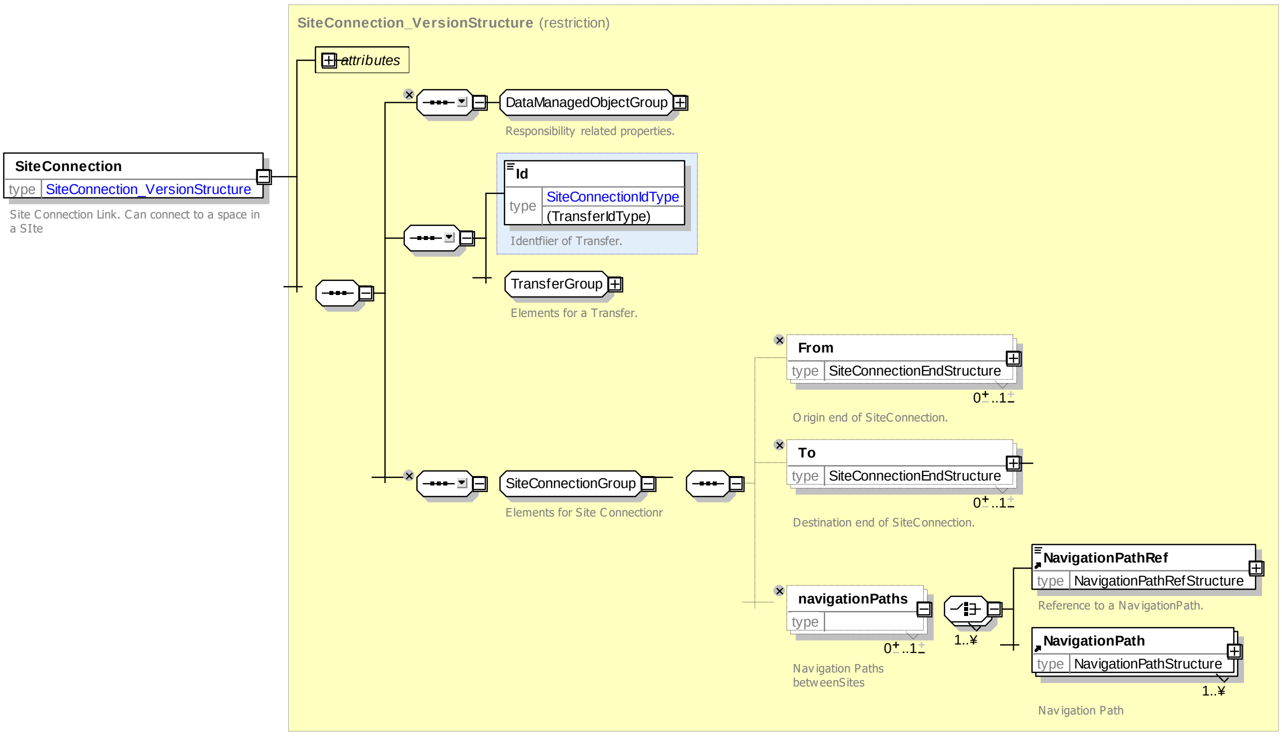


Figure 705 – *SiteConnection* – *XSD*

8.5.2.7.4 SiteConnectionEnd – Model Subelement

One end of a SITE CONNECTION link. It must reference a STOP PLACE SPACE and may additionally reference an ENTRANCE and an OPERATOR.

Table 604 – *SiteConnectionEnd* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DefaultConnectionEnd*	::>	SITE CONNECTION END inherits from Connection End.
«FK»	***StopPlaceRef***	*StopPlaceRef*	0:1	Reference to destination STOP PLACE of SITE CONNECTION.
«FK»	***StopPlaceSpaceRef***	(*StopPlaceSpaceRef*)	1:1	Reference to destination space of SITE CONNECTION. May be a QUAY or an ACCESS SPACE.
		CHOICE		Assignment to a SPACE and or BOARDING POSITION
a «FK»	a ***AccessSpace-Ref***	(*AccessSpaceRef*)	0:1	Reference to an ACCESS SPACE.
FK»	b ***Boarding-PositionRef***	*BoardingPositionRef*	0:1	Reference to a BOARDING POSITION.
FK»	c ***QuayRef***	*QuayRef+*	0:1	Reference to a QUAY.

	FK»	***StopPlaceEntranceRef***	*StopPlaceEntranceRef*	0:1	Reference to a STOP PLACE ENTRANCE.
			CHOICE		
b	«FK»	***PointOfInterestRef***	*PointOfInterestRef*	0:1	Reference to a POINT OF INTEREST.
	«FK»	***PointOfInterestSpaceRef***	*PointOfInterestSpaceRef*	0:1	Reference to a POINT OF INTEREST SPACE.
	«FK»	***PointOfInterestEntranceRef***	*PointOfInterestEntranceRef*	0:1	Reference to a POINT OF INTEREST ENTRANCE.
			CHOICE		
c	«FK»	***ParkingRef***	*ParkingRef*	0:1	Reference to a PARKING.
	«FK»	***ParkingAreaRef***	*ParkingAreaRef*	0:1	Reference to a PARKING AREA.
	«FK»	***ParkingEntranceRef***	*ParkingEntranceRef*	0:1	Reference to a PARKING ENTRANCE.
«FK»		***OperatorRef***	*OperatorRef*	0:1	Reference to destination point of SITE CONNECTION.
		Label	*MultilingualString*	0:1	Label at end of site connection.

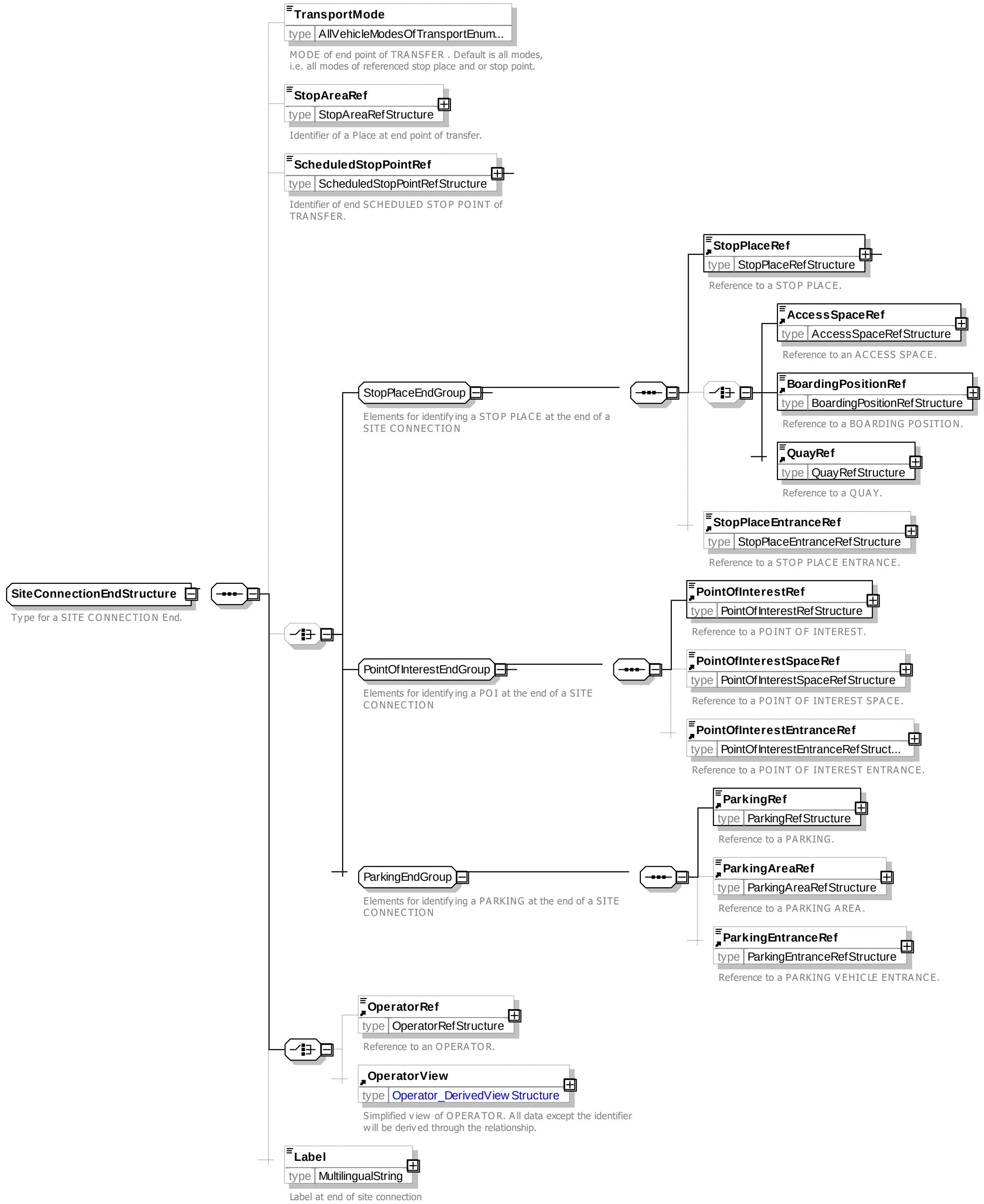


Figure 706 – *SiteConnectionEnd – XSD*

8.5.2.8 XML Examples Transfers, Access and Connections

8.5.2.8.1 Default Transfer Times for a Stop Place – XML Examples

The following XML code fragment of a CONNECTION shows an example of a default TRANSFER DURATION at a SCHEDULED STOP POINT regardless of platform - the ‘from’ and ‘to’ points are the same.

EXAMPLE XML Example of Default Transfer times

```
<Connection id=“tbd:wimcon_01”>
    <FromPointRef>910GWIMBLDN</FromPointRef>
    <ToPointRef>910GWIMBLDN</ToPointRef>
    <Name>Default transfer duration for Wimbledon</Name>
    <TransferDuration>
        <DefaultDuration>PT5M</DefaultDuration>
        <FrequentTravellerDuration>PT2M</FrequentTravellerDuration>
        <OccasionalTravellerDuration>PT5M</OccasionalTravellerDuration>
        <MobilityRestrictedTravellerDuration>PT15M</MobilityRestrictedTravellerDuration>
    </TransferDuration>
</Connection>
```

8.5.2.8.2 Default Transfer Times between two points – XML Examples

The following XML code fragment shows an example of a default TRANSFER DURATION for transferring between a tram SCHEDULED STOP POINT and a tube SCHEDULED STOP POINT.

EXAMPLE XML Example of Transfer Times between two points

```
<Connection id=“tbd:wimcon_01”>
    <FromPointRef>napt:9400ZZCRWIM</FromPointRef>
    <ToPointRef>mapt:9940GZZLUWIM</ToPointRef>
    <Name>Default transfer duration for Wimbledon between tram and Tube</Name>
    <TransferDuration>
        <DefaultDuration>PT8M</DefaultDuration>
        <FrequentTravellerDuration>PT4M</FrequentTravellerDuration>
        <OccasionalTravellerDuration>PT8M</OccasionalTravellerDuration>
        <MobilityRestrictedTravellerDuration>PT19M</MobilityRestrictedTravellerDuration>
    </TransferDuration>
</Connection>
```

8.5.2.8.3 Navigation Path specific Transfer Time – XML Examples

The following XML code fragment shows an example of a NAVIGATION PATH with a summary TRANSFER DURATION set. The times this should be derived from the sum of it’s the times of the individual PATH LINKs.

EXAMPLE XML Example of Transfer Times on a Navigation Path

```
    <NavigationPath created="2010-05-17T09:30:47Z" id=“tbd:9100WIMBLDN_5n6_to_7n8-acc”>
        <AccessibilityAssessment created="2010-05-17T09:30:47Z">
            <MobilityImpairedAccess>true</MobilityImpairedAccess>
            <limitations>
                <AccessibilityLimitation id=“tbd:9100WIMBLDN_5n6_to_7n8-acc_01”>
                    <WheelchairAccess>true</WheelchairAccess>
                    <StepFreeAccess>true</StepFreeAccess>
                    <EscalatorFreeAccess>true</EscalatorFreeAccess>
                    <LiftFreeAccess>false</LiftFreeAccess>
                </AccessibilityLimitation>
            </limitations>
        </AccessibilityAssessment>
```

```xml
<features>
    <AccessSummary>
        <AccessFeatureType>lift</AccessFeatureType>
        <Count>1</Count>
        <Transition>down</Transition>
    </AccessSummary>
</features>
<TransferDuration>
    <DefaultDuration>PT5M</DefaultDuration>
    <MobilityRestrictedTravellerDuration>PT10<</MobilityRestrictedTravellerDuration>
</TransferDuration>
<Name>Platform 5 and 6 to Platform 7 and 8 - Accessible</Name>
<TypeOfNavigation>quayToQuay</TypeOfNavigation>
<pathLinksInSequence>
    <PathLinkInSequence order="1">
    <SitePathLink id="tbd:9100WIMBLDN_lnk_A2b-EI5_A2-J2">
                <Name>From Upper Concourse Lift area Entrance EI1 to Path Junction 2</Name>
                <Covered>indoors</Covered>
                <From>
                <    <PlaceRef>tbd:9100WIMBLDN_A2b</PlaceRef>
                <    <EntranceRef>tbd:9100WIMBLDN_A2b_EI5</EntranceRef>
                </From>
                <To>
                <    <PlaceRef>tbd:9100WIMBLDN_A2_J2</PlaceRef>
                </To>
                <Distance>5.00</Distance>
                <NumberOfSteps>0</NumberOfSteps>
                <AllowedUse>twoWay</AllowedUse>
                <FromToUpDown>level</FromToUpDown>
                <AccessFeatureType>confinedSpace</AccessFeatureType>
                <TransferDuration>
                    <DefaultDuration>PT30S</DefaultDuration>
                </TransferDuration>
                <LevelRef>tbd:9100WIMBLDN_Lvl_G0</LevelRef>
    </SitePathLink>
```

8.5.3 Journey Pattern

NOTE The following explanations use excerpts from Transmodel.

The JOURNEY PATTERN model is concerned with the spatial description of services, i.e. spatial aspects of the work of the vehicles. The concerns of this model are different from those of describing the ROUTEs and LINEs. The latter are describing schematic paths of vehicles through the road network whereas JOURNEY PATTERNs describe how the work of vehicles is performed. These work patterns describes the sequence of points where vehicles stop and specific points attributed with timing information.

This means that a JOURNEY PATTERN consists of:

— An ordered sequence of SCHEDULED STOP POINTs to be served;

— An ordered sequence of TIMING POINTs at which timing information is scheduled. These TIMING POINTs may be SCHEDULED STOP POINTs or other POINTs.

The JOURNEY PATTERN model is directly extracted from EN 12896 (Transmodel V5.1).

8.5.3.1 JOURNEY PATTERN – Conceptual MODEL

The JOURNEY PATTERN is defined through a sequence of POINTs, which play specific roles in that JOURNEY PATTERN. The modelling specifies the roles that a POINT plays relatively to a particular JOURNEY PATTERN.

For instance, a POINT in the network that is a SCHEDULED STOP POINT and also a TIMING POINT may be used by a JOURNEY PATTERN only as a SCHEDULED STOP POINT or only as a TIMING POINT (or both, as a TIMING POINT and a SCHEDULED STOP POINT). This POINT IN JOURNEY PATTERN expresses the fact that the role of a POINT defining a JOURNEY PATTERN is specific to that JOURNEY PATTERN.

The working pattern described by a JOURNEY PATTERN must be related to a ROUTE it covers. Several JOURNEY PATTERNs may use the same ROUTE (e.g. express service or service stopping at all stops).

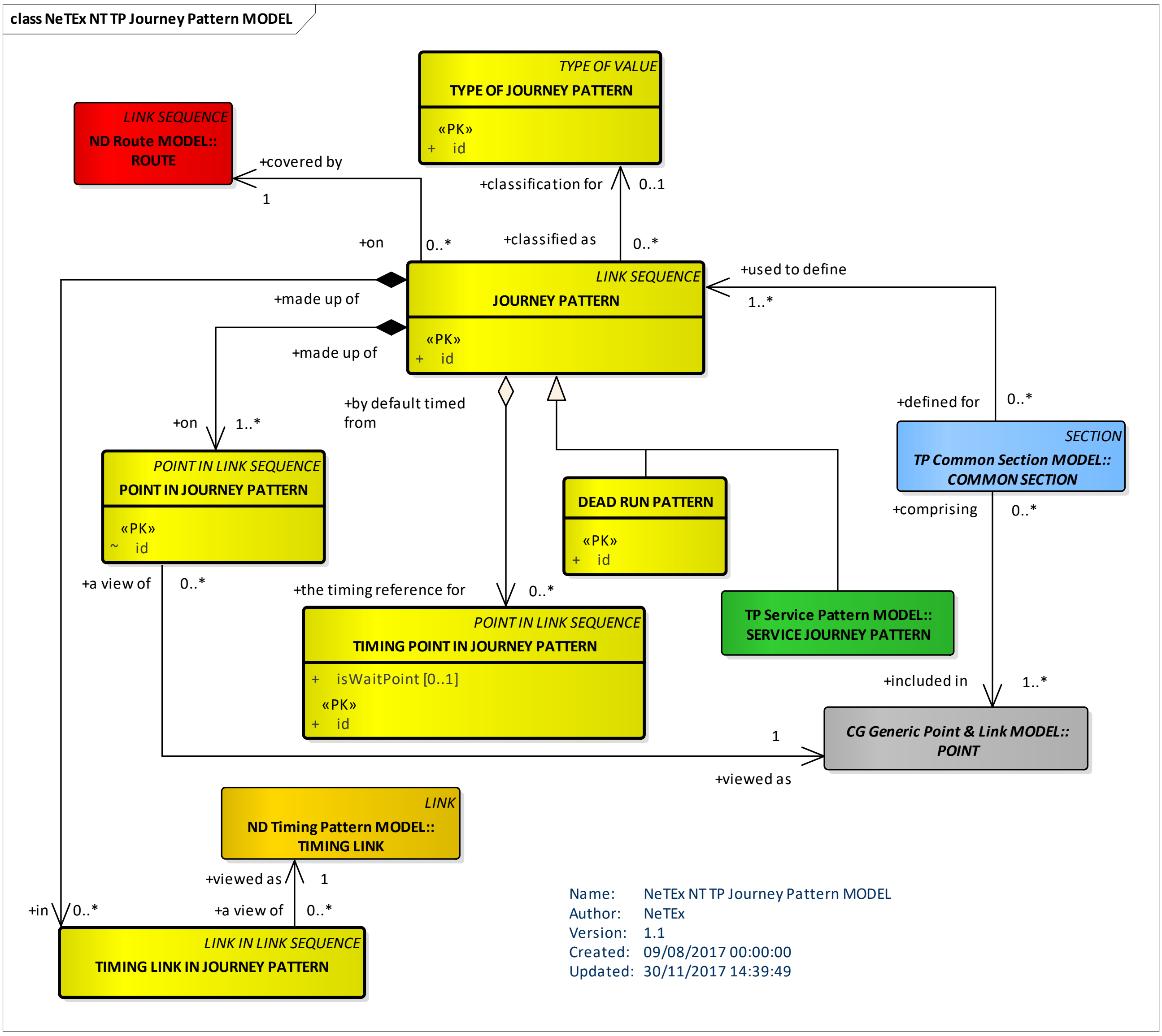


Figure 707 – JOURNEY PATTERN – Conceptual MODEL (UML)

The above model is a concrete instance of the abstract “Generic Point and Link” conceptual model: its main classes are specialisations of POINT, LINK and LINK SEQUENCE. However, such an inheritance from the generic classes is not shown each time in order not to complicate the presentation.

A SERVICE FRAME (see earlier) is used to describe the relevant service – related set of data to which the same validity period has been assigned. It is composed of POINT, LINK, LINK SEQUENCE – related classes. The diagram above shows, as an example, the fact that ROUTE is part of that set. All classes considered as being part of SERVICE FRAME are shown in the Service Frame MODEL.

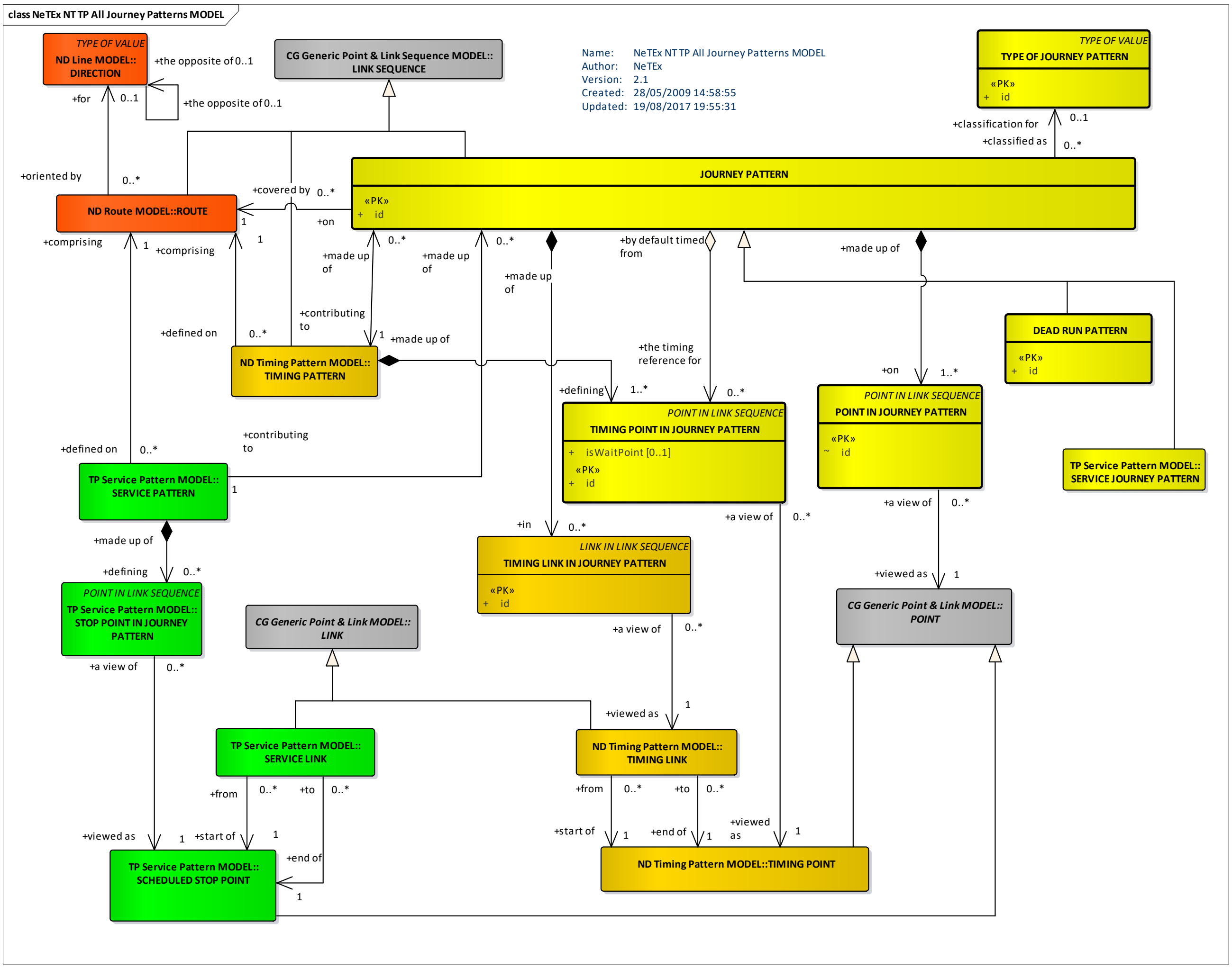


Figure 708 – Points in all Journey Pattern (UML)

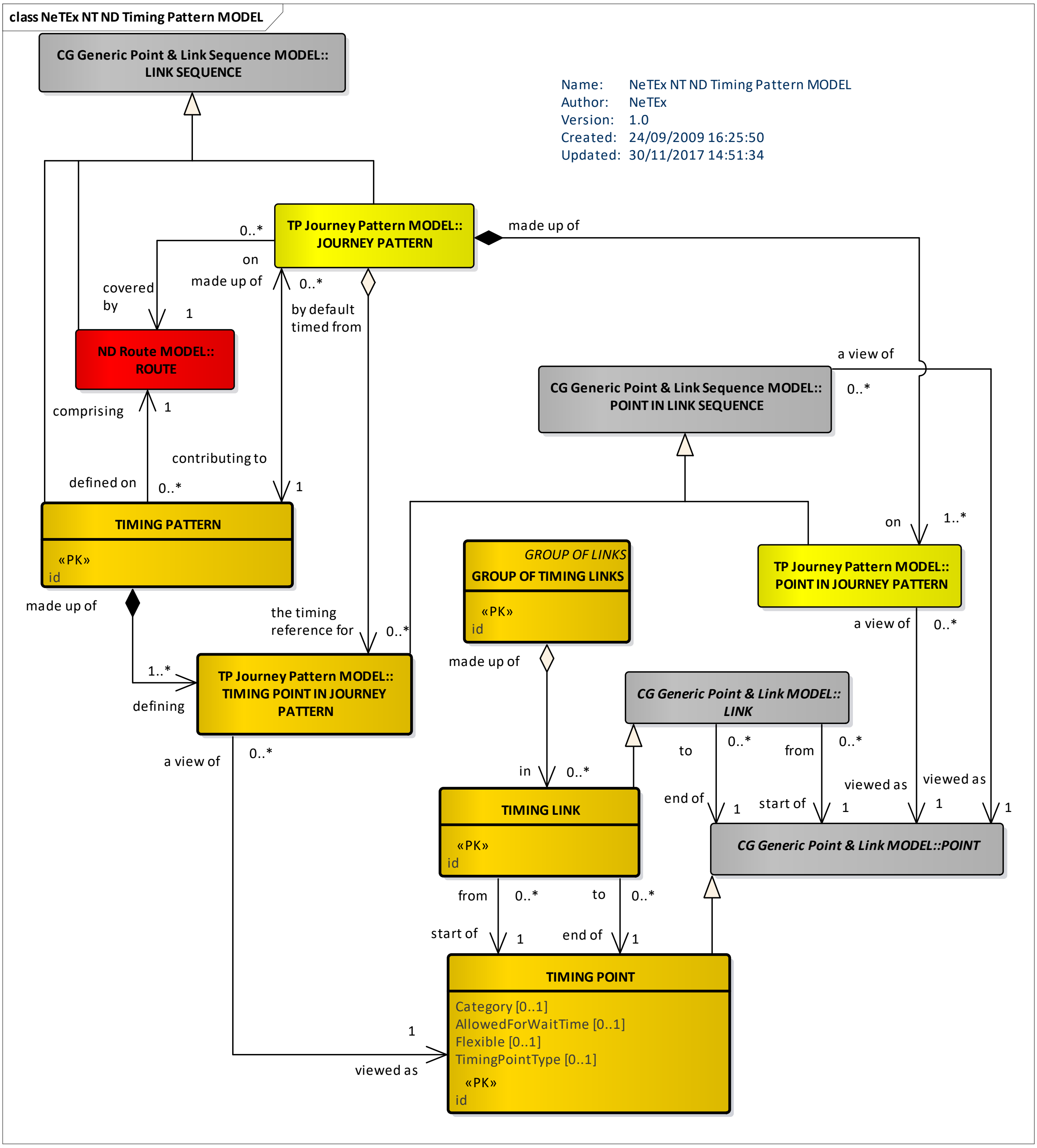


Figure 709 – TIMING PATTERN – Conceptual MODEL (UML)

8.5.3.2 Journey Pattern – Physical Model

The following figure shows detailed attributes of the JOURNEY PATTERN model.

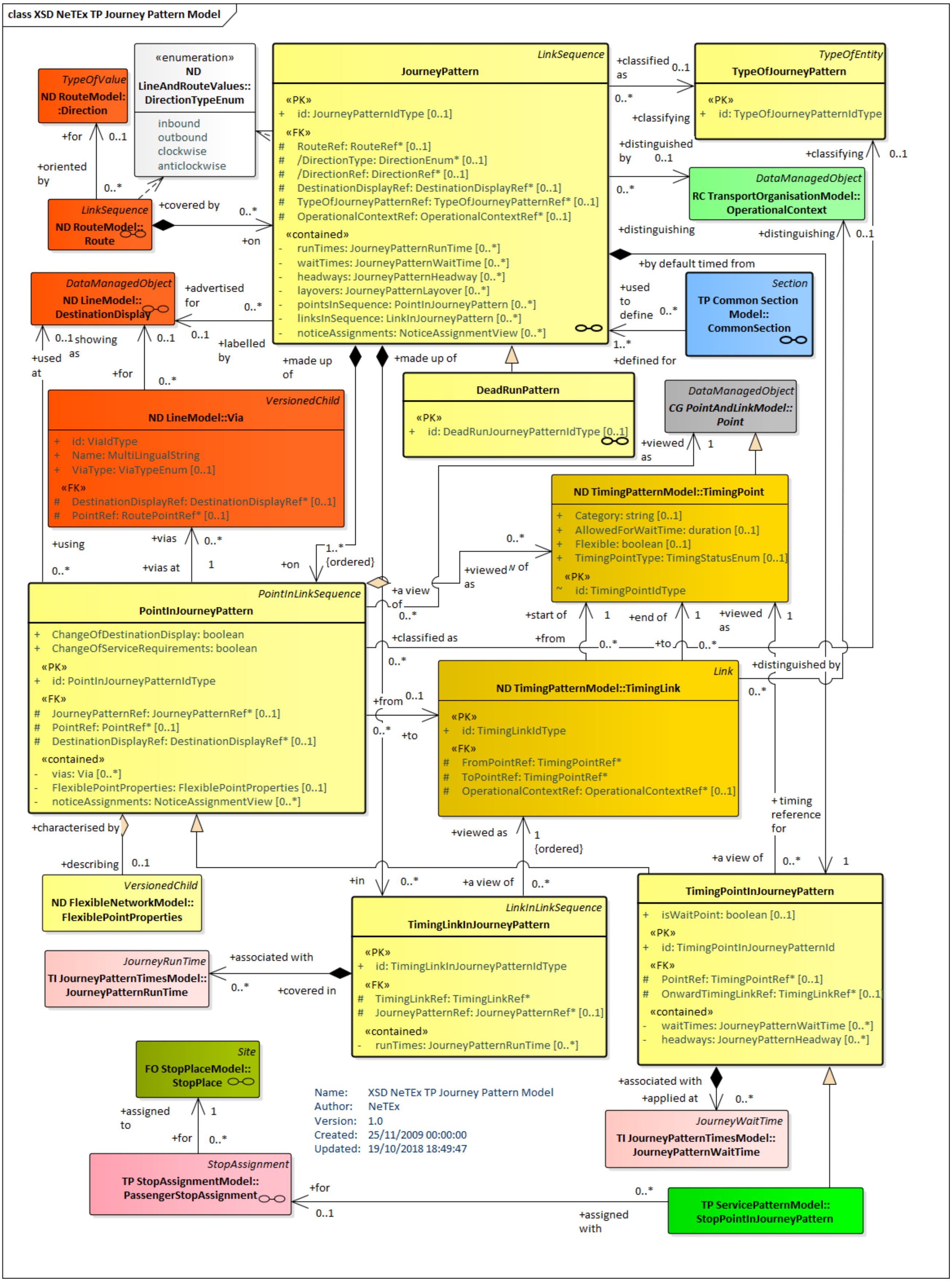


Figure 710 – Journey Pattern – Physical Model (UML)

The physical model simplifies the conceptual model for JOURNEY PATTERN in order to take into account specific XML implementation constraint (in particular to avoid multiple inheritance.). The defining characteristic of the Journey Pattern in the Physical Model is that it is considered as being an ordered sequence of POINTs that are either TIMING POINTs or SCHEDULED STOP POINTs. SCHEDULED STOP POINT is considered a specialisation of TIMING POINT to simplify the implementation. Whenever a SCHEDULED STOP POINT is also a TIMING POINT, the attribute *TimingStatus* indicates it. Not all the SCHEDULED STOP POINTs need to be TIMING POINTs.

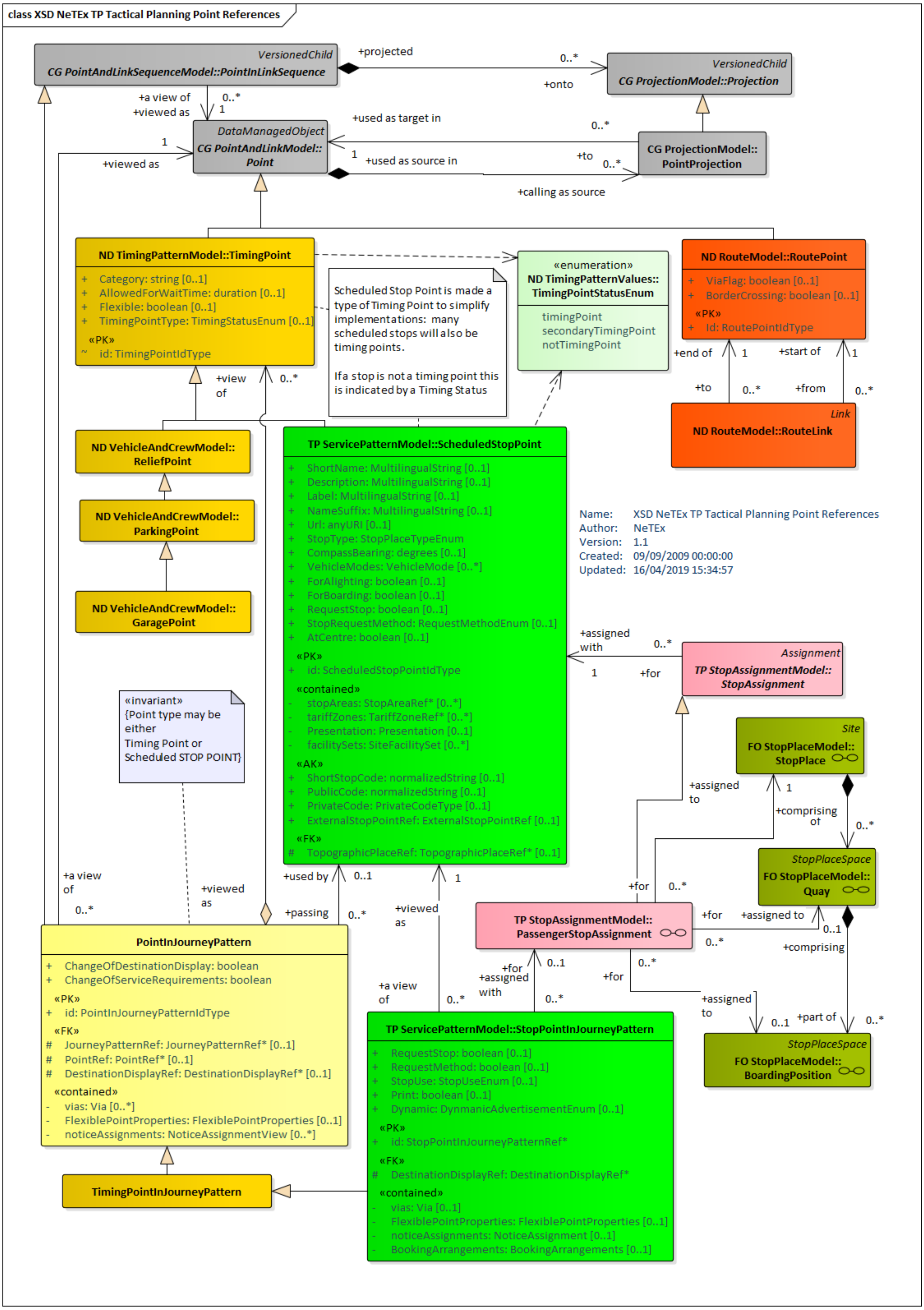


Figure 711 – Tactical planning point references (UML)

8.5.3.3 Journey Pattern – Attributes and XSD

8.5.3.3.1 JourneyPattern – Model Element

A*n* ordered list of STOP POINTs and TIMING POINTs on a single ROUTE, describing the pattern of working for public transport vehicles. As a reminder, a POINT IN JOURNEY PATTERN can be a STOP POINT a TIMING POINT, both or none (only a POINT IN JOURNEY PATTERN). A JOURNEY PATTERN may pass through the same POINT more than once. The first point of a JOURNEY PATTERN is the origin. The last point is the destination.

Table 605 – *JourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*LinkSequence*	::>	JOURNEY PATTERN inherits from LINK SEQUENCE.
«PK»	***id***	*JourneyPatternIdType*	0:1	Identifier of JOURNEY PATTERN.
XGRP	***JourneyPattern-Properties***Group	***xmlGroup***	0:1	Elements describing properties of a JOURNEY PATTERN.
XGRP	***JourneyPattern-TimesGroup***	***xmlGroup***	0:1	Elements describing timings for a JOURNEY PATTERN.
XGRP	***JourneyPattern-SequencesGroup***	***xmlGroup***	0:1	Elements describing component LINKs and POINTs of a JOURNEY PATTERN.

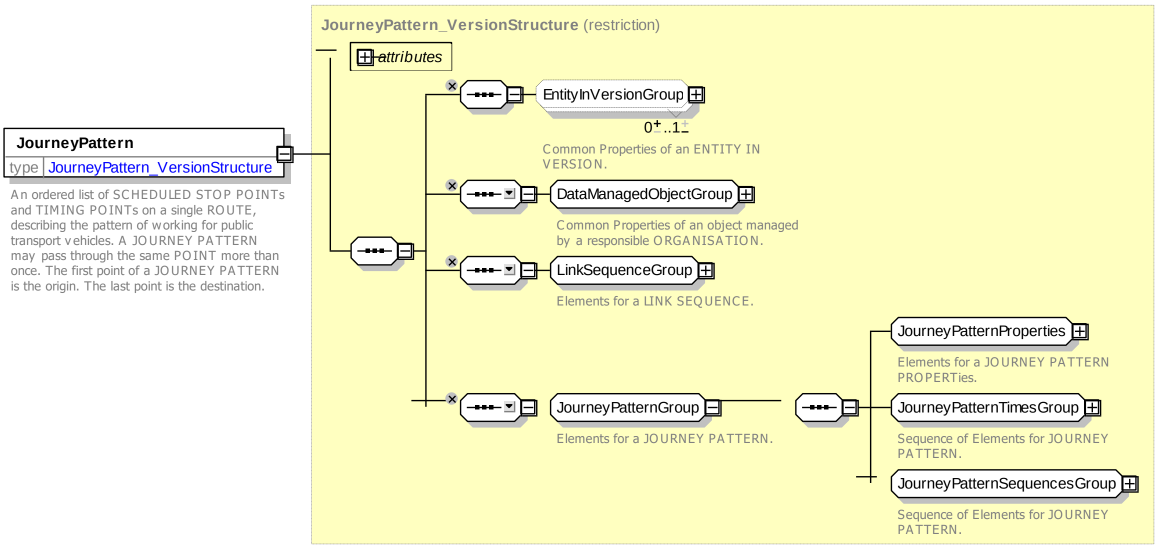


Figure 712 – *JourneyPattern – XSD*

8.5.3.3.1.1 JourneyPatternProperties – XML Group

Elements describing properties of a JOURNEY PATTERN.

Table 606 – *JourneyPatternPropertiesGroup* – Group

Classification	Name	Type	Cardinality	Description
«FK»	***RouteRef***	*RouteRef*	0:1	Reference to ROUTE that JOURNEY PATTERN follows.
«enum»	***DirectionType***	*DirectionTypeEnum*	0:1	Direction type. See DIRECTION element for allowed values.
«FK»	***DirectionRef***	*DirectionRef*	0:1	DIRECTION of JOURNEY PATTERN (often used to distinguish groups of Journey Patterns using the same branches (i.e. Routes) of a Line)
«FK»	***Destination-DisplayRef***	*DestinationDisplayRef*	0:1	DESTINATION DISPLAY associated with JOURNEY PATTERN.
«FK»	***TypeOfJourney-PatternRef***	*TypeOfJourneyPatternRef*	0:1	Type of JOURNEY PATTERN.
«FK»	***Operational-ContextRef***	*OperationalContextRef*	0:1	OPERATIONAL CONTEXT associated with JOURNEY PATTERN
«FK»	***TimingPatternRef***	*TimingPatternRef*	0:1	Reference to a TIMING PATTERN.
«cntd»	***notice-Assignments***	*NoticeAssignment*	0:*	NOTICE ASSIGNMENTS for JOURNEY PATTERN.

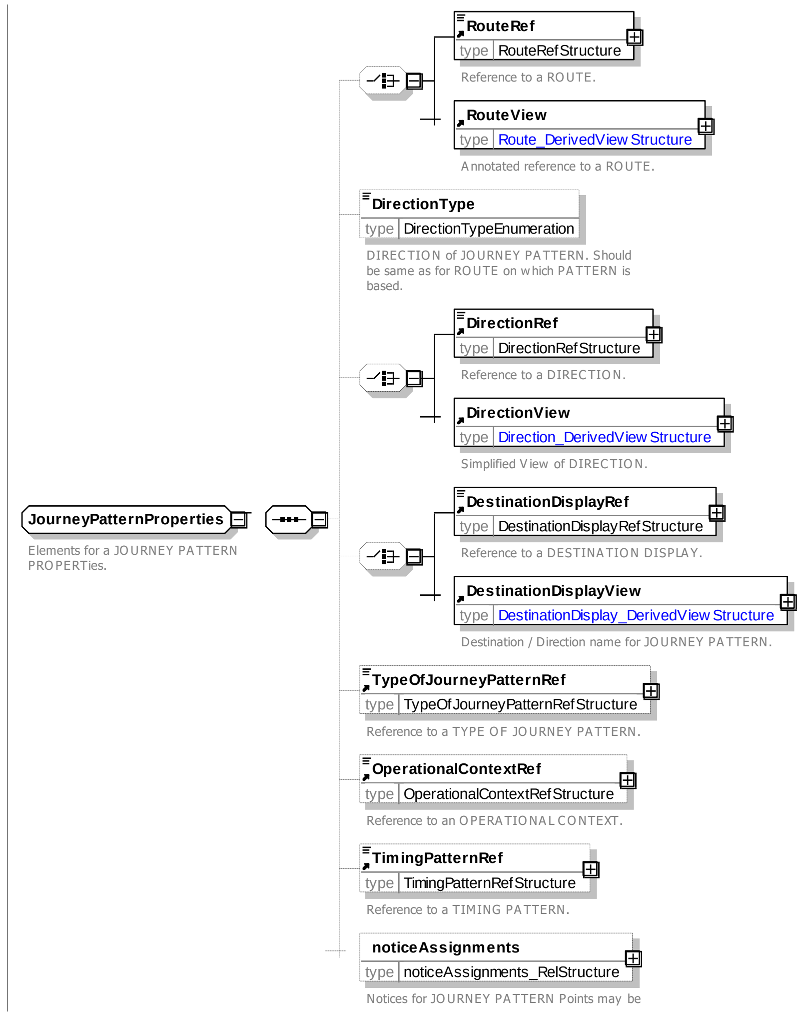


Figure 713 – *JourneyPatternPropertiesGroup* – *XSD*

8.5.3.3.1.2 JourneyPatternTimesGroup – XML Group

Elements describing timings for a JOURNEY PATTERN.

Table 607 – *JourneyPatternTimesGroup* – Group

Classification	Name	Type	Cardinality	Description
«cntd»	***runTimes***	*JourneyPatternRunTime*	0:*	RUN TIMEs in JOURNEY PATTERN.
«cntd»	***waitTimes***	*JourneyPatternWaitTime*	0:*	WAIT TIMEs in JOURNEY PATTERN.
«cntd»	***headways***	*JourneyPatternHeadway*	0:*	HEADWAYs in JOURNEY PATTERN.
«cntd»	***layovers***	*JourneyPatternLayover*	0:*	LAYOVERs in JOURNEY PATTERN.

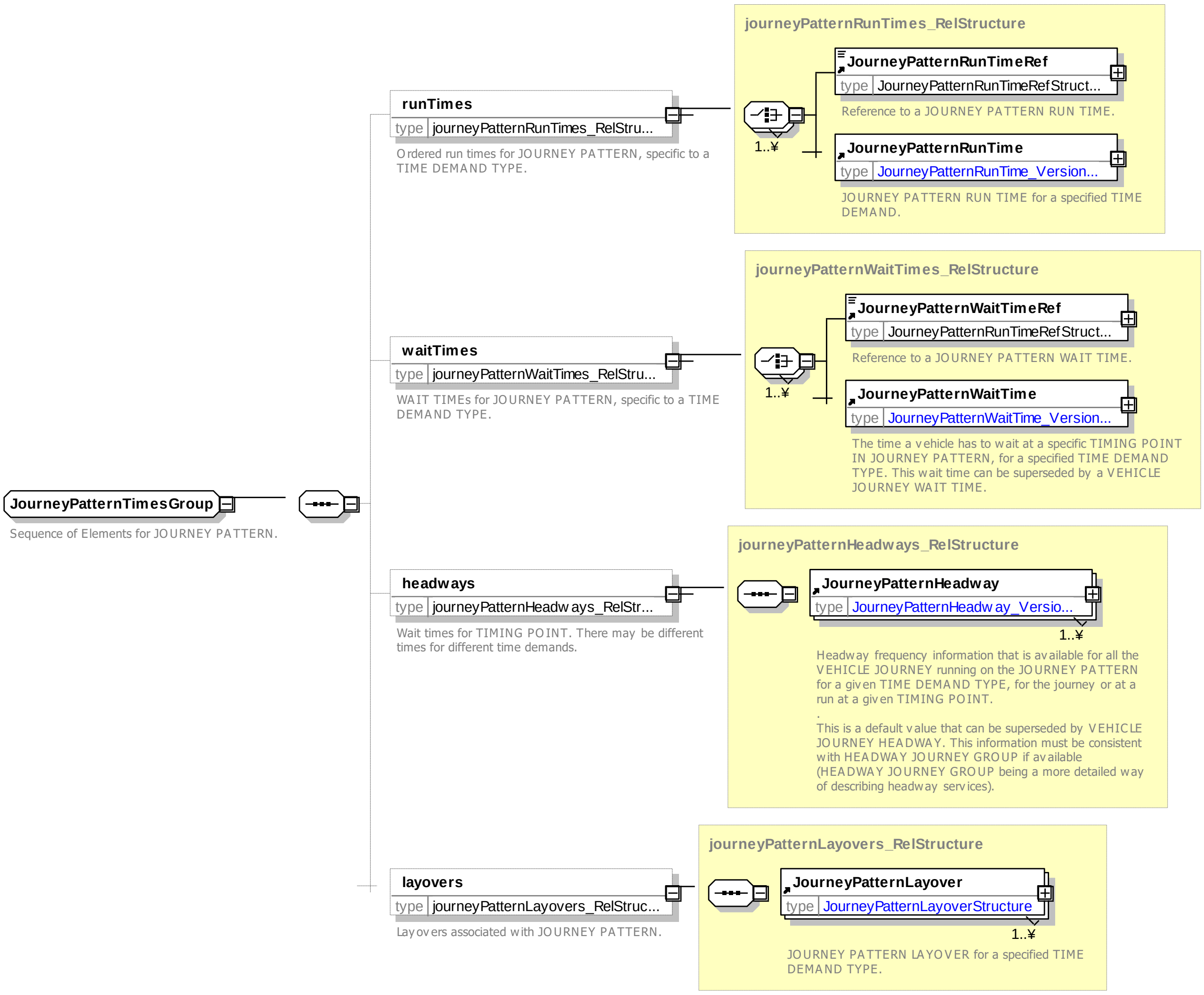


Figure 714 – *JourneyPatternTimesGroup – XSD*

8.5.3.3.1.3 JourneyPatternSequencesGroup – XML Group

Elements describing POINT and LINK components of a JOURNEY PATTERN.

Table 608 – *JourneyPatternSequencesGroup* – Group

Classification	Name	Type	Cardinality	Description
«cntd»	***pointsInSequence***	*PointInJourneyPattern*	0:*	TIMING POINTs in JOURNEY PATTERN.
«cntd»	***linksInSequence***	*LinkInJourneyPattern*	*0:**	TIMING LINKs in JOURNEY PATTERN.

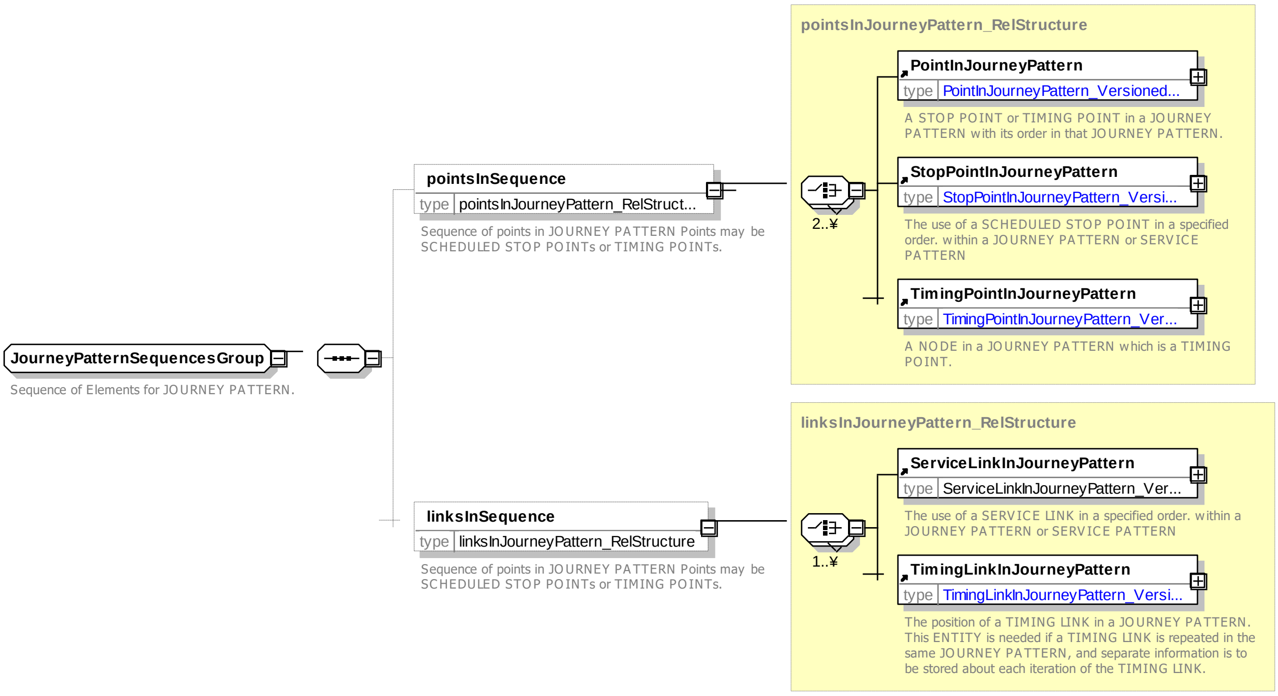


Figure 715 – *JourneyPatternSequencesGroup – XSD*

8.5.3.3.2 DeadRunJourneyPattern – Model Element

A JOURNEY PATTERN to be used for DEAD RUNs.

Table 609 – *DeadRunPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*JourneyPattern*	::>	DEAD RUN JOURNEY PATTERN inherits from JOURNEY PATTERN
«PK»	***id***	*DeadRunJourneyPatternIdType*	0:1	Identifier of DEAD RUN JOURNEY PATTERN.

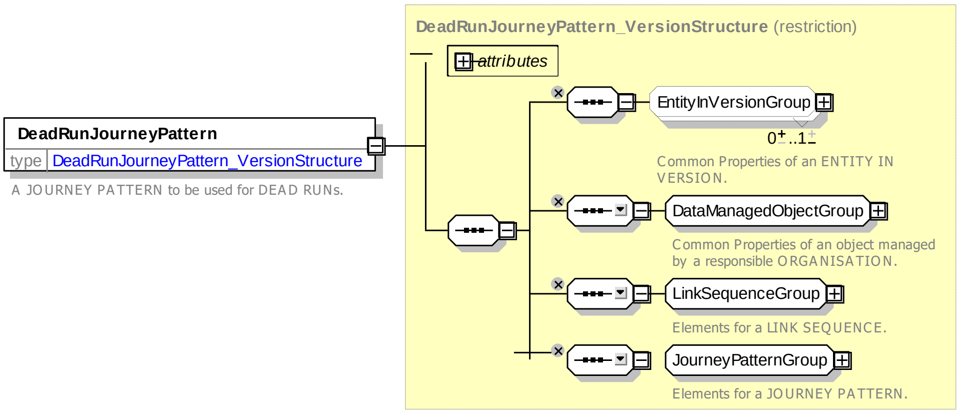


Figure 716 – *DeadRunPattern – XSD*

8.5.3.3.3 *PointInJourneyPattern – Model Element*

A STOP POINT or TIMING POINT in a JOURNEY PATTERN with its order in that JOURNEY PATTERN.

Table 610 – *PointInJourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PointInLinkSequence*	*::>*	POINT IN JOURNEY PATTERN inherits from POINT IN LINK SEQUENCE
«PK»	***id***	*PointInJourneyPattern-IdType*	1:1	Identifier of POINT IN JOURNEY PATTERN.
«FK»	***JourneyPatternRef***	*JourneyPatternRef*	0:1	Parent JOURNEY PATTERN (this is in fact the inherited ***LinkSequenceRef***, from POINT IN LINK SEQUENCE, using the XSD substitution group ***JourneyPatternRef*** belongs to)
«FK»	***PointRef***	*(PointRef)*	0:1	POINT associated with POINT IN JOURNEY PATTERN.
«FK»	***DestinationDisplay-Ref***	*DestinationDisplayRef*	0:1	DESTINATION DISPLAY associated with POINT IN JOURNEY PATTERN.
«cntd»	***vias***	*Via*	0:*	Via points associated with POINT IN PATTERN.
«cntd»	***FlexiblePoint-Properties***	*FlexiblePointProperties*	0:1	Flexible properties of a LINK.
	ChangeOf-DestinationDisplay	*xsd:boolean*	0:1	Whether DESTINATION DISPLAY changes at this point.
	ChangeOfService-Requirements	*xsd:boolean*	0:1	Whether SERVICE REQUIREMENTs change at this point.
«cntd»	***noticeAssignments***	*NoticeAssignmentView*	0:*	NOTICE ASSIGNMENTs, footnotes etc associated with the POINT in PATTERN.

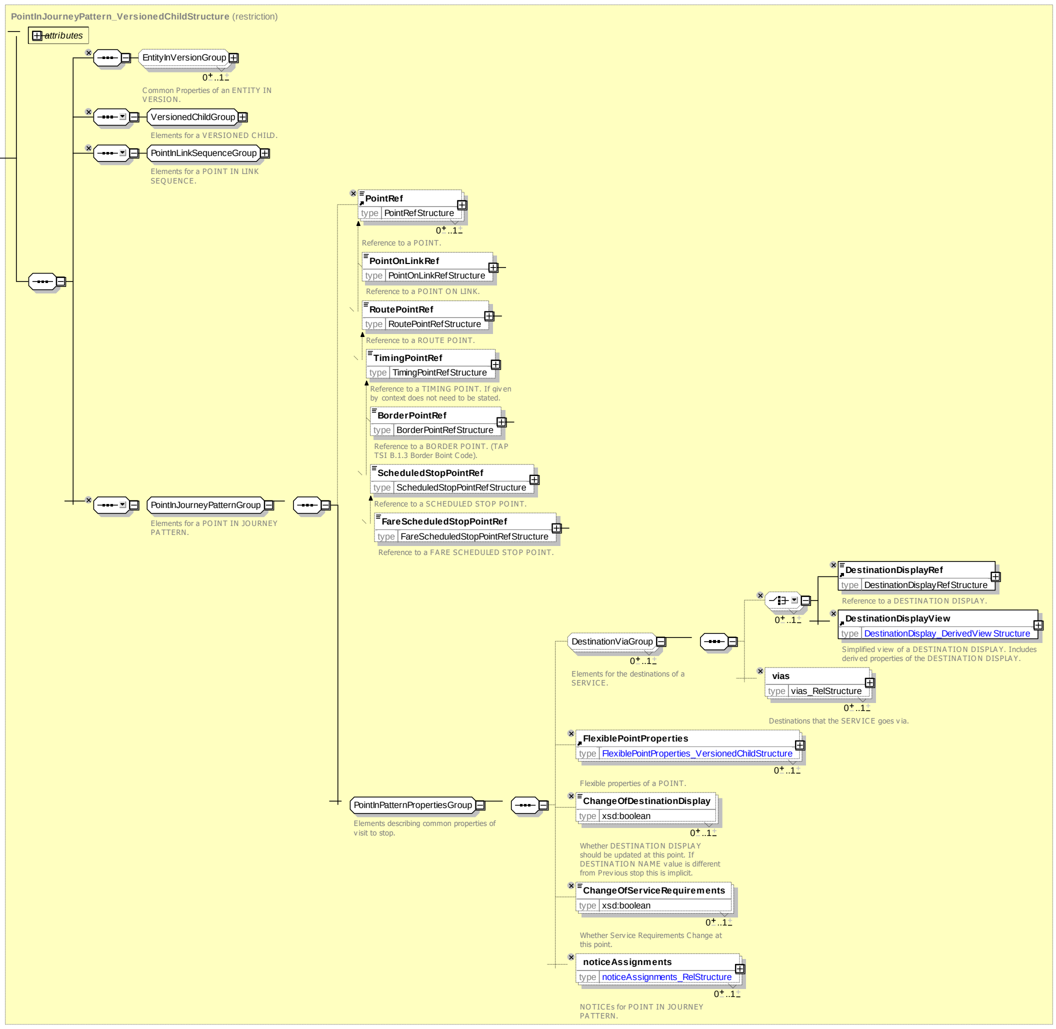


Figure 717 – *PointInJourneyPattern – XSD*

8.5.3.3.4 TimingPointInJourneyPattern – Model Element

A POINT in a JOURNEY PATTERN which is a TIMING POINT.

Table 611 – *TimingPointInJourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*PointInSequence*	*::>*	TIMING POINT IN JOURNEY PATTERN inherits from POINT IN SEQUENCE
«PK»	***id***	*TimingPointInJourney-PatternIdType*	1:1	Identifier of TIMING POINT IN JOURNEY PATTERN.
«FK»	***TimingPointRef***	*TimingPointRef+*	1:1	Reference to the TIMING POINT that is visited.
«FK»	***OnwardTimingLinkRef***	*TimingLinkRef+*	0:1	Next TIMING LINK traversed in pattern after this POINT.
	IsWaitPoint	*xsd:boolean*	0:1	Whether TIMING POINT is a wait point.

			CHOICE		
	a	***WaitTime***	*xsd:duration*	0:1	Waiit time value at POINT IN JOURNEY PATTERN.
«cntd»	*b*	***waitTimes***	*JourneyPatternWaitTime*	0:*	*Wait times at POINT IN JOURNEY PATTERN.*
«cntd»	***headways***		*JourneyPatternHeadway*	0:*	Frequency of services at POINT IN JOURNEY PATTERN.
«cntd»	***noticeAssignments***		*NoticeAssignmentView*	0:*	NOTICE ASSIGNMENTs, footnotes etc associated with the TIMING POINT in PATTERN.

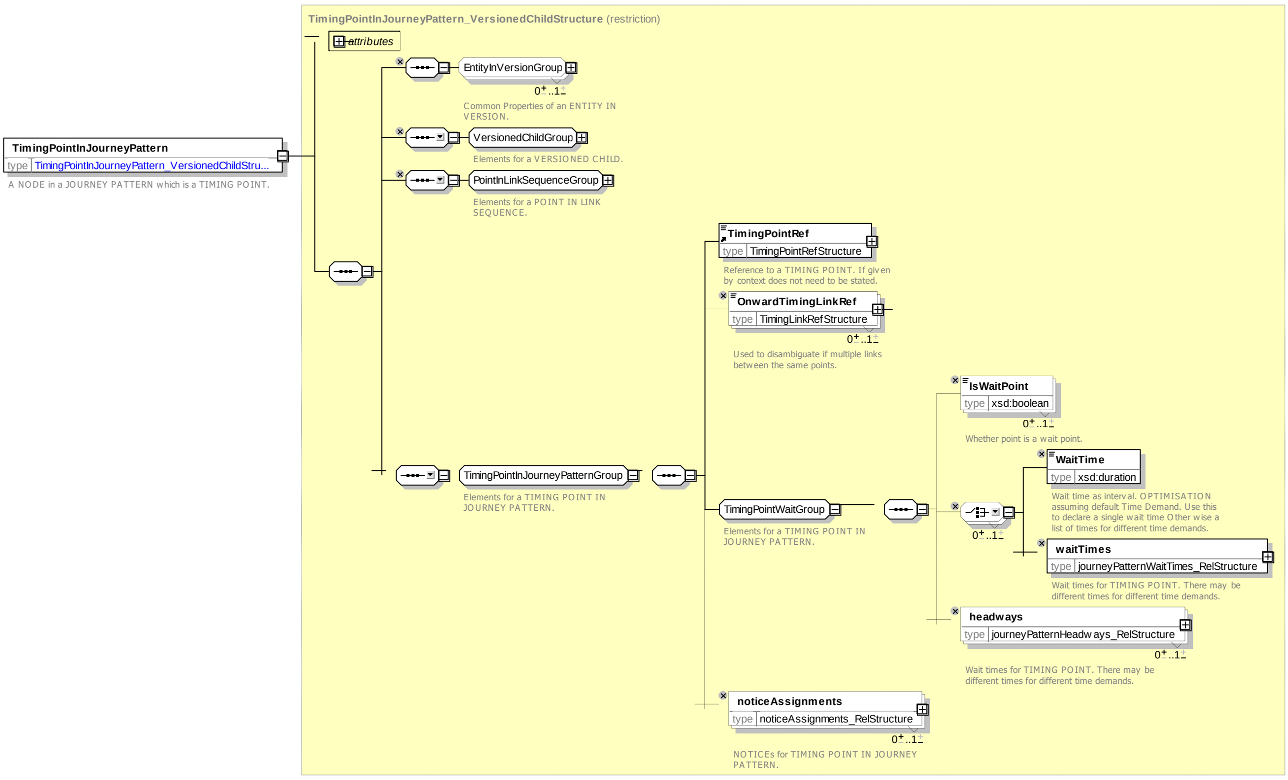


Figure 718 – *TimingPointInJourneyPattern – XSD*

8.5.3.3.5 TimingLinkInJourneyPattern – Model Element

The position of a TIMING LINK in a JOURNEY PATTERN. This entity is needed if a TIMING LINK is repeated in the same JOURNEY PATTERN, and separate information is to be stored about each iteration of the TIMING LINK.

Table 612 – *TimingLinkInJourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*LinkInSequence*	*::>*	TIMING LINK IN JOURNEY PATTERN inherits from LINK IN SEQUENCE.
«PK»	***id***	*TimingLinkInJourneyPatternIdType*	1:1	Identifier of TIMING LINK IN JOURNEY PATTERN.

«FK»	***TimingLinkRef***	*TimingLinkRef*	1:1	TIMING LINK associated with TIMING LINK IN JOURNEY PATTERN.
«cntd»	***runTimes***	*JourneyPatternRunTime*	0:*	Run times associated with TIMING LINK IN JOURNEY PATTERN.

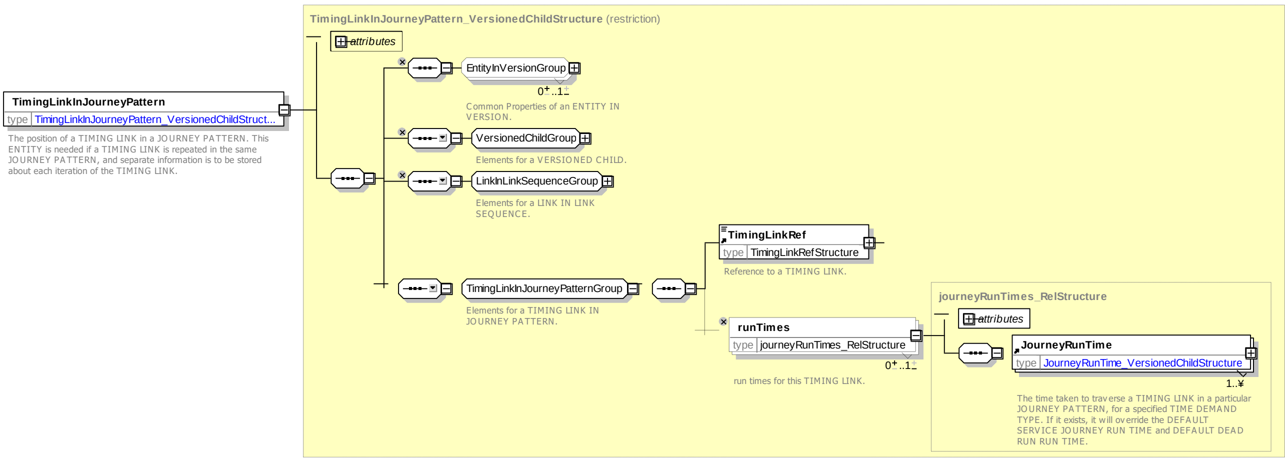


Figure 719 – *TimingLinkInJourneyPattern – XSD*

8.5.3.3.6 TypeOfJourneyPattern – Model Element

A classification of JOURNEY PATTERNs used to distinguish other categories of JOURNEY PATTERN than SERVICE JOURNEY PATTERN and DEAD RUN PATTERN.

Table 613 – *TypeOfJourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF JOURNEY PATTERN inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfJourneyPatternIdType*	1:1	Identifier of TYPE OF JOURNEY PATTERN.

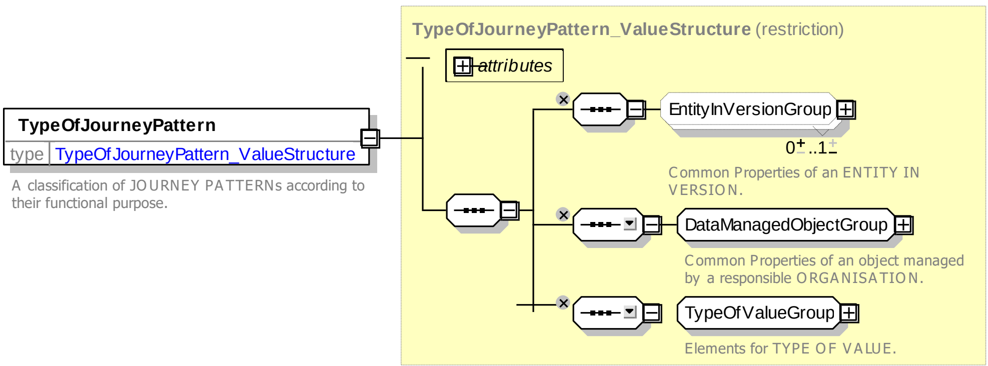


Figure 720 – *TypeOfJourneyPattern – XSD*

8.5.4 Service Pattern

NOTE The following explanations use excerpts from Transmodel.

A Service Pattern is a view of a JOURNEY PATTERN, i.e. the vehicle service from the point of view of the passenger. Passengers are not interested in points dedicated to the scheduling process or vehicle follow up, but in SCHEDULED STOP POINTs, i.e. locations where they will be able to board or leave a vehicle.

Warning: it should be noted that in many cases (systems or companies) the sequence of such points is often called a line. However, the objective of Transmodel is the separation of concerns: LINEs are groupings of ROUTEs, i.e. schematic views of physical paths through the network, determined through ROUTE POINTs, whereas SERVICE PATTERNs are sequences of points of another type. Thus a Transmodel LINE is conceptually *different* from a grouping of SERVICE PATTERN even if a link exists (a LINE is a group of ROUTEs and each ROUTE is linked to one or more SERVICE PATTERNs).

Of course, from the point of view of the precise physical path (along the road network) both types of points (ROUTE POINTs and SCHEDULED STOP POINTs) are on the INFRASTRUCTURE LINKs taken by the vehicle but the ROUTE and SERVICE PATTERN are not the same LINK SEQUENCEs as shown in the figure below.

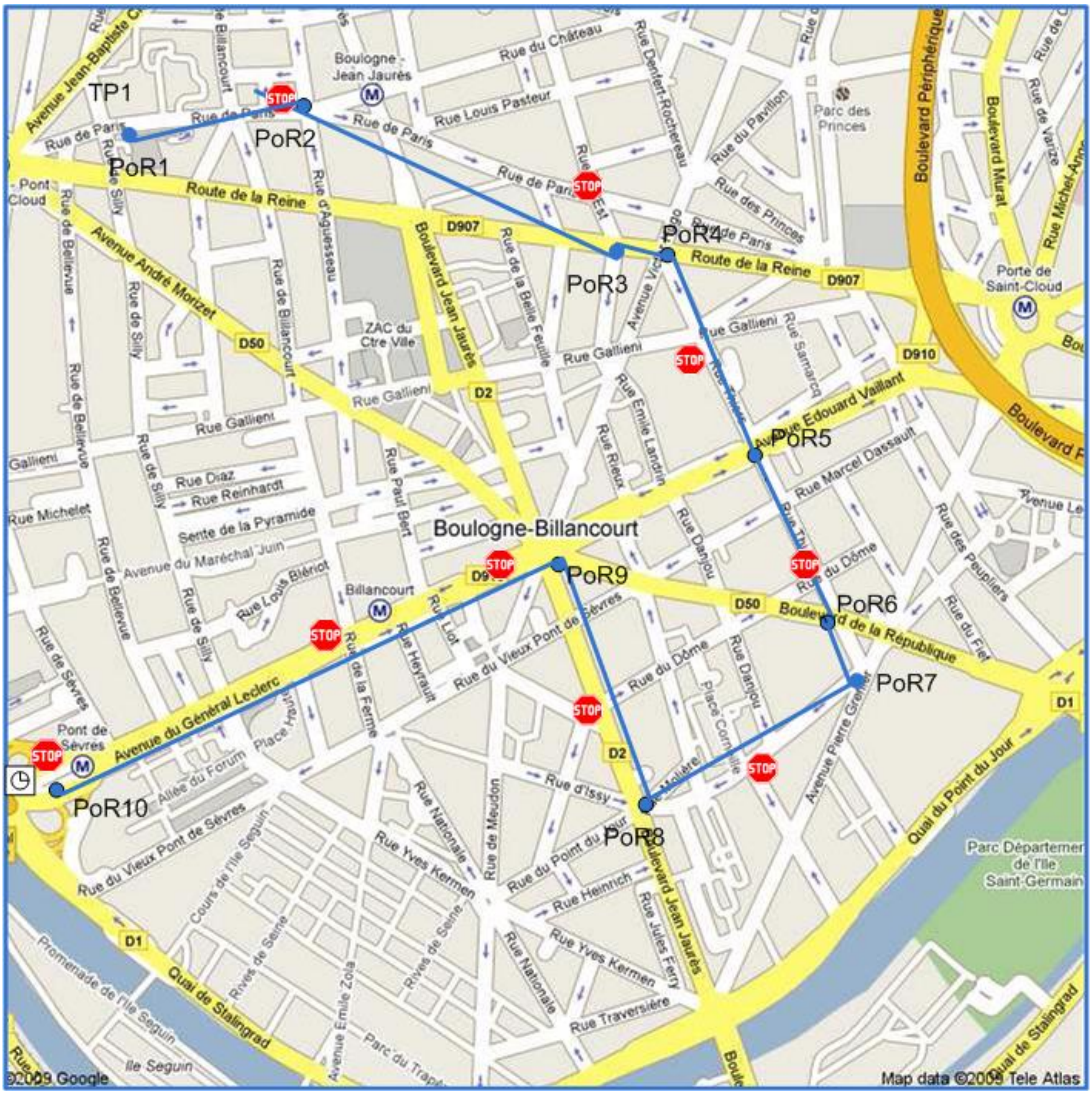


Figure 721 – ROUTE and SERVICE PATTERN Example

The example above shows the different LINK SEQUENCEs: a ROUTE, defined by a sequence of POINTs ON ROUTE (each of them also being a ROUTE POINT: a single ROUTE POINT can be used by several POINTs ON ROUTE in case of circular ROUTE for example) and a SERVICE PATTERN is defined by an ordered sequence of SCHEDULED STOP POINTs. Additionally is shown a TIMING PATTERN, defined by a sequence of TIMING POINTs which are, except the first and the last point in this sequence (in this example), different from SCHEDULED STOP POINTs.

8.5.4.1 SERVICE PATTERN – Conceptual MODEL

A SERVICE PATTERN is made up of an ordered sequence of STOP POINTs IN JOURNEY PATTERN, with a SERVICE LINK between each pair of consecutive SCHEDULED STOP POINTs. As the same SCHEDULED STOP POINT may occur more than once in the same SERVICE PATTERN, a STOP POINT IN JOURNEY PATTERN is identified by that SERVICE PATTERN together with an 'order' attribute. A relationship between

STOP POINT IN JOURNEY PATTERN and SCHEDULED STOP POINT determines the SCHEDULED STOP POINT that is related to that order.

It has to be noted that a SERVICE PATTERN is a sequence of SCHEDULED STOP POINTs to be served by one or several JOURNEY PATTERNs.

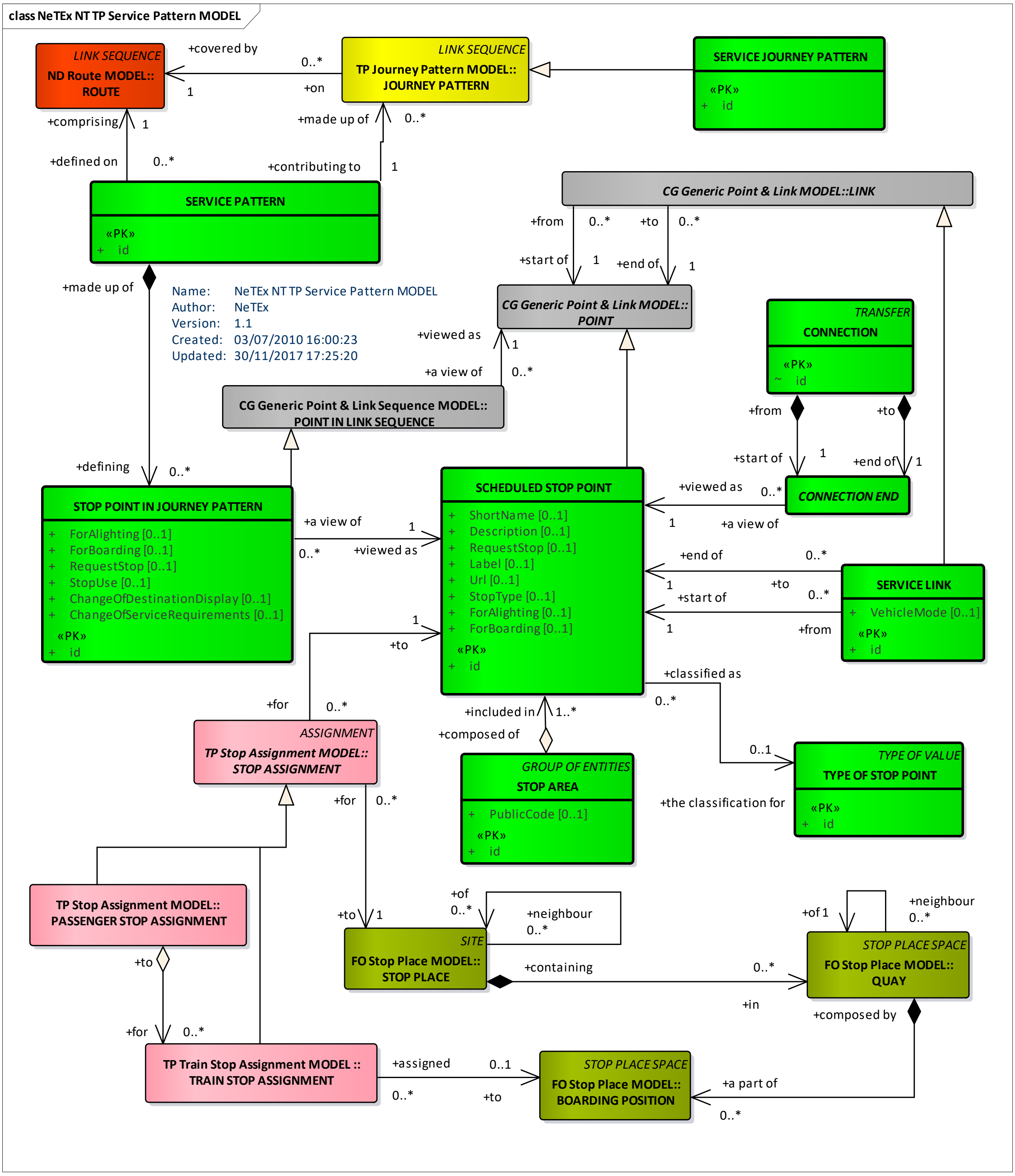


Figure 722 – Service Pattern – Conceptual MODEL (UML)

8.5.4.2 Service Pattern – Examples

The following figure introduces the Eurostar route network. There are two main destinations (Paris and Brussels) and three minor destinations. In NeTEx terms we can consider each of these to be a LINE.

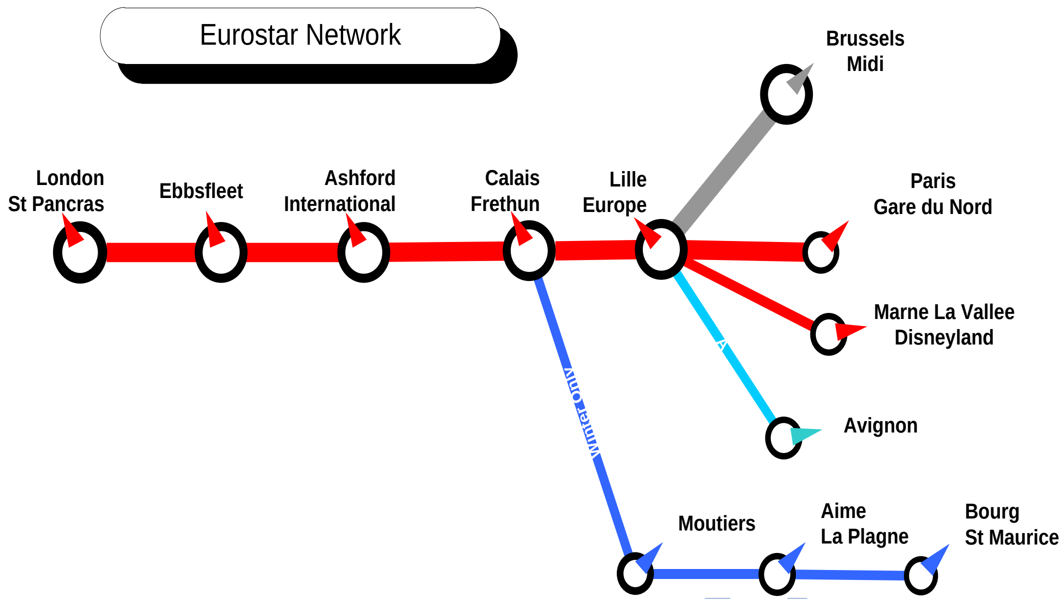


Figure 723 – Eurostar Lines – Example

In fact services to specific destinations only stop at particular stops, i.e. have different SERVICE PATTERNs, as shown in the following figure.

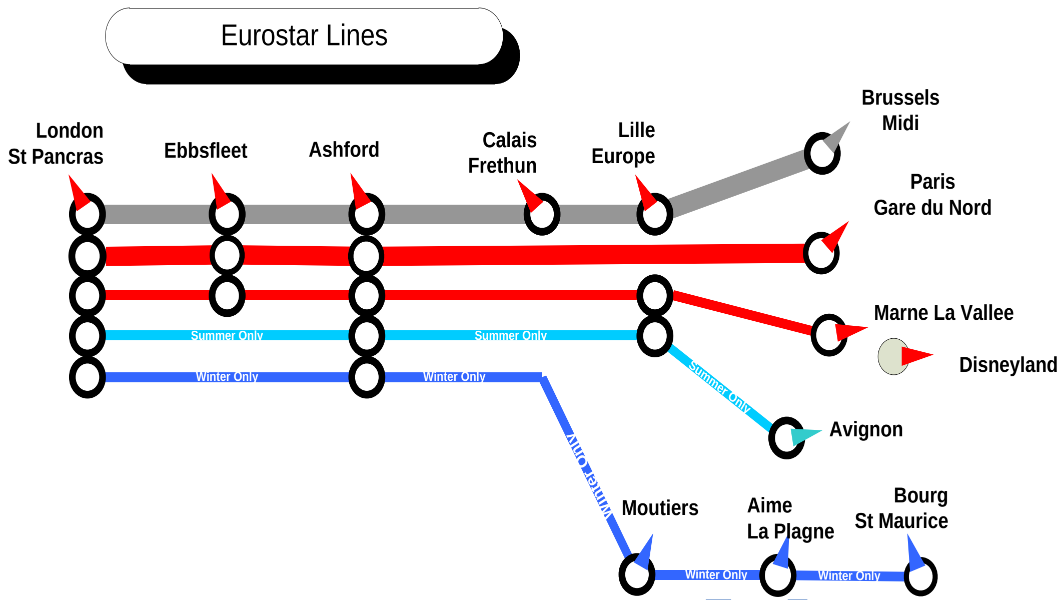


Figure 724 Lines – Example Eurostar stops – Example

Furthermore if we consider the journeys for the two main “Lines”, *London to Paris*, in one direction, as shown in the following timetable, we see that different services on the same line may also have different SERVICE PATTERNs, representing different subsets of stops visited.

London ▸ Paris/Brussels

Monday to Friday

Notes	LONDON	EBBSFLEET	ASHFORD	CALAIS	LILLE	BRUSSELS	PARIS	Train no
1	05:25	05:42	-	-	-	-	08:50	9078
	06:19	06:36	06:58	08:32	09:10	09:42		9108
2	06:12	06:29	06:53	-	-	-	09:54	9002
1	06:53	-	07:25	-	-	-	10:17	9004
	07:22	07:41	-	-	-	-	10:47	9006
	07:34	-	-	-	-	10:27		9112
	08:02	-	-	-	-	-	11:17	9008
	08:27	08:45	-	-	10:58	11:33		9120
1	08:55	09:12	-	-	-	-	12:17	9012
	09:22	-	09:55	-	-	-	12:47	9014
	10:25	10:42	-	-	-	-	13:47	9018
	10:57	11:15	-	13:02	13:35	14:12		9126
3	11:32	-	-	-	-	-	14:47	9022
	12:28	12:45	-	-	-	-	15:50	9024
	12:58	13:15	-	-	15:28	16:05		9132
	14:02	-	-	-	-	-	17:23	9030
	14:34	-	-	-	16:58	17:33		9138
	15:02	-	-	-	-	-	18:17	9034
	16:02	-	-	-	-	-	19:17	9038
	16:22	-	16:55	-	-	-	19:47	9040
	17:04	-	-	-	19:28	20:03		9148
	17:32	-	-	-	-	-	20:47	9044
	18:02	-	-	-	-	-	21:17	9046
3	18:31	-	-	-	-	-	21:47	9048
4	18:34	-	-	20:32	21:02	21:37		9154
3	18:34	-	-	20:33	-	21:33		9154
	19:02	-	-	-	-	-	22:17	9050
	19:34	-	-	-	21:58	22:33		9158
	20:02	-	-	-	-	-	23:17	9054

Figure 725 Lines –Eurostar Example - London to Paris trains

If we analyse the journeys for just one of the “Lines”, London to Paris, we find five distinct SERVICE PATTERNs, most of which are shared between more than one journey

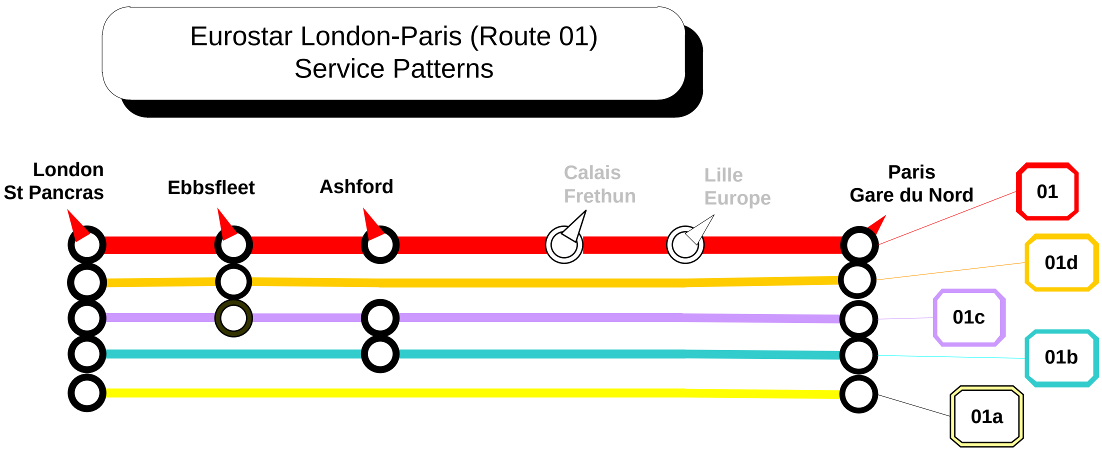


Figure 726– Eurostar Example - Outbound Journeys

The London Paris service may be described using the following set of patterns (Figure 727).

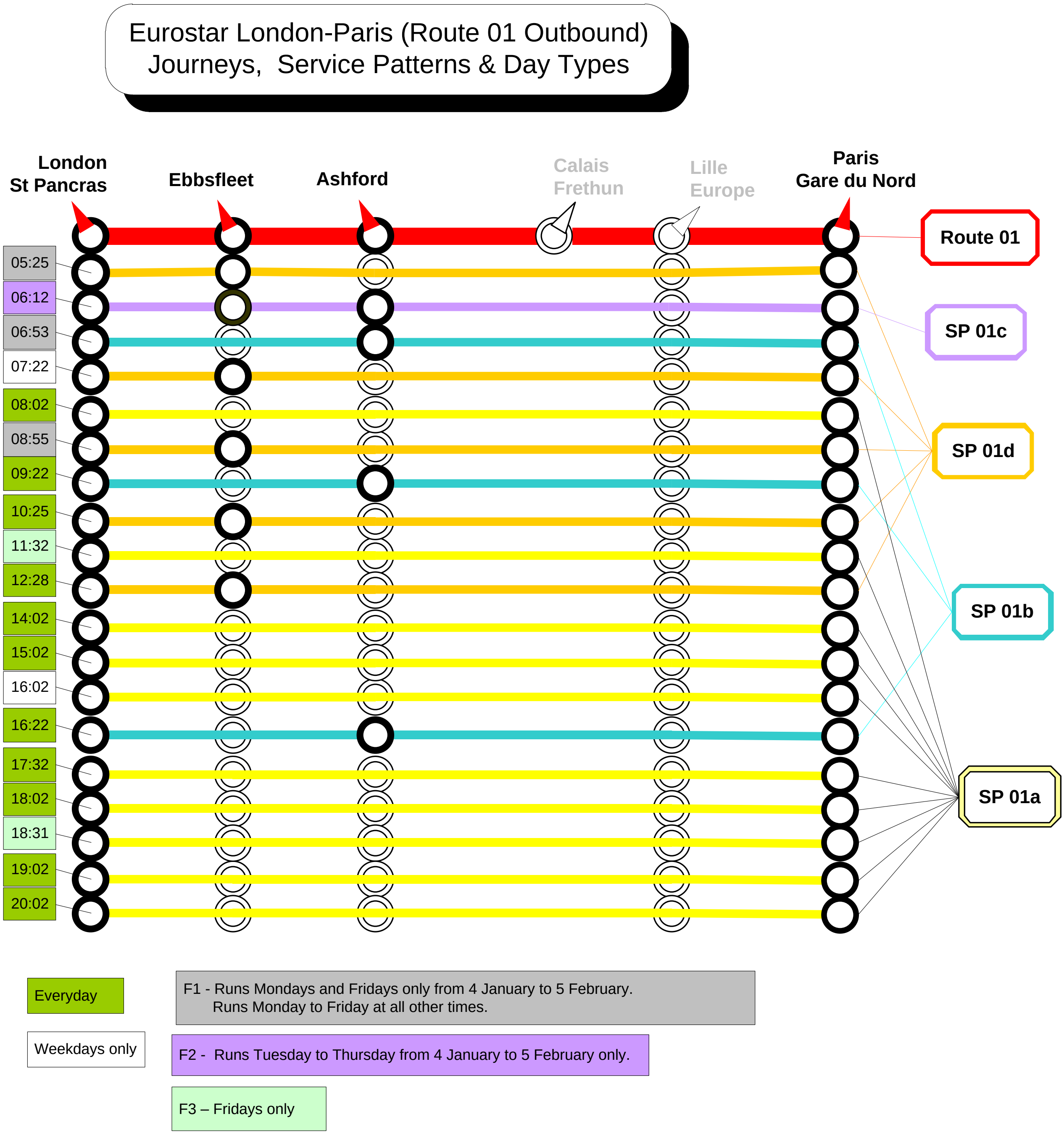
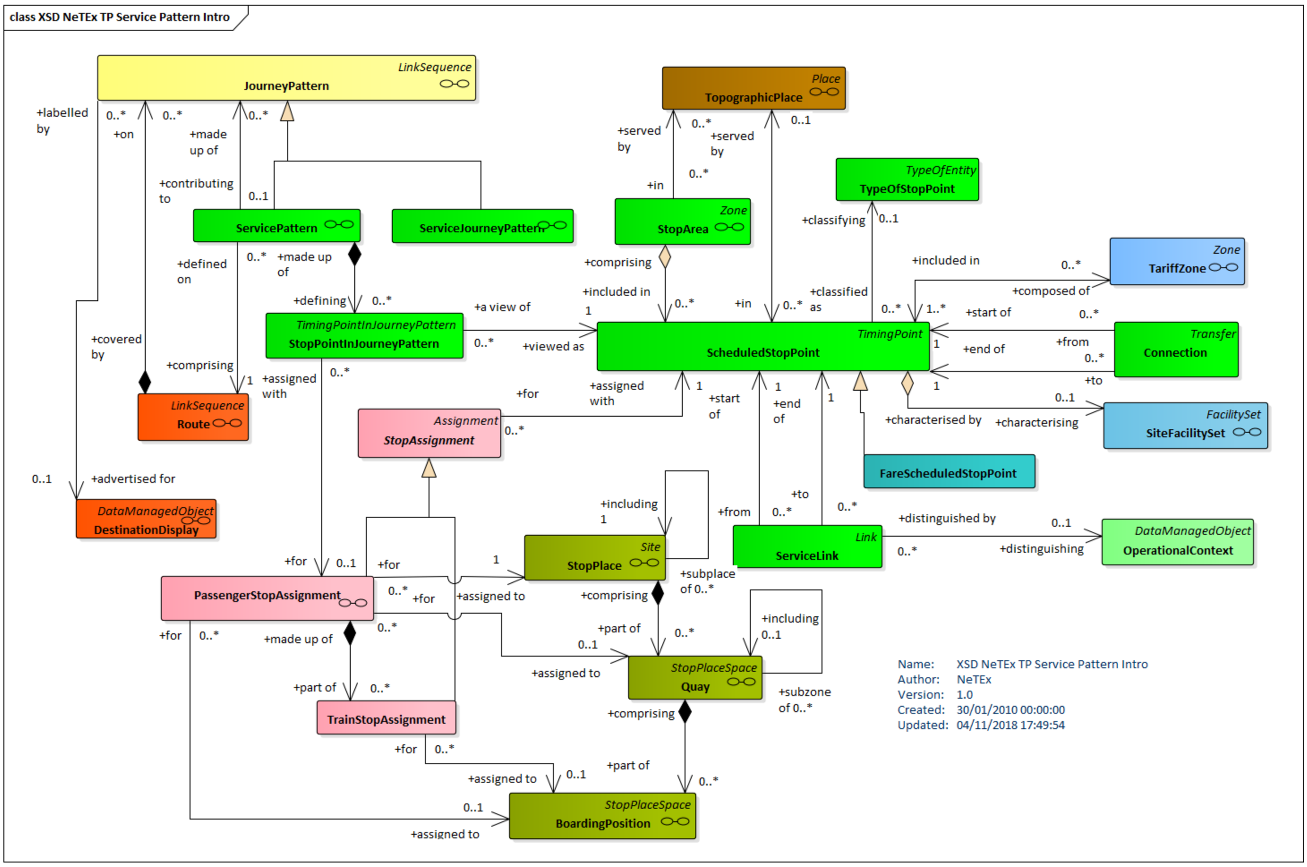


Figure 727– Eurostar Example – Summary Service Patterns

8.5.4.3 Service Pattern – Physical Model

8.5.4.3.1 Service Pattern Overview – Physical Model

The following figure gives an overview of the SERVICE PATTERN physical model.



8.5.4.3.2 Service Pattern Overview – Physical Model

The following figure shows detailed attributes of the SERVICE PATTERN physical model.

In accordance with the simplification (already described above in description of the Journey Pattern Physical Model) of considering the SCHEDULED STOP POINTs as specifications of TIMING POINTs, JOURNEY PATTERNs may be considered as an ordered sequence of TIMING POINTs.

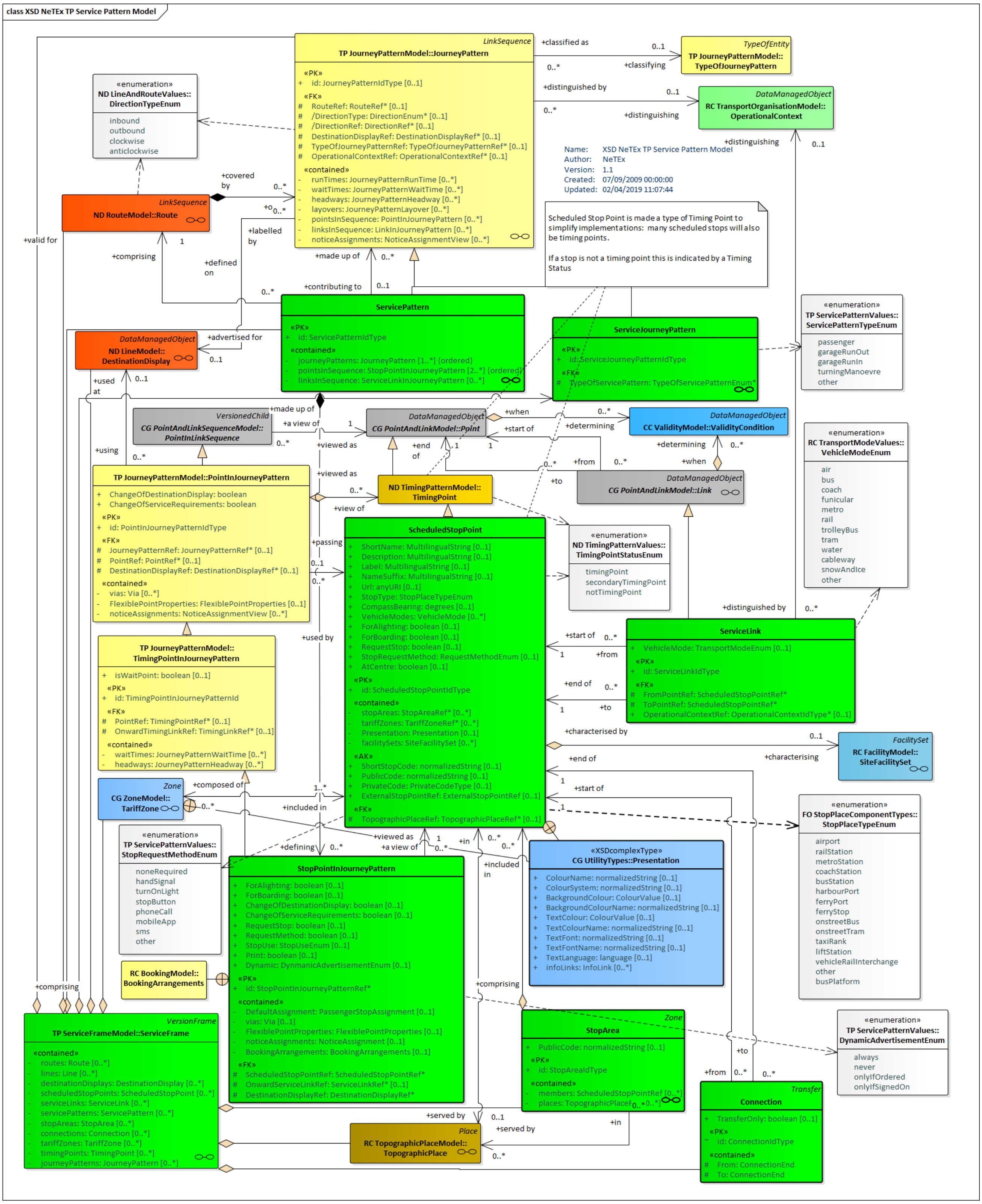


Figure 728 – Service Pattern – Physical Model (UML)

8.5.4.4 Service Pattern – Attributes and XSD

8.5.4.4.1 StopArea – Model Element

A group of SCHEDULED STOP POINTs close to each other.

Table 614 – *StopArea* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Zone*	::>	STOP AREA inherits from ZONE.
«PK»	***id***	*StopAreaIdType*	1:1	Identifier of a STOP AREA.
	PublicCode	*xsd:normalizedString*	0:1	Public code for a STOP AREA.
«FK»	***ParentStopArea-Ref***	*StopAreaRef*	0:1	Reference to any parent STOP AREA of the STOP AREA.
«FK»	***TopographicPlaceRef***	*TopographicPlaceRef*	0:*	TOPOGRAPHIC PLACEs associated with a STOP AREA.

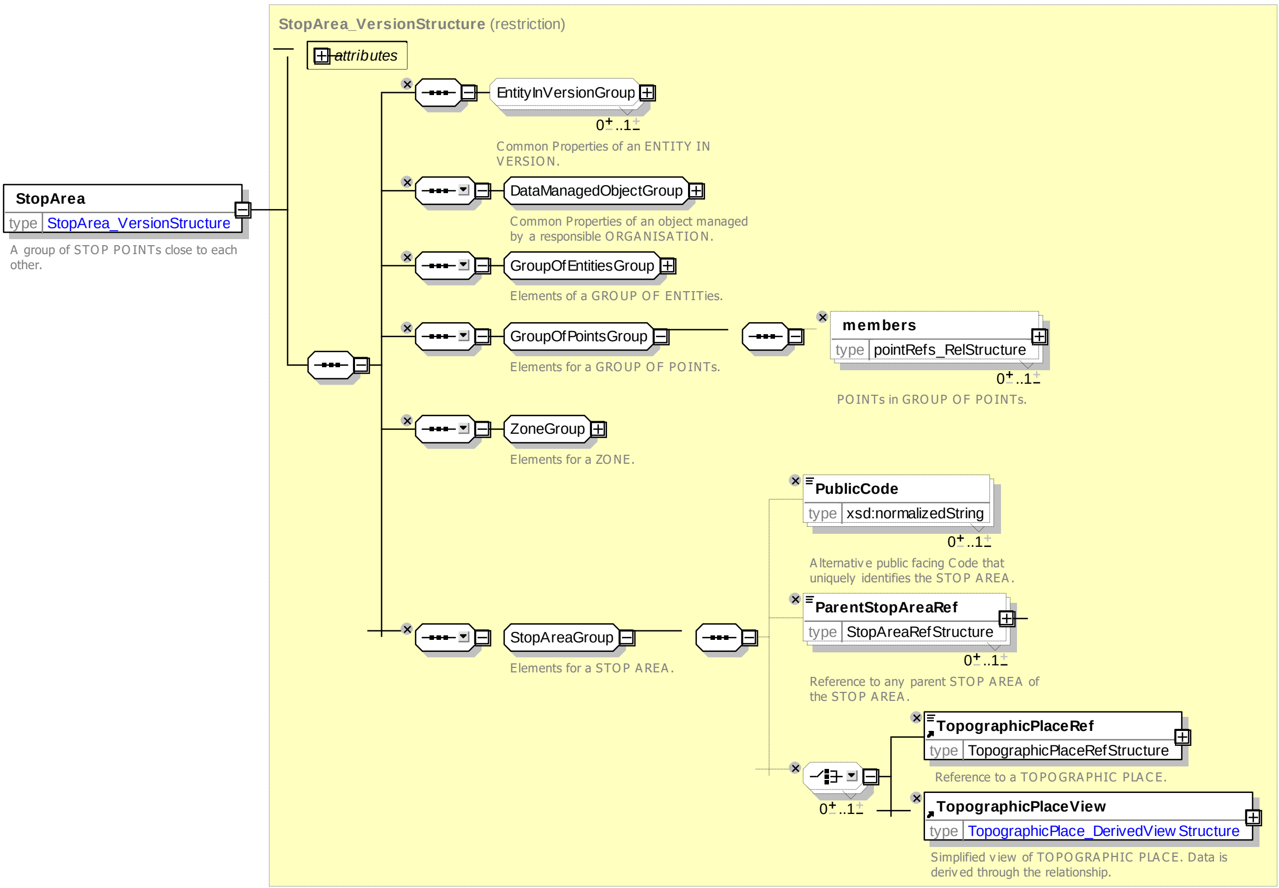


Figure 729 – *StopArea* – *XSD*

8.5.4.4.2 ScheduledStopPoint – Model Element

A POINT where passengers can board or alight from vehicles.

Table 615 – *ScheduledStopPoint* – Element

Classification	Name		Type	Cardinality	Description
::>	::>		*TimingPoint*	::>	SCHEDULED STOP POINT inherits from TIMING POINT.
«PK»	***id***		*ScheduledStopPointIdType*	1:1	Identifier of a SCHEDULED STOP POINT.
«cntd»	***stopAreas***		*StopAreaRef*	0:*	STOP AREAs to which SCHEDULED STOP POINT belongs.
«cntd»	***tariffZones***		*TariffZoneRef*	0:*	TARIFF ZONEs to which SCHEDULED STOP POINT belongs.
XGRP	***ScheduledStop-PointDescription-Group***		xmlGroup	0:1	Elements describing a SCHEDULED STOP POINT.
«enum»	***VehicleModes***		*VehicleMode*Enum	0:*	VEHICLE MODEs associated with SCHEDULED STOP POINT. See TRANSPORT MODE for allowed values.
	ForAlighting		*xsd:boolean*	0:1	Default for whether stop may be used for alighting. May be overridden on specific services.
	ForBoarding		*xsd:boolean*	0:1	Default for whether stop may be used for boarding. May be overridden on specific services.
	RequestStop		*xsd:boolean*	0:1	Default for whether stop is a request stop. May be overridden on specific services.
«enum»	***RequestMethod***		*RequestMethodEnum*	0:1	Method of request stop. The default is '*noneRequired'*. See allowed values below.+ v1.1
«FK»	***CountryRef***		*CountryRef*	0:1	Reference to a country ISO 3166-1
			CHOICE		Can be reference or inline.
«FK»	***a***	***Topographic-PlaceRef***	*TopographicPlaceRef*	0:1	Principle TOPOGRAPHIC PLACE associated with SCHEDULED STOP POINT.
«FV»	***b***	***Topographic-PlaceView***	*TopographicPlaceView*	0:1	Principle TOPOGRAPHIC PLACE associated with SCHEDULED STOP POINT.
	AtCentre		*xsd:boolean*	0:1	Whether STOP POINT can be considered as being at the centre of a TOPOGRAPHIC PLACE. The default is '*false'*.

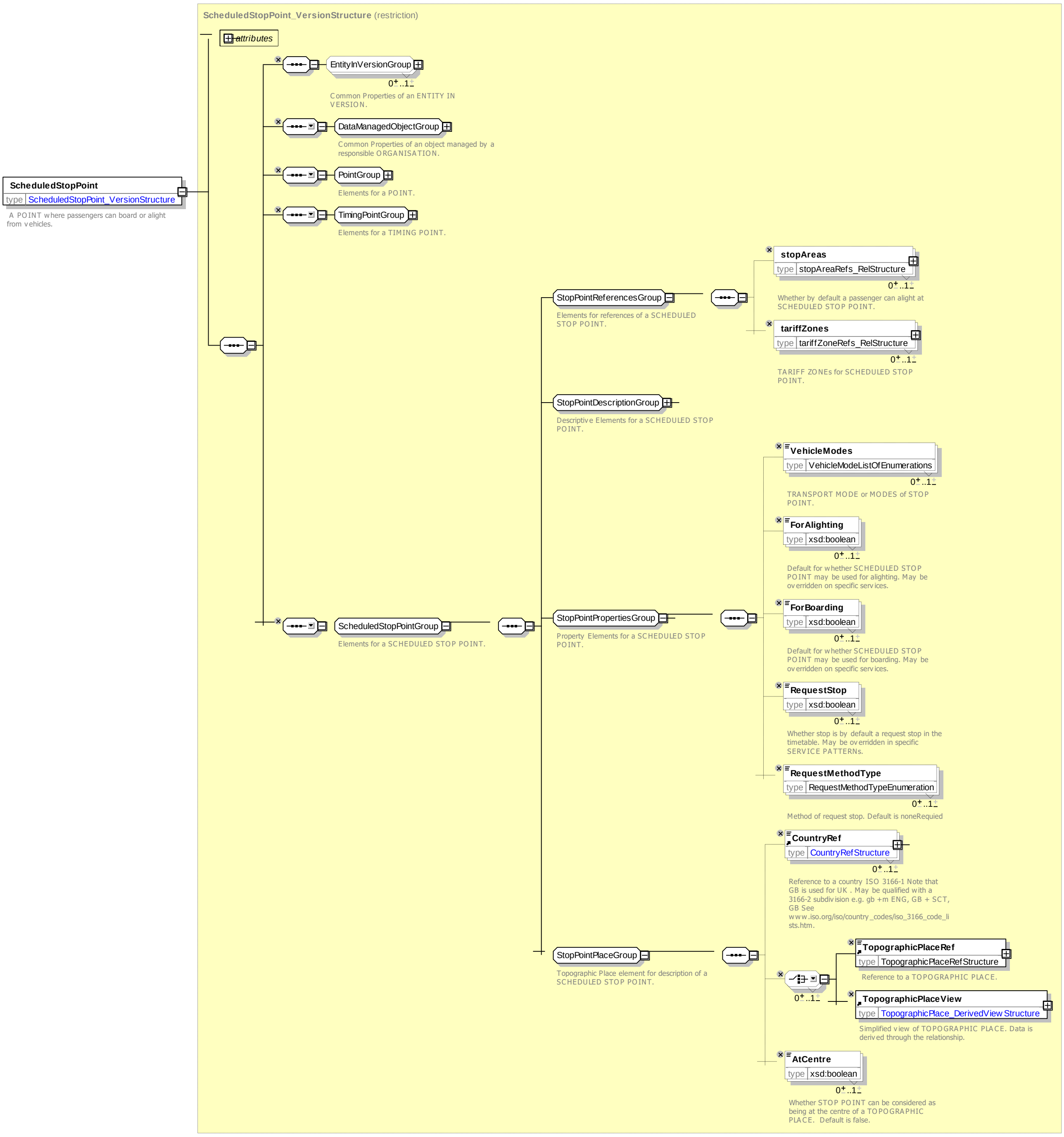


Figure 730 – *ScheduledStopPoint – XSD*

8.5.4.4.2.1 ScheduledStopPointDescriptionGroup – XML Group

Elements describing a SCHEDULED STOP POINT.

Table 616 – *ScheduledStopDescriptionPoint* – Group

Classification	Name	Type	Cardinality	Description
	ShortName	*MultilingualString*	0:1	Short Name of SCHEDULED STOP POINT.

	NameSuffix	*MultilingualString*	0:1	Suffix to append to name in some contexts.
	Description	*MultilingualString*	0:1	Description of SCHEDULED STOP POINT.
	Label	*MultilingualString*	0:1	Label of SCHEDULED STOP POINT.
«AK»	***ShortStopCode***	*xsd:normalizedString*	0:1	Short identifier of a SCHEDULED STOP POINT.
«AK»	***PublicCode***	*xsd:normalizedString*	0:1	Public Code of a SCHEDULED STOP POINT.
«AK»	***PrivateCode***	*xsd:normalizedString*	0:1	Alternative identifier of a SCHEDULED STOP POINT. can be used to associate with legacy systems.
«AK»	***ExternalStopPointRef***	*ExternalObjectRef*	0:1	An alternative code that uniquely identifies the SCHEDULED STOP POINT specifically for use in AVMS systems. For VDV compatibility. +v1.1
	Url	*xsd:anyURI*	0:1	URL associated with SCHEDULED STOP POINT.
«enum»	***StopType***	*StopPlaceTypeEnum*	1:1	Type of STOP PLACE. See STOP PLACE for allowed values.
	CompassBearing	*AbsoluteBearingType*	0:1	Heading of STOP relative to street. Degrees from North. This should be considered as a derived value that can be used for presentation purposes when information about the physical stop is not available. The definitive value is the compass bearing found on the QUAY (i.e. physical stop) to which a SCHEDULED STOP POINT is assigned.
«cntd»	***Presentation***	*Presentation*	0:1	Presentation values to use when rendering STOP POINT such as a colour. and font. See Utility Types model.

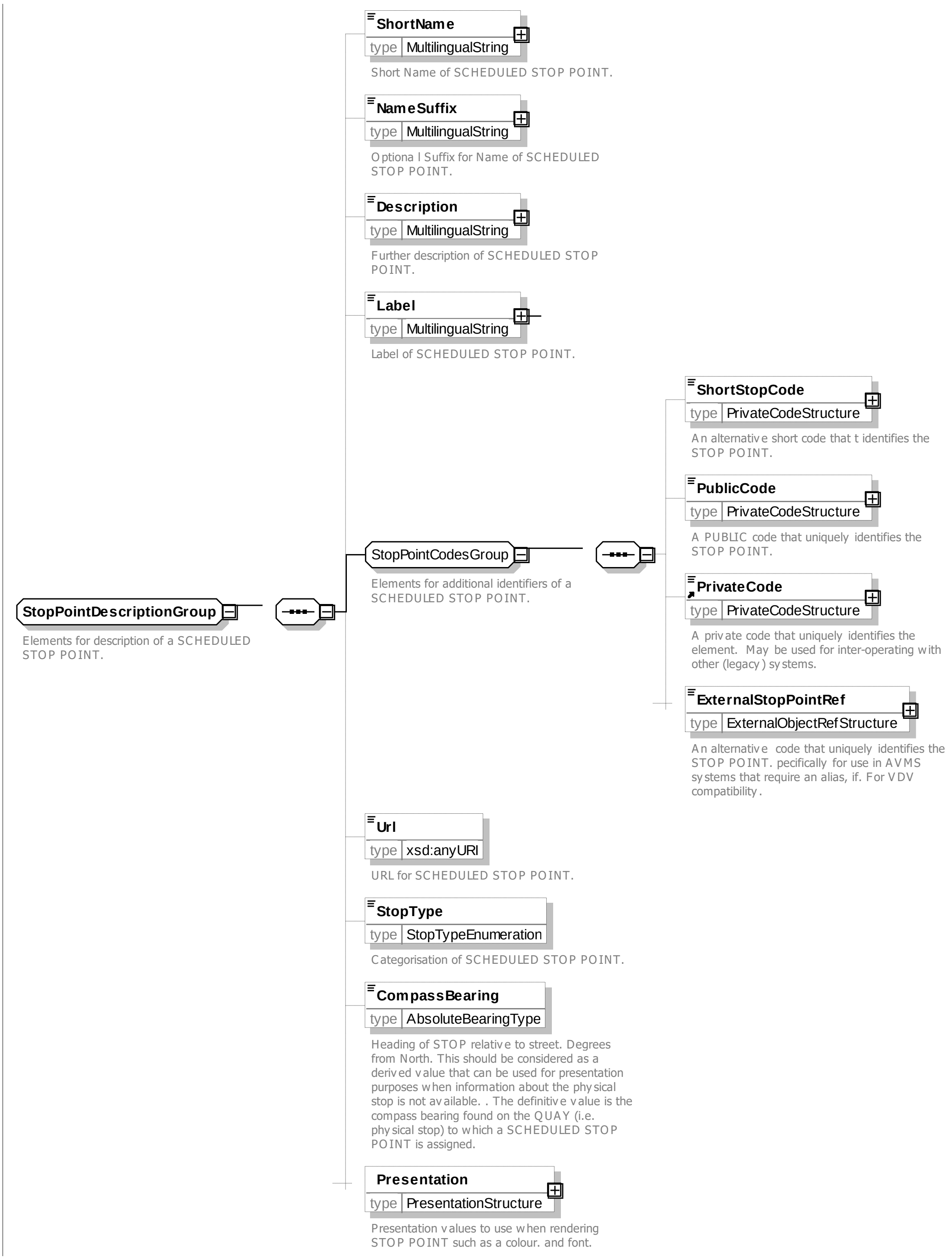


Figure 731 – *ScheduledStopPointDescriptionGroup – XSD*

8.5.4.4.3 ServiceLink – Model Element

A LINK between an ordered pair of SCHEDULED STOP POINTs.

Table 617 – *ServiceLink* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*Link*	::>	SERVICE LINK inherits from LINK
«PK»	***id***	*ServiceLinkIdType*	1:1	Identifier of a SERVICE LINK.
«FK»	***FromPointRef***	*ScheduledStopPointRef*	1:1	Reference to SCHEDULED STOP POINT at which SERVICE LINK starts.
«FK»	***ToPointRef***	*ScheduledStopPointRef*	1:1	Reference to SCHEDULED STOP POINT at which SERVICE LINK ends.
«enum»	***VehicleMode***	*TransportModeEnum*	0:1	Mode of transport for which SERVICE LINK applies. See TRANSPORT MODE for allowed values.
«FK»	***Operational-ContextRef***	*OperationalContextRef*	0:1	Reference to OPERATIONAL CONTEXT used to distinguish SERVICE LINK from similar links between the same two points.

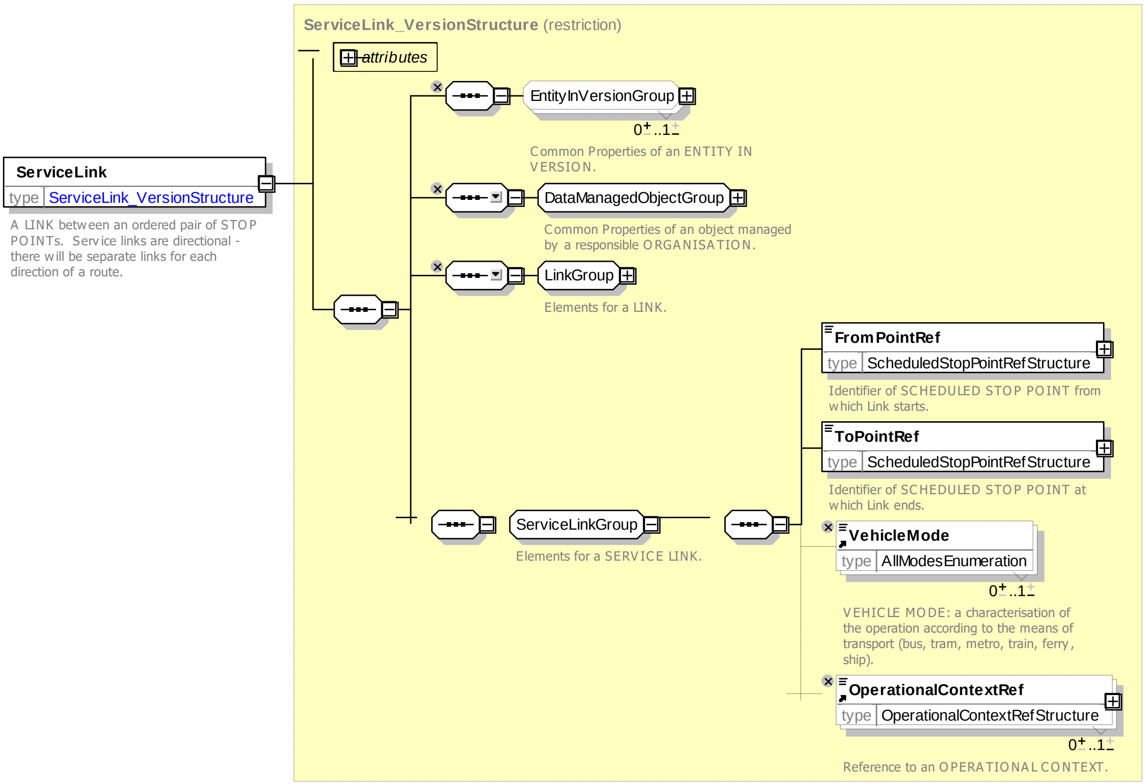


Figure 732 – *ServiceLink – XSD*

8.5.4.4.4 ServicePattern – Model Element

The subset of a JOURNEY PATTERN made up only of STOP POINTs IN JOURNEY PATTERN.

Table 618 – *ServicePattern* – Element

Classification	Name	Type	Cardinality	Description

::>	::>	*JourneyPattern*	::>	SERVICE PATTERN inherits from LINKS SEQUENCE GROUP and embeds the properties of a JOURNEY PATTERN.
«PK»	***id***	*ServicePatternIdType*	1:1	Identifier of a SERVICE PATTERN.
«FK»	***journeyPatterns***	*JourneyPatternRef*	0:*	JOURNEY PATTERNs associated with SERVICE PATTERN.
«cntd»	***pointsInSequence***	*StopPointInJourney-Pattern*	2:*	Ordered sequence of STOP POINTs in JOURNEY PATTERN that makes up SERVICE PATTERN.
«cntd»	***linksInSequence***	*ServiceLinkInJourney-Pattern*	0:*	Ordered sequence of SERVICE LINKs in JOURNEY PATTERN that make up SERVICE PATTERN.

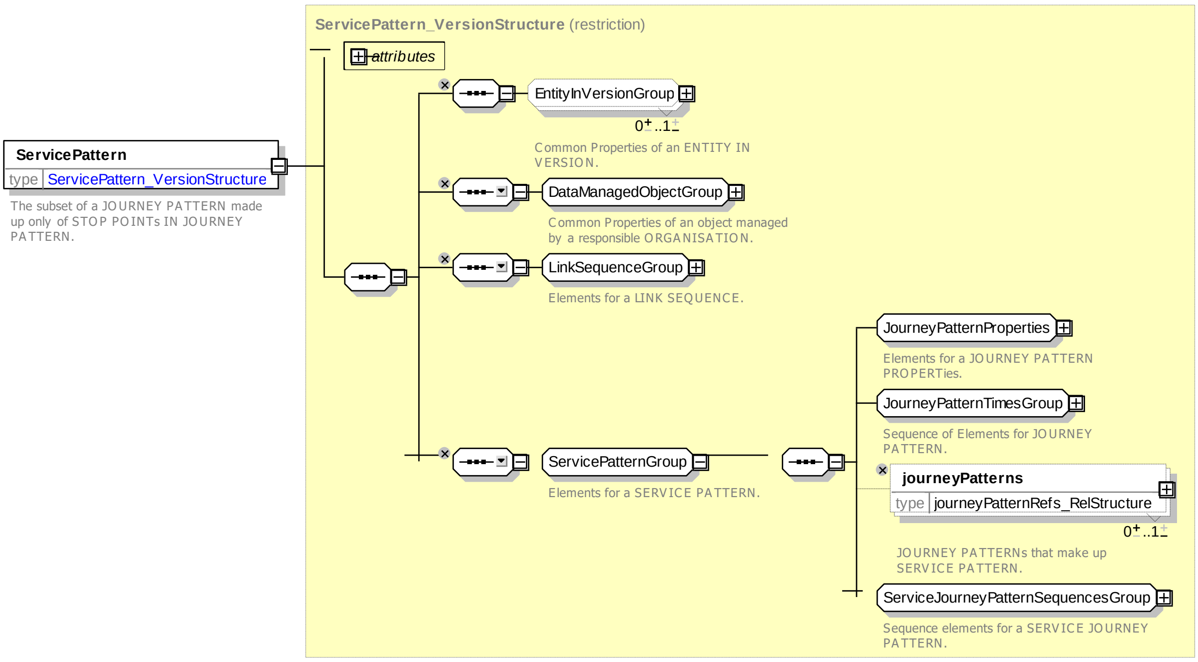


Figure 733 – *ServicePattern – XSD*

8.5.4.4.5 ServiceJourneyPattern – Model Element

The JOURNEY PATTERN for a (passenger carrying) SERVICE JOURNEY.

Table 619 – *ServiceJourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*JourneyPattern*	::>	SERVICE JOURNEY PATTERN inherits from JOURNEY PATTERN.
«PK»	***id***	*ServiceJourneyPattern-IdType*	1:1	Identifier of SERVICE JOURNEY PATTERN.

«enum»	***ServiceJourney-PatternType***	*ServicePatternTypeEnum*	1:1	Type of SERVICE JOURNEY PATTERN. See allowed values below.

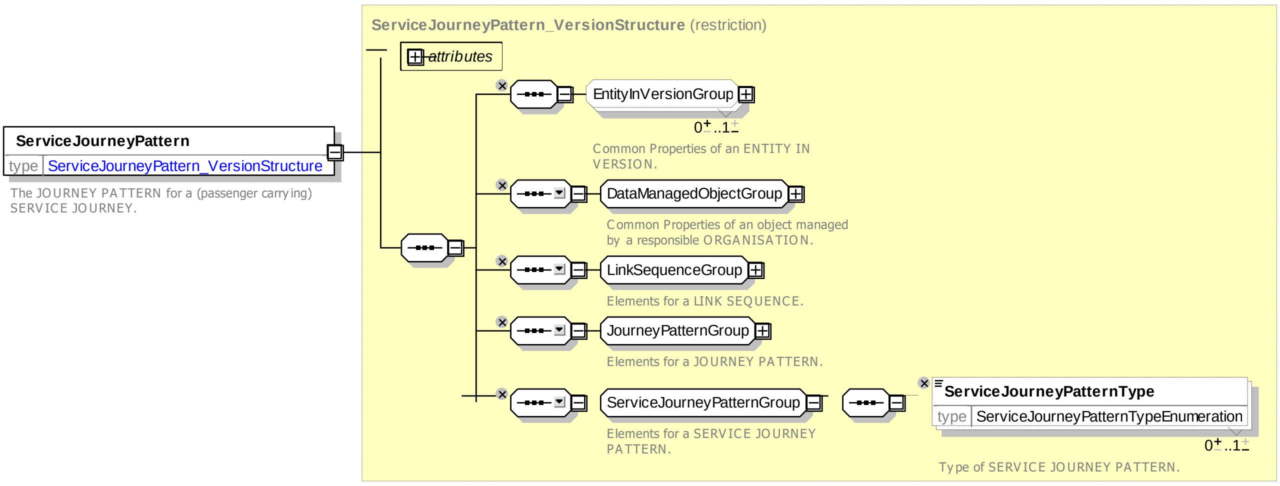


Figure 734 – *ServiceJourneyPattern – XSD*

8.5.4.4.5.1 ServicePatternType – Allowed Values

Allowed values for SERVICE PATTERN type (*ServicePatternTypeEnumeration*)

Table 620 – *ServicePatternType* – Allowed Values

Value	Description
passenger	Run with passengers on board
garageRunOut	Run out from garage to first stop
garageRunIn	Run in from last stop to garage.
turningManoeuvre	Turning manoeuvre
other	Other

8.5.4.4.6 *StopPointInJourneyPattern – Model Element*

A POINT IN JOURNEY PATTERN which is a visit to a SCHEDULED STOP POINT.

Table 621 – *StopPointInJourneyPattern* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*TimingPointInJourneyPattern*</u>	::>	STOP POINT IN JOURNEY PATTERN inherits from TIMING POINT IN JOURNEY PATTERN.
«PK»	***id***	*StopPointInJourneyPatternIdType*	1:1	Identifier of a STOP POINT IN JOURNEY PATTERN.
«FK»	***ScheduledStop-PointRef***	*ScheduledStopPointRef*	1:1	Reference to SCHEDULED STOP POINT which STOP POINT in JOURNEY PATTERN visits. NOTE: overwrite the TimingPointRef inherited from TIMING POINT IN JOURNEY PATTERN

«FK»	***OnwardTiming-LinkRef***	*ServiceLinkRef*	0:1	Next TIMING LINK after visit to STOP POINT in JOURNEY PATTERN.
	isWaitPoint	*xsd:boolean*	0:1	Whether point is a wait point as a default.
«ctnd»	***waitTimes***	*JournePatternWaitTime*	0:*	Set of JOURNEY PATTERN WAIT TIMEs that may apply to TIMING POINT.
XGRP	***StopPointIn-PatternTiming-Group***	***xmlGroup***	0:*	Elements relatimg to timing for JOURNEY PATTERN may apply to TIMING POINT. See below,
«FK»	***OnwardService-LinkRef***	*ServiceLinkRef*	0:1	Next SERVICE LINK after visit to STOP POINT in JOURNEY PATTERN.
	ForAlighting	*xsd:boolean*	0:1	Whether stop may be used for alighting.
	ForBoarding	*xsd:boolean*	0:1	Whether stop may be used for boarding.
XGRP	***StopPointIn-PatternProperties-Group***	***xmlGroup***	0:*	Elements describing STOP POINT IN JOURNEY. See below.

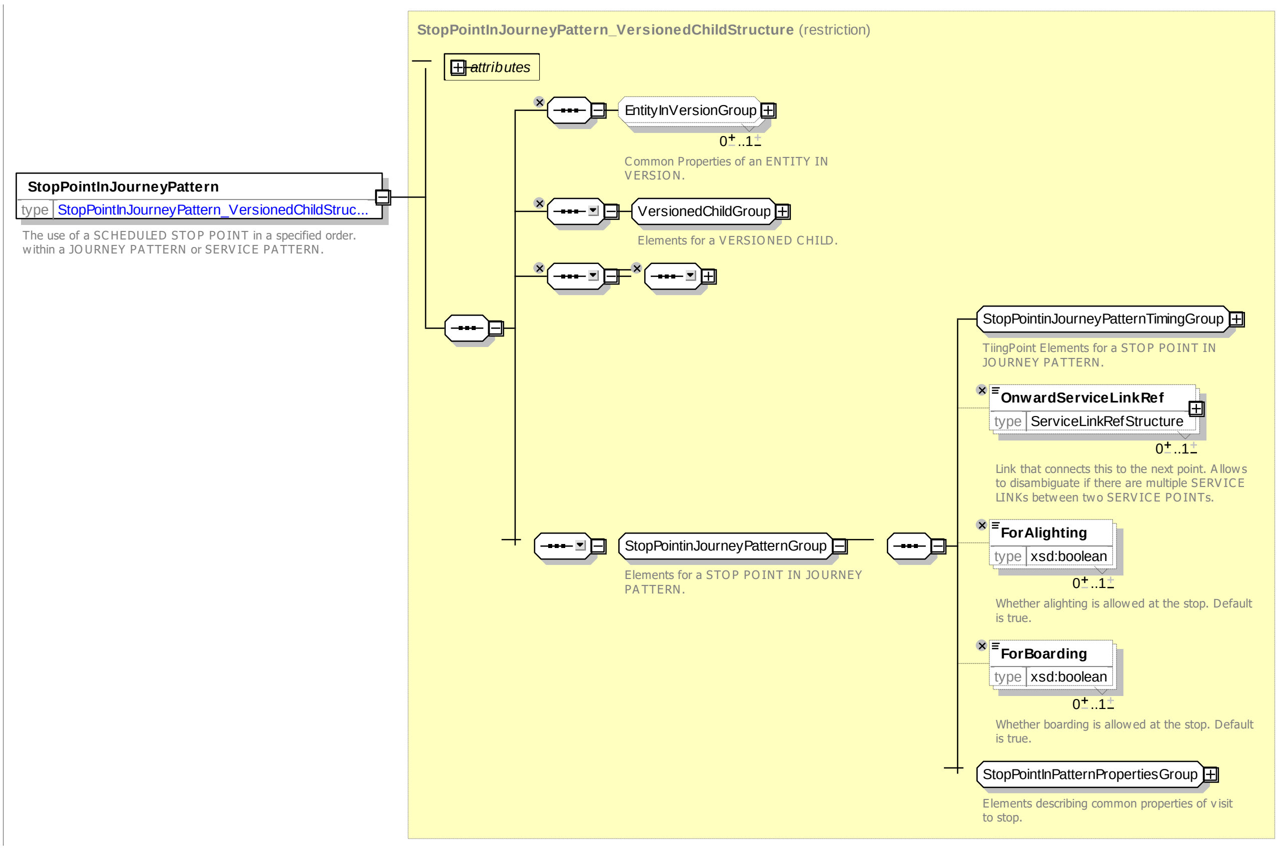


Figure 735 – *StopPointInJourneyPattern – XSD*

8.5.4.4.6.1 StopPointInPatternTimingGroup – XML Group

The ***StopPointInPatternTimingGroup*** holds timing related properties for a SERVICE JOURNEY PATTERN.

Table 622 – *StopPointInPatternTimingGroup* – Group

Classification	Name	Type	Cardinality	Description
«FK»	***ScheduledStop-PointRef***	*ScheduledStopPointRef*	1:1	Reference to SCHEDULED STOP POINT which STOP POINT in JOURNEY PATTERN visits. NOTE: overwrite the TimingPointRef inherited from TIMING POINT IN JOURNEY PATTERN
«FK»	***OnwardTiming-LinkRef***	*ServiceLinkRef*	0:1	Next TIMING LINK after visit to STOP POINT in JOURNEY PATTERN.
	isWaitPoint	*xsd:boolean*	0:1	Whether point is a wait point as a default.
«ctnd»	***waitTimes***	*JourneyPatternWaitTime*	0:*	Set of JOURNEY PATTERN WAIT TIMEs that may apply to TIMING POINT.
«ctnd»	***headways***	*JournePatternHeadway*	0:*	Set of JOURNEY PATTERN HEADWAYs that may apply to TIMING POINT.

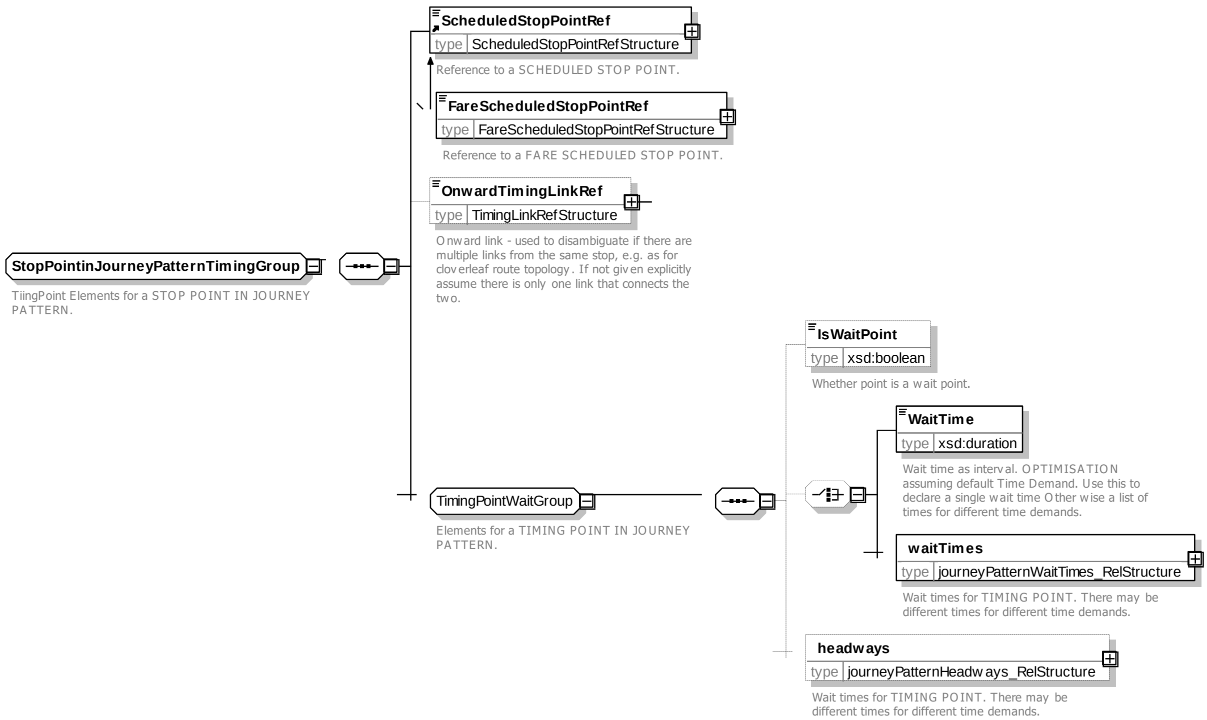


Figure 736 – *StopPointInJourneyPatternTimingGroup – XSD*

8.5.4.4.6.2 StopPointInPatternPropertiesGroup – XML Group

The ***StopPointInPatternPropertiesGroup*** holds common properties for a SERVICE JOURNEY PATTERN.

Table 623 – *StopPointInPatternPropertiesGroup* – Group

Classification	Name	Type	Cardinality	Description

«FK»	***Destination-DisplayRef***		*DestinationDisplayRef*	0:1	DESTINATION DISPLAY values to show at this point (also available as a ***DestinationDisplayView***).
«ctnd»	***vias***		CHOICE		
	a	***None***	*EmptyType*	0:1	There are no vias shown.
	b	***Via***	*Via*	0:*	Destinations that the service is show nas going via from this point.
«ctnd»	***FlexiblePoint-Properties***		*FlexiblePointProperties*	0:1	Flexible properties of a LINK.
	ChangeOf-Destination-Display		*xsd:boolean*	0:1	Whether DESTINATION DISPLAY changes at this point.
	ChangeOfService Requirements		*xsd:boolean*	0:1	Whether SERVICE REQUIREMENTs change at this point.
«ctnd»	***notice-Assignments***		*NoticeAssignment*	0:*	NOTICEs for POINT IN JOURNEY PATTERN.
	RequestStop		*xsd:boolean*	0:1	Whether stop is a request stop for this journey. The default is '*false*'.
«enum»	***RequestMethod***		*RequestMethodEnumeration*	0:1	Method of requesting stop. See allowed values below. The default is '*noneRequired'*. +v1.
«enum»	***StopUse***		*StopUseEnumeration*	0:1	Nature of use of stop, e.g. access, interchange only, or pass through. See allowed values below. The default is '*access'*.
«ctnd»	***Booking-Arrangements***		*BookingArrangements*	0:1	BOOKING ARRANGEMENTs for stop if different from those for SERVICE JOURNEY. +v1.1
	Print		*xsd:boolean*	0:1	Whether the stop is included in printed media. The default is '*true'*. +v1.1
«enum»	***Dynamic***		*DynamicAdvertisementEnum*	0:1	When the STOP POINT IN PATTERN is to be publicised in dynamic media. The default is '*always'*. See values below. +v1.1

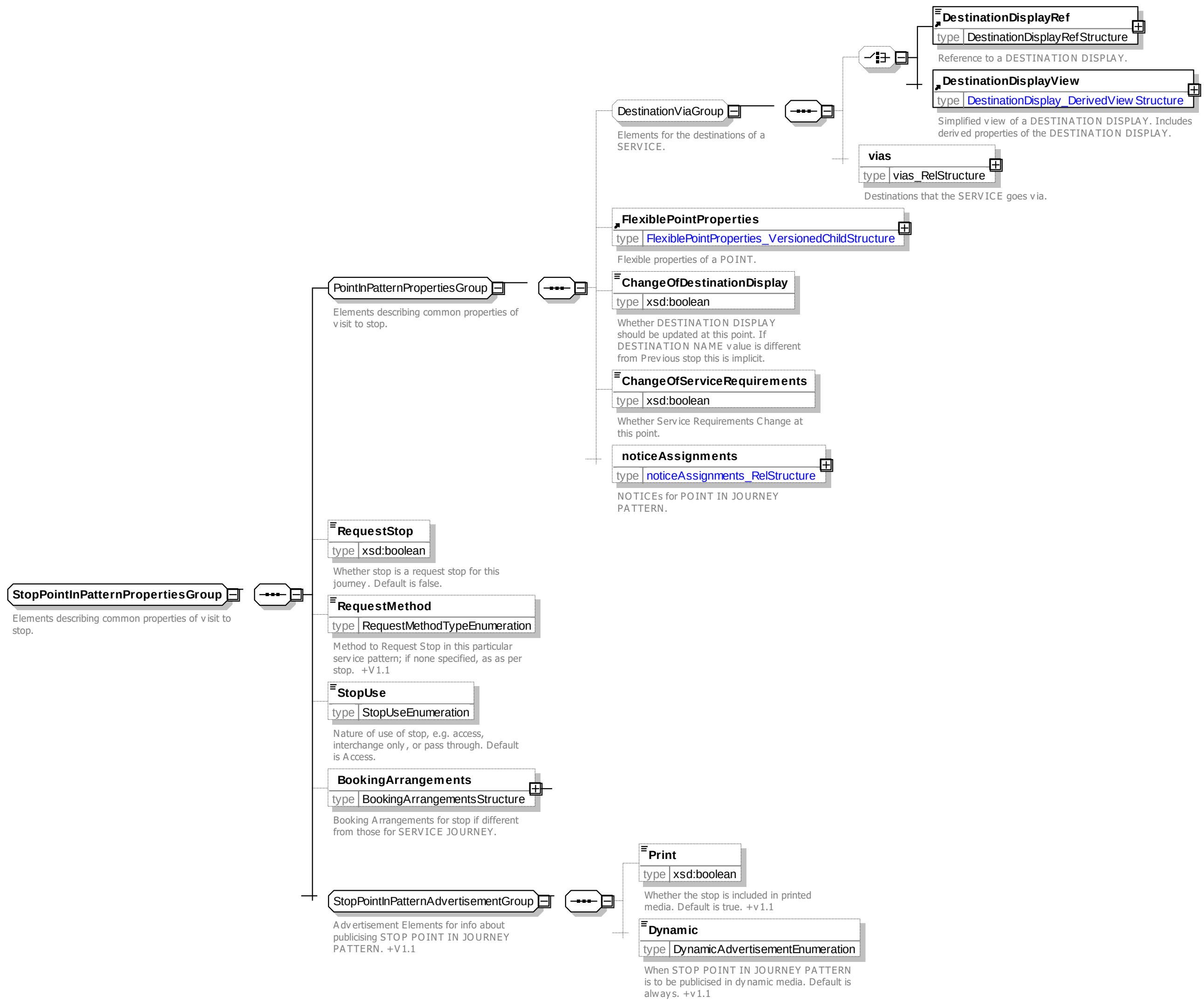


Figure 737 – *StopPointInPatternPropertiesGroup – XSD*

***StopUse*Type – Allowed Values**

Allowed values for ***StopUse*** type (*StopUseEnumeration*)

Table 624 – *StopUseType* – Allowed Values

Value	**Description**
access	Access
interchangeOnly	Interchange only
passthrough	Pass through
noBoardingOrAlighting	No boarding or alighting

***RequestMethod*Type – Allowed Values**

Allowed values for ***RequestMethod*** type (*RequestMethodTypeEnumeration*)

Table 625 – *RequestMethodType* – Allowed Values

Value	**Description**
noneRequired	No action required to request stop

handSignal	Make hand signal to request stop
turnOnLight	Call number to request stop
stopButton	Press button at stop to request stop
phoneCall	Call number to request stop
mobileApp	Use mobile Application to request stop
sms	Use SMS to request stop
other	Use other method to request stop

DynamicAdvertisement – Allowed Values

Allowed values for ***DynamicAdvertisement*** type (*DynamicAdvertisement Enumeration*)

Table 626 – *DynamicAdvertisement* – Allowed Values

Value	Description
always	Always show stop.
never	Never show stop.
*onlyIf**Xxxxx**ed*	Only show stop to users who have ordrered transport to it.
onlyIfSignedOn	Only show stop to users who are logged on to the system

8.5.4.4.7 *Connection – Model Element*

The physical (spatial) possibility for a passenger to change from one public transport vehicle to another to continue the trip, determined by two SCHEDULED STOP POINTs. Different times may be necessary to cover the link between these points, depending on the kind of passenger.

Table 627 – *Connection* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	<u>*Transfer*</u>	*::>*	CONNECTION inherits from TRANSFER.
«PK»	***id***	*ConnectionIdType*	1:1	Identifier of a CONNECTION.
«AK»	***ExternalStopPointRef***	*ExternalObjectRef*	0:1	An alternative code that uniquely identifies the CONNECTION specifically for use in AVMS systems. *NOTE For VDV compatibility.*
«cntd»	***From***	<u>*ConnectionEnd*</u>	1:1	Properties of end at which CONNECTION starts.
«cntd»	***To***	<u>*ConnectionEnd*</u>	1:1	Properties of end at which CONNECTION ends.
	TransferOnly	*xsd:boolean*	0:1	Whether when connecting at this stop passengers may only transfer and may not enter or exit at the station. If ‘true’ can only transfer.

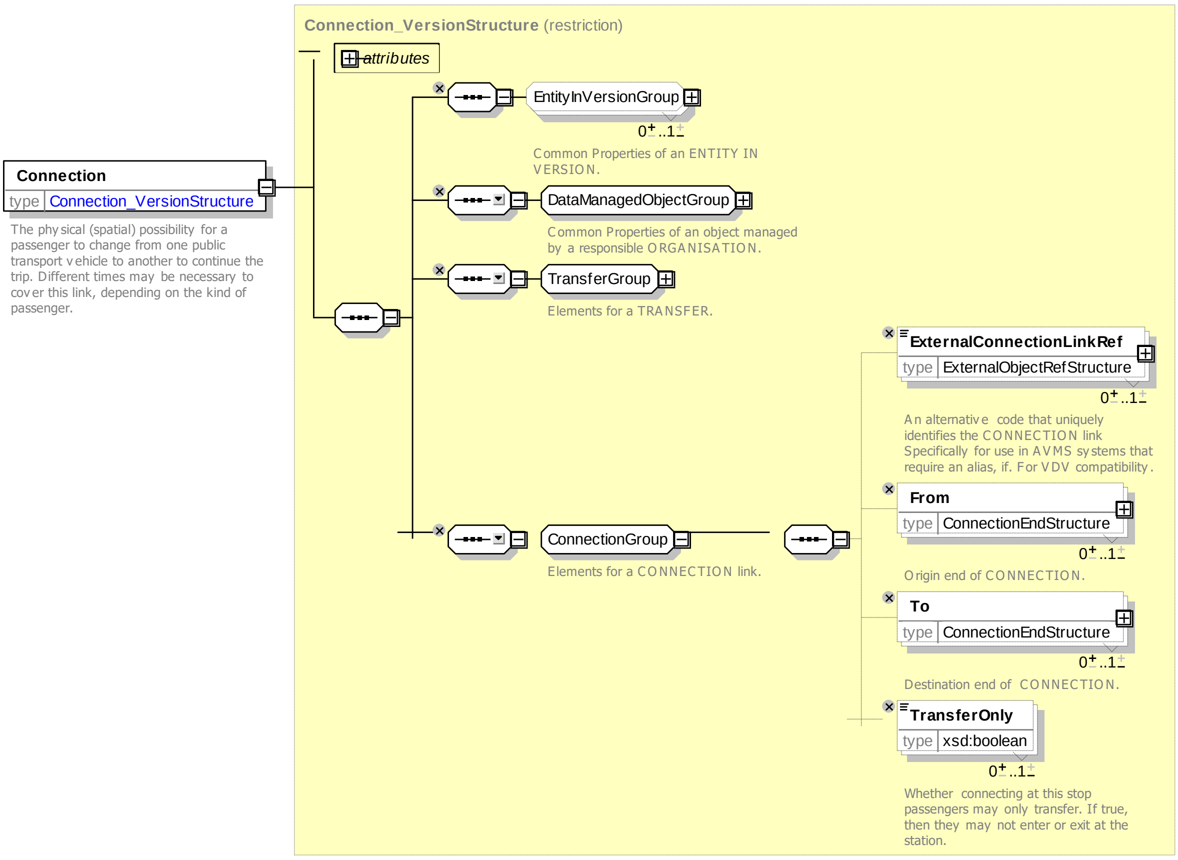


Figure 738 – *Connection – XSD*

8.5.4.4.8 ConnectionEnd – Model Element

One end of a CONNECTION link.

Table 628 – *ConnectionEnd* – Element

Classification	Name	Type	Cardinality	Description
«enum»	***Mode***	*TransportModeEnum*	0:1	Reference to mode of SCHEDULED STOP POINT to which connection connects. See TRANSPORT MODE for allowed values
«FK»	***ScheduledStop-PointRef***	*ScheduledStopPointRef*	0:1	Reference to SCHEDULED STOP POINT to which connection connects.

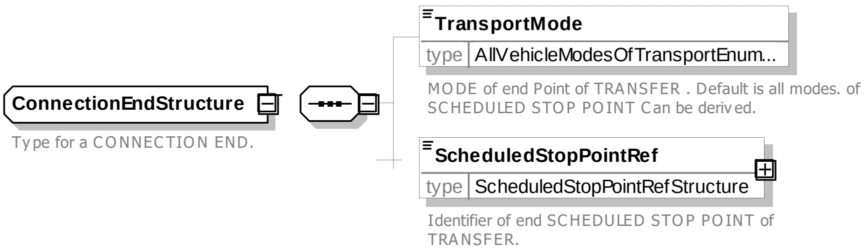


Figure 739 – *ConnectionEnd – XSD*

8.5.4.5 XML Examples for Service Pattern

8.5.4.5.1 Service Points – XML Example

The following XML code fragment shows definitions of four SCHEDULED STOP POINTs (‘*Alpha*’, ‘*Bravo*’, ‘*Charley*‘ and ‘*Delta*’) and three SERVIICE LINKs between them.

EXAMPLE XML Example of SCHEDULED STOP POINTs

```
<scheduledStopPoints>
    <ScheduledStopPoint version="any" id="mybus::SSP_001">
        <Name>Alpha & Castle</Name>
        <Location>
            <Longitude>53.0000</Longitude>
            <Latitude>0.1000</Latitude>
        </Location>
        <StopType>onstreetBus</StopType>
    </ScheduledStopPoint>
    <ScheduledStopPoint version="any" id="mybus::SSP_002">
        <Name>Bravo Street</Name>
        <Location>
            <Longitude>53.2000</Longitude>
            <Latitude>0.2000</Latitude>
        </Location>
        <PublicCode>BRAV</PublicCode>
        <StopType>onstreetBus</StopType>
    </ScheduledStopPoint>
    <ScheduledStopPoint version="any" id="mybus:SSP_077">
        <Name>Charley Crescent</Name>
        <Location>
            <Longitude>53.3000</Longitude>
            <Latitude>0.3000</Latitude>
        </Location>
        <PublicCode>CHAS</PublicCode>
        <StopType>onstreetBus</StopType>
    </ScheduledStopPoint>
    <ScheduledStopPoint version="any" id="mybus::SSP_078">
        <Name>Delta Lane</Name>
        <Location>
            <Longitude>53.34000</Longitude>
            <Latitude>0.34000</Latitude>
        </Location>
        <StopType>onstreetBus</StopType>
    </ScheduledStopPoint>
</scheduledStopPoints>

<serviceLinks>
    <ServiceLink version="any" id="mybus:SSP_001_to_SSP_002">
        <Name>Alpha & Castle to Bravo</Name>
        <FromPointRef version="any" ref="mybus:SSP_001"/>
        <ToPointRef version="any" ref="mybus:SSP_002"/>
    </ServiceLink>
    <ServiceLink version="any" id="mybus:SSP_002_to_SSP_077">
        <Name>Bravo to Charley</Name>
        <FromPointRef version="any" ref="mybus:SSP_002"/>
        <ToPointRef version="any" ref="mybus:SSP_077"/>
    </ServiceLink>
    <ServiceLink version="any" id="mybus:SSP_077_to_SSP_078">
        <Name>Charley to Delta</Name>
```

```
        <FromPointRef version="any" ref="mybus:SSP_077"/>
        <ToPointRef version="any" ref="mybus:SSP_078"/>
    </ServiceLink>
</serviceLinks>
    -->
```

8.5.4.5.2 Service Pattern – XML Example

The following XML code fragment shows a definition of a SERVICE PATTERN that connects all four (Alpha, Bravo, Charley and Delta). It has both stop sequences and link sequences -0 only one is needed.

EXAMPLE XML Example of SERVICE PATTERN – 1

```
<ServicePatterns>
        <!--Full                              A - B - C - D :     -->
    <ServicePattern version="any" id="hde:svp_24o_ABCD">
        <Name> Alpha to Delta All stops</Name>
        <RouteRef version="any" ref="mybus:RT_24o"/>
        <DirectionType>outbound</DirectionType>
        <journeyPatterns>
            <ServiceJourneyPatternRef version="any" ref="hde:sjp_24o_ABCD"/>
        </journeyPatterns>
        <pointsInSequence>
            <StopPointInJourneyPattern version="any" order="1" id="hde:spijp_24o_ABCD_01">
                <ScheduledStopPointRef version="any" ref="mybus:SSP_001"/>
                <ForAlighting>false</ForAlighting>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="any" order="2" id="hde:spijp_24o_ABCD_02">
                <ScheduledStopPointRef version="any" ref="mybus:SSP_002"/>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="any" order="3" id="hde:spijp_24o_ABCD_03">
                <ScheduledStopPointRef version="any" ref="mybus:SSP_077"/>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="any" order="4" id="hde:spijp_24o_ABCD_04">
                <ScheduledStopPointRef version="any" ref="mybus:SSP_078"/>
                <ForBoarding>false</ForBoarding>
            </StopPointInJourneyPattern>
            </pointsInSequence>
        <linksInSequence>
            <ServiceLinkInJourneyPattern version="any" order="1" id="hde:slijp_24o_ABCD_01">
                <ServiceLinkRef version="any" ref="mybus:SSP_001_to_SSP_002"/>
                </ServiceLinkInJourneyPattern>
            <ServiceLinkInJourneyPattern version="any" order="2" id="hde:slijp_24o_ABCD_02">
                <ServiceLinkRef version="any" ref="mybus:SSP_002_to_SSP_077"/>
            </ServiceLinkInJourneyPattern>
            <ServiceLinkInJourneyPattern version="any" order="3" id="hde:slijp_24o_ABCD_03">
                <ServiceLinkRef version="any" ref="mybus:SSP_077_to_SSP_078"/>
            </ServiceLinkInJourneyPattern>
        </linksInSequence>
    </ServicePattern>
```

8.5.4.5.3 Express Service Pattern – XML Example

The following XML code fragment shows a definition of another SERVICE PATTERN that goes directly between the first and last stop ('*Alpha'*, to '*Delta'* above). It uses the same links but only stops at two stops.

EXAMPLE XML Example of SERVICE PATTERN - 2

```
<ServicePattern version="any" id="hde:svp_24o_AxxD">
            <!-- Express                      A - (B) - (C) - D  -->
```

```xml
        <Name> Alpha to Delta Express</Name>
        <RouteRef version="any" ref="mybus:RT_24o"/>
        <DirectionType>outbound</DirectionType>
        <journeyPatterns>
            <ServiceJourneyPatternRef version="any" ref="hde:sjp_24o_AxxD"/>
        </journeyPatterns>
        <pointsInSequence>
            <StopPointInJourneyPattern version="any" order="1" id="hde::spijp_24o_AxxD_01">
                <ScheduledStopPointRef version="any" ref="mybus::SSP_001"/>
            </StopPointInJourneyPattern>
            <StopPointInJourneyPattern version="any" order="4" id="hde::spijp_24o_AxxD_04">
                <ScheduledStopPointRef version="any" ref="mybus::SSP_078"/>
                <ForBoarding>false</ForBoarding>
            </StopPointInJourneyPattern>
        </pointsInSequence>
        <linksInSequence>
            <ServiceLinkInJourneyPattern version="any" order="1" id="hde::slijp_24o_AxxD_01">
                <ServiceLinkRef version="any" ref="mybus::SSP_001_to_SSP_002"/>
                </ServiceLinkInJourneyPattern>
            <ServiceLinkInJourneyPattern version="any" order="2" id="hde:slijp_24o_AxxD_02">
                <ServiceLinkRef version="any" ref="mybus:SSP_002_to_SSP_077"/>
            </ServiceLinkInJourneyPattern>
            <ServiceLinkInJourneyPattern version="any" order="3" id="hde:slijp_24o_AxxD_03">
                <ServiceLinkRef version="any" ref="mybus:SSP_077_to_SSP_078"/>
            </ServiceLinkInJourneyPattern>
        </linksInSequence>
    </ServicePattern>
```

8.5.5 Common Section

8.5.5.1 COMMON SECTION – Conceptual MODEL

In many public transport networks, particularly in big cities, there will be parts of the network where ROUTEs (and JOURNEY PATTERNs) overlap. In other words, lines may be bundled on parts of the network where passengers have the choice among several lines to reach their destinations.

The fact that several JOURNEY PATTERNs are associated in such a bundled section is not necessarily depending on a common topology. The relevant parts of the network where a synchronised schedule will be offered may be chosen arbitrarily by the schedulers, and they need not cover the whole length of the common itinerary of the ROUTEs and JOURNEY PATTERNs in question. Some common itineraries may not be chosen for schedule synchronisation at all. In addition, a synchronised schedule may be offered for parts of ROUTEs that are not identical, but parallel (e.g. a tramway line on a specific track, besides a road on which synchronised bus lines operate).

Therefore, an entity COMMON SECTION is introduced in order to define those network sections where bundled lines have to be considered jointly in the scheduling process. A COMMON SECTION is determined by a set of JOURNEY PATTERNs, together with a set of POINTs, each of which has to be a view of a POINT IN one of those JOURNEY PATTERNs.

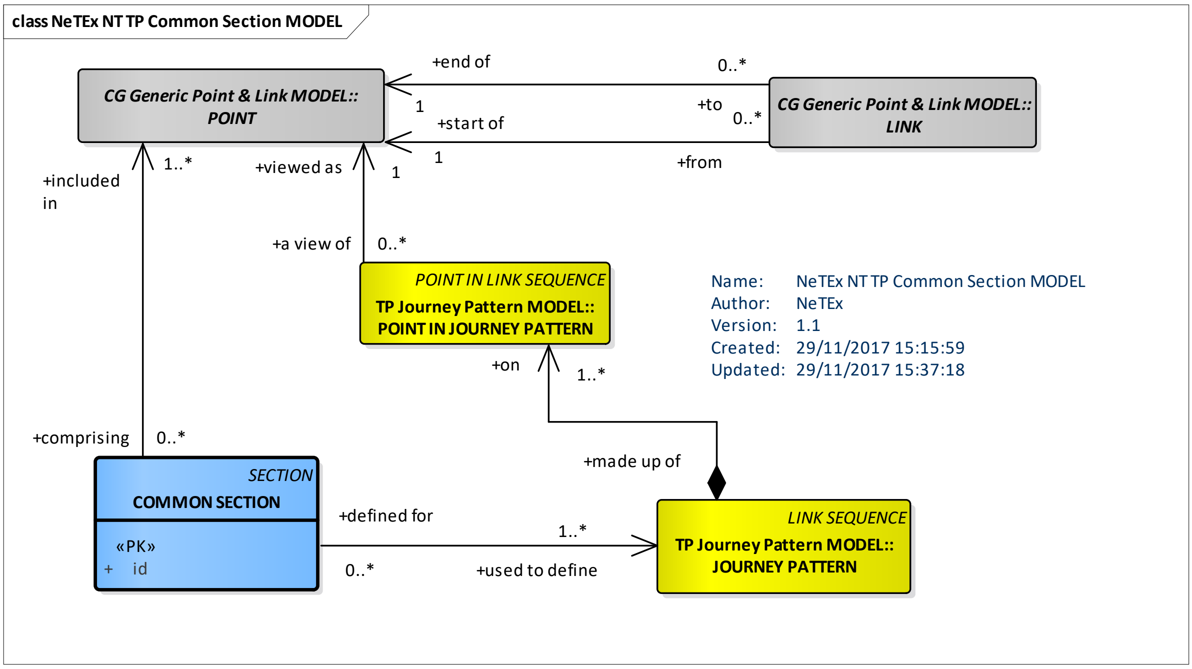


Figure 740 — Common Section – Conceptual MODEL (UML)

8.5.5.2 Common Section – Physical Model

COMMON SECTION is a specialisation of SECTION. A SECTION may include a sequence of POINTs IN LINKSEQUENCE, for example POITs in JOURNEY PATTERN.

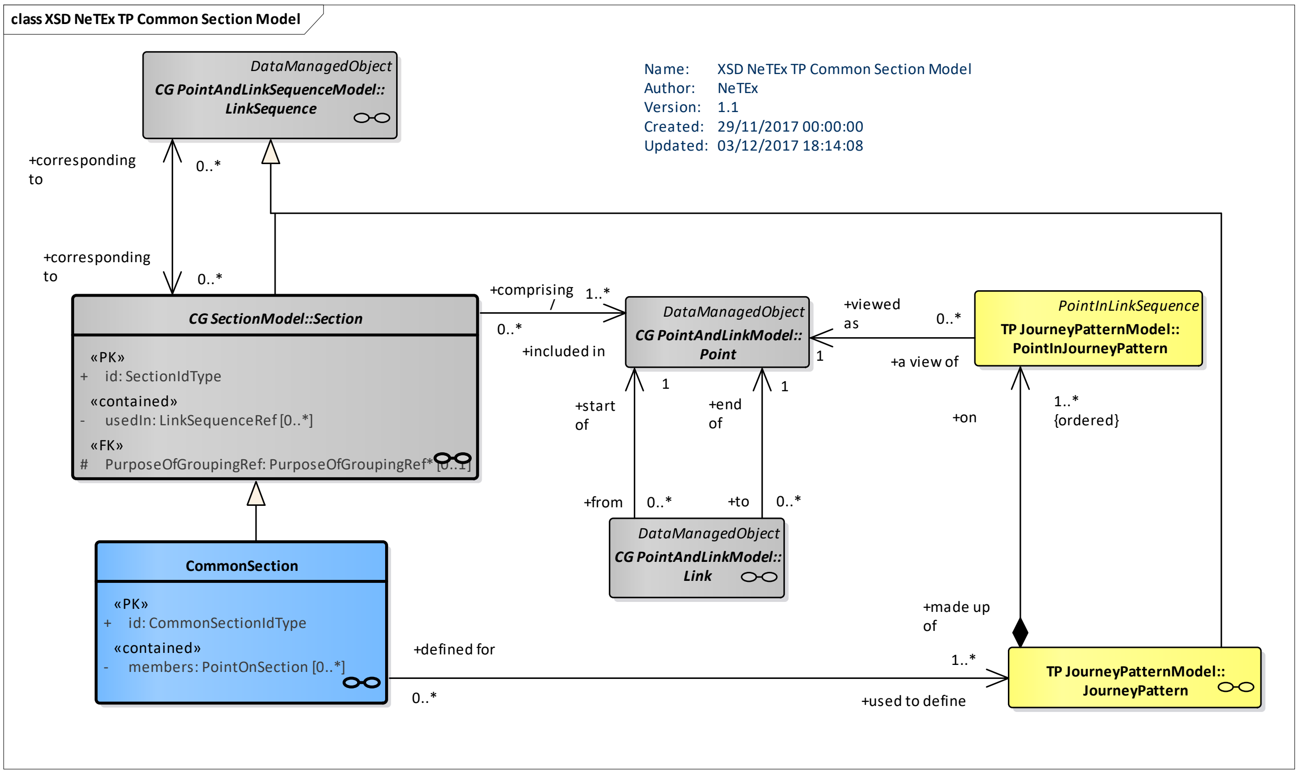


Figure 741 — Common section – Physical Model (UML)

8.5.5.3 Common Section – Attributes and XSD

8.5.5.3.1 CommonSection – Model Element

A part of a public transport network where the ROUTEs of several JOURNEY PATTERNs are going in parallel and where the synchronisation of SERVICE JOURNEYs may be planned and controlled with respect to commonly used LINKs and STOP POINTs. COMMON SECTIONs are defined arbitrarily and need not cover the total lengths of topologically bundled sections.

Table 629 —*CommonSection – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	COMMON SECTION inherits from DATA MANAGED OBJECT.
«PK»	***id***	*CommonSectionIdType*	1:1	Identifier of COMMON SECTION.
«cntd»	***pointsOnSection***	*PointOnSectionr*	0:*	POINTS on SECTION of of COMMON SECTION. See Generic Section Model earliers.
«cntd»	***members***	*PointOnSectionr*	0:*	DEPRECATED use POINTs ON SECTION. See Generic Section earlier.

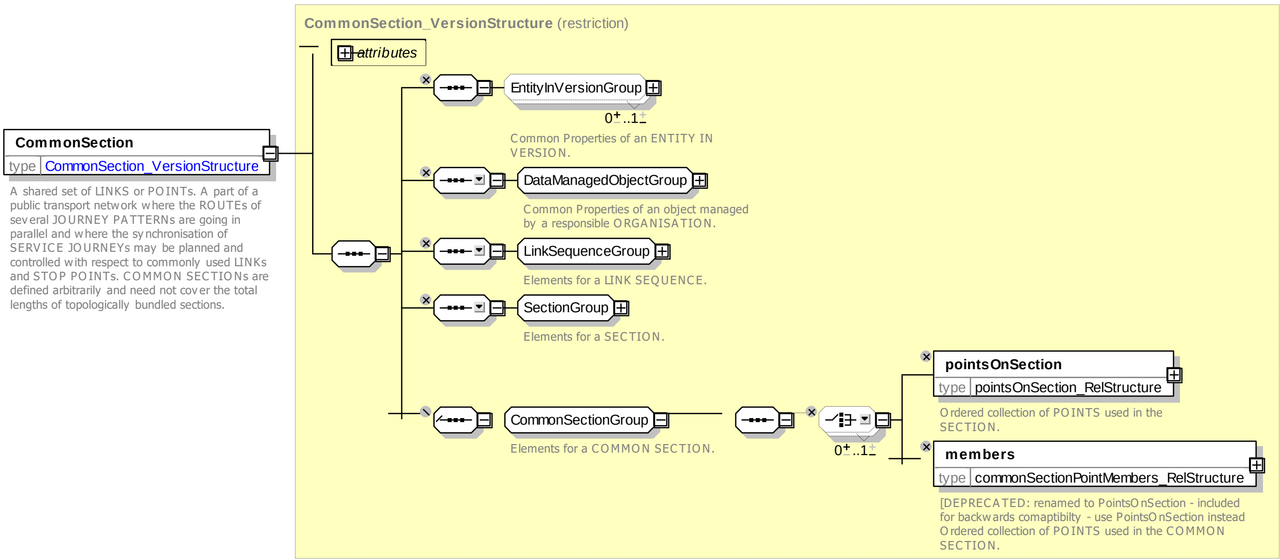


Figure 742 — *CommonSection – XSD*

8.5.5.4 XML Example of Common Section

The following fragment shows an example of a common section.

EXAMPLE

```
<commonSections>
        <!--- ===SCommon Section ========= -->
        <CommonSection version="any" id="hde:CSc001o_JP">
            <Name>A - B -C Common Section: Journey </Name>
            <usedIn>
                <CommonSectionSequenceMember id="hde:CSCM_CSc001o_01">
```

```
                    <LinkSequenceRef        version="any"        xsi:type="JourneyPatternRefStructure"
ref="hde:JP001o"/>
                </CommonSectionSequenceMember>
            </usedIn>
            <members>
                <CommonSectionPointMember id="hde:CSc001o_JP_01">
                    <ScheduledStopPointRef version=" any " ref="hde:SSP0001A"/>
                    <ServiceLinkRef version="001" ref="hde:SL_SSP0001A-SSP0002B"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember id="hde:CSc001o_JP_02">
                    <ScheduledStopPointRef version=" any " ref="hde:SSP0002B"/>
                    <ServiceLinkRef version="001" ref="hde:SL_SSP0002B-SSP0003C"/>
                </CommonSectionPointMember>
                <CommonSectionPointMember id="hde:CSc001o_JP_03">
                    <ScheduledStopPointRef version=" any " ref="hde:SSP0003C"/>
                </CommonSectionPointMember>
            </members>
        </CommonSection>
```

8.5.6 Routing Constraints

8.5.6.1 ROUTING CONSTRAINTs – Conceptual MODEL

In order to manage competition between operators or bus lines, PT authorities sometimes define ROUTING CONSTRAINTs, preventing passenger to from boarding or alighting from a vehicle under certain circumstances.

Several types of constraints may be defined.

Zone based constraints are defined by a ROUTING CONSTRAINT ZONE. The ZONE may be defined by its contained SCHEDULED STOP POINTS or by its boundary points. ZONEs are usually used to express constraints like "If you board in this ZONE, you can't alight in the same ZONE", or "only alighting is permitted in this ZONE". The constaint applies to all the POINTs IN JOURNEY PATTERN of specific LINEs included in the ZONE.

A SERVICE EXCLUSION constraint expresses the fact that the service, on a specific JOURNEY PATTERN (usually a flexible JOURNEY PATTERN) cannot operate when another (regular) service operates. This may occur only on a subpart of the JOURNEY PATTERN, or only on one or some specific SCHEDULED STOP POINTS. This type of constraint is usually defined to prevent a demand responsive service competing with regular lines (meaning that the demand responsive service has to avoid certain areas at hours when the regular lines operate). The model relates the SERVICE EXCLUSION to JOURNEY PATTERN but the "constrained by" relation has to be understood as "*constrained by all the JOURNEYs running on that JOURNEY PATTERN from the time of the first service to the time of the last one*"

TRANSFER RESTRICTIONs are constraints that can be applied on a CONNECTION or INTERCHANGE between two SCHEDULED STOP POINTs, preventing or forbidding the passenger to make a transfer there. They are often used to favour certain connection over others possible on the same couple of line (in order to uses the more appropriate stop of connection, to manage security issues, to manage passenger flow on connections, etc.).

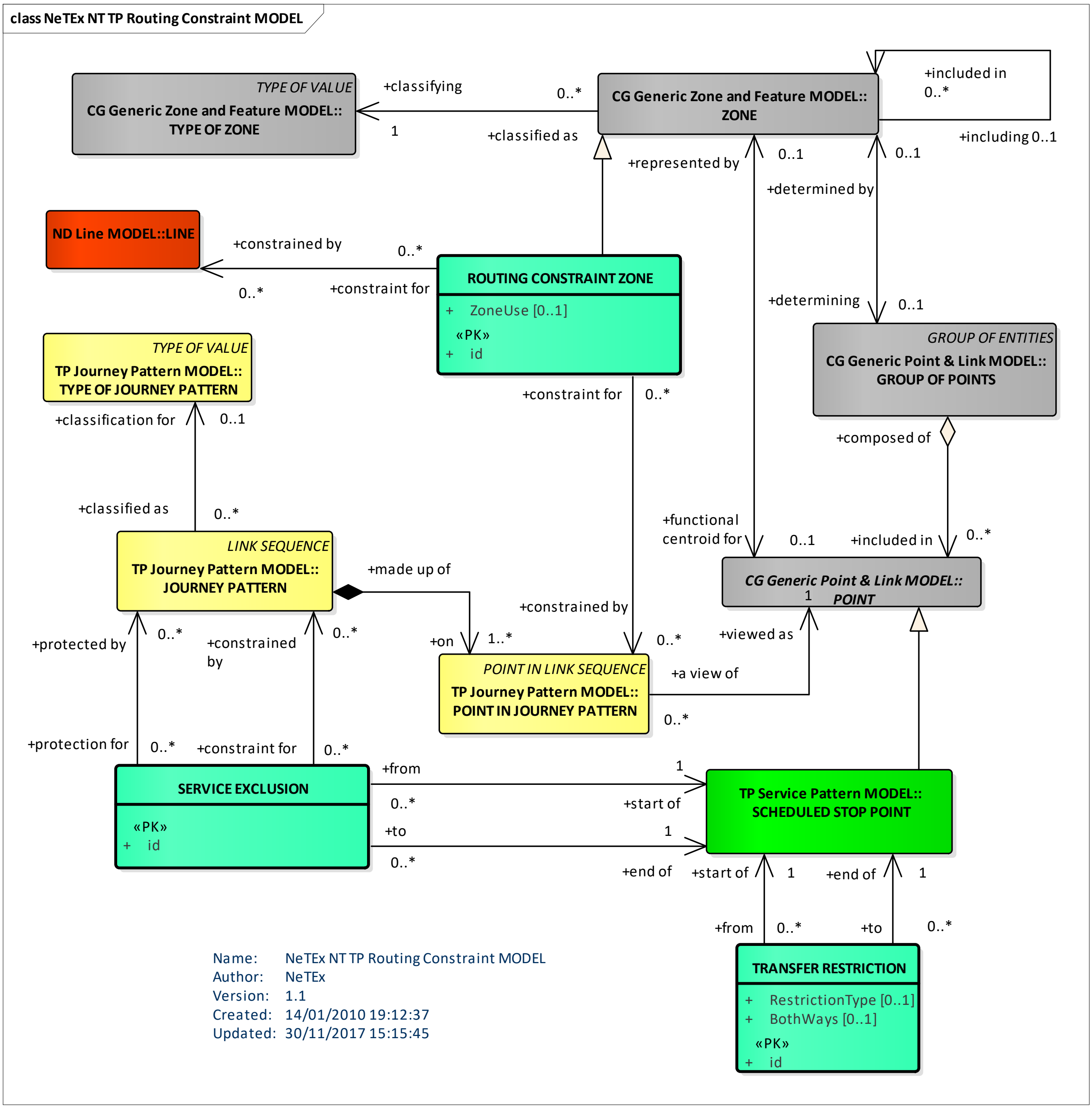


Figure 743 – Routing constraints – Conceptual MODEL (UML)

8.5.6.2 Routing Constraints – Examples

The following figure provides a typical example of ROUTING CONSTRAINT. A regional bus (Magenta Line) joins several towns. Towns B and C have their own local bus network and that don't want the Magenta Line to compete with these local bus lines. Therefore two ROUTING CONSTRAINT ZONEs are defined (one for town B and the other for town C). Inside these ZONEs, a passenger can't board and alight while staying inside the ZONE, so that the Magenta Line manage only the journey between towns, but not the one inside a town.

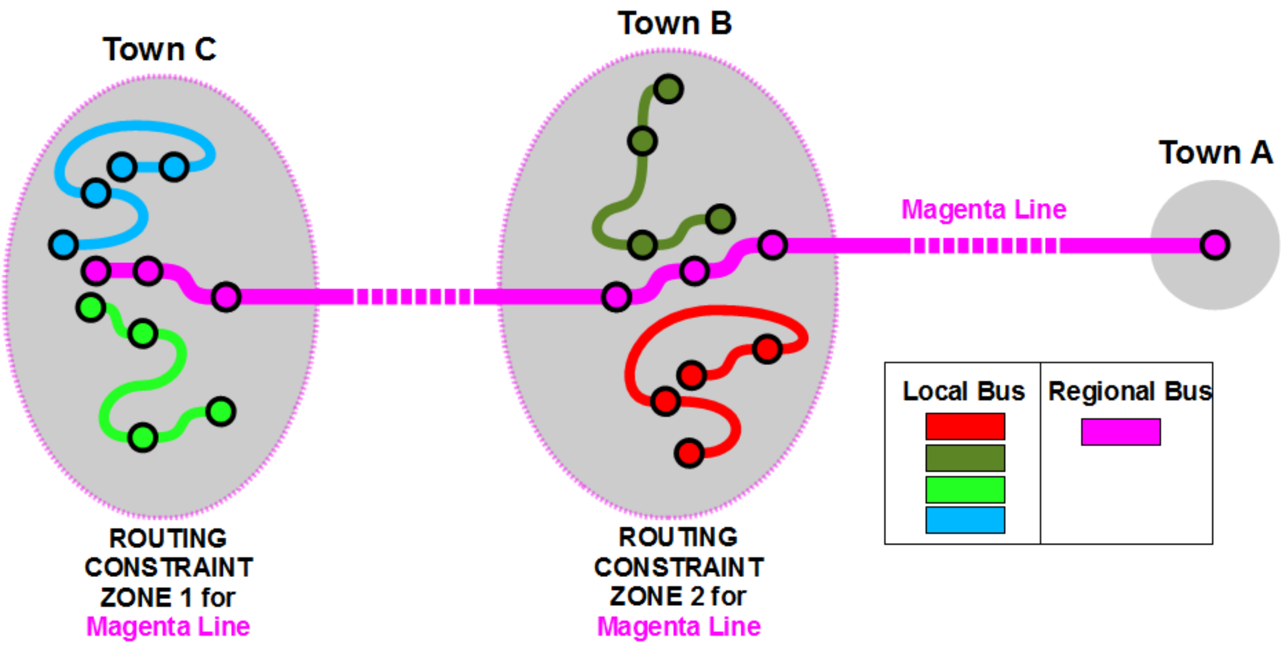


Figure 744 – Routing constraint example

8.5.6.3 Routing Constraints – Physical Model

The following figure shows detailed attributes of the ROUTING CONSTRAINT model.

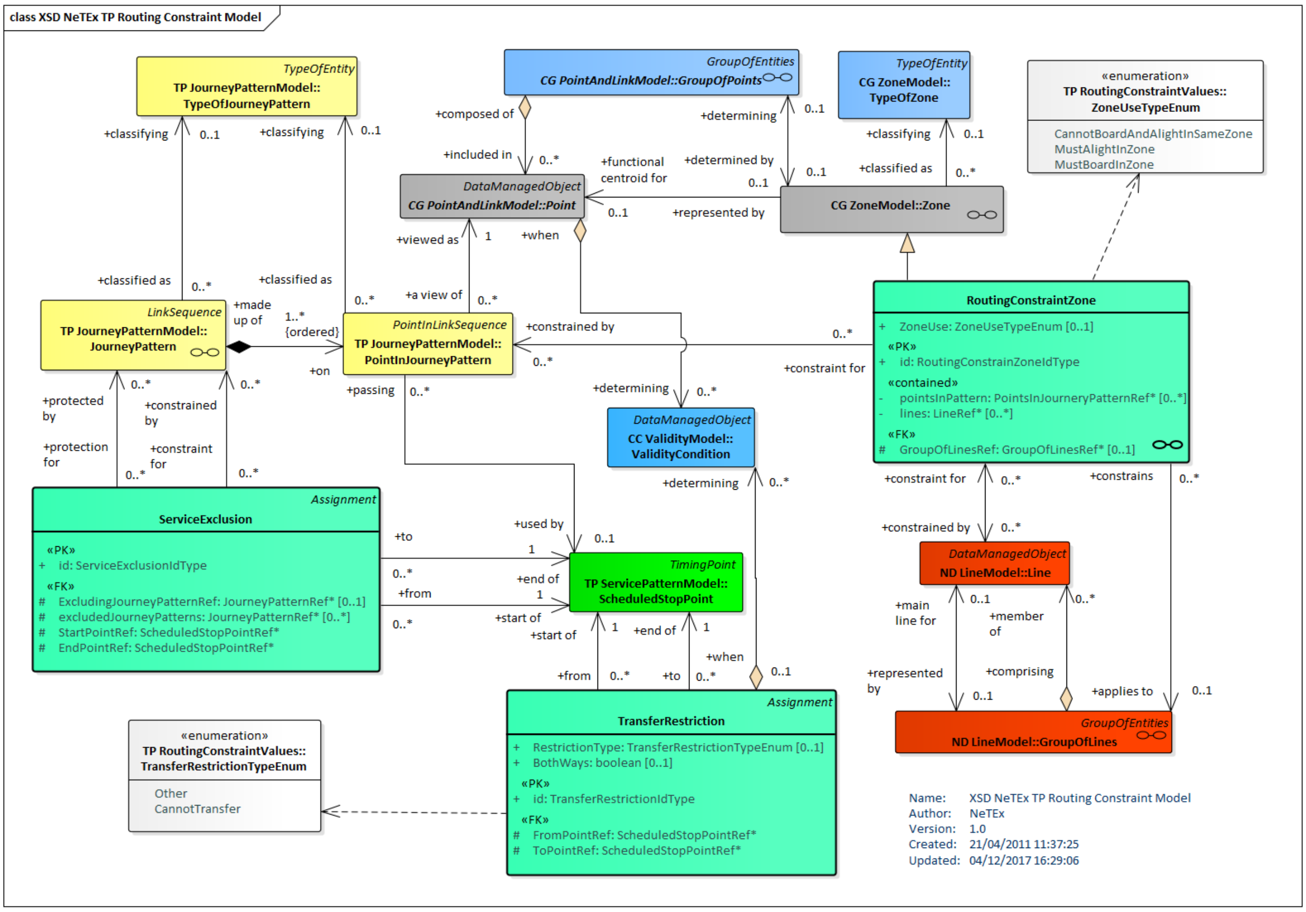


Figure 745 – Routing Constraints – Physical model (UML)

8.5.6.4 Routing Constraints– Attributes and XSD

8.5.6.4.1 RoutingConstraintZone – Model Element

ZONE defining a ROUTING CONSTRAINT. The ZONE may be defined by its contained SCHEDULED STOP POINTs or by its boundary points.

Some examples of routing constraints are:

- "If you board in this ZONE, you can't alight in the same ZONE"

- "If you board in this ZONE, you can alight only if the journey has passed through at least one STOP outside of the ZONE"

- "Journey will take place only if there are passengers on board"

- etc.

Table 630 – *RoutingConstraintZone* – Element

Classification	**Name**	**Type**	**Cardinality**	**Description**
::>	*::>*	*Zone*	*::>*	ROUTING CONSTRAINT ZONE inherits from ZONE.
«PK»	***id***	*RoutingConstrainZone-IdType*	1:1	Identifier of ROUTING CONSTRAINT ZONE.
«enum»	***ZoneUse***	*ZoneUseTypeEnum*	0:1	How zone may be used. See allowed values below.
«cntd»	***pointsInPattern***	*PointInJourneyPatternRef*	0:*	References to POINTs IN PATTERN associated with routing.
«cntd»	***lines***	*LineRef*	0:*	References to LINEs associated with routing.
«FK»	***GroupOfLinesRef***	*GroupOfLinesRef*	0:1	References to a GROUP OF LINEs associated with routing.

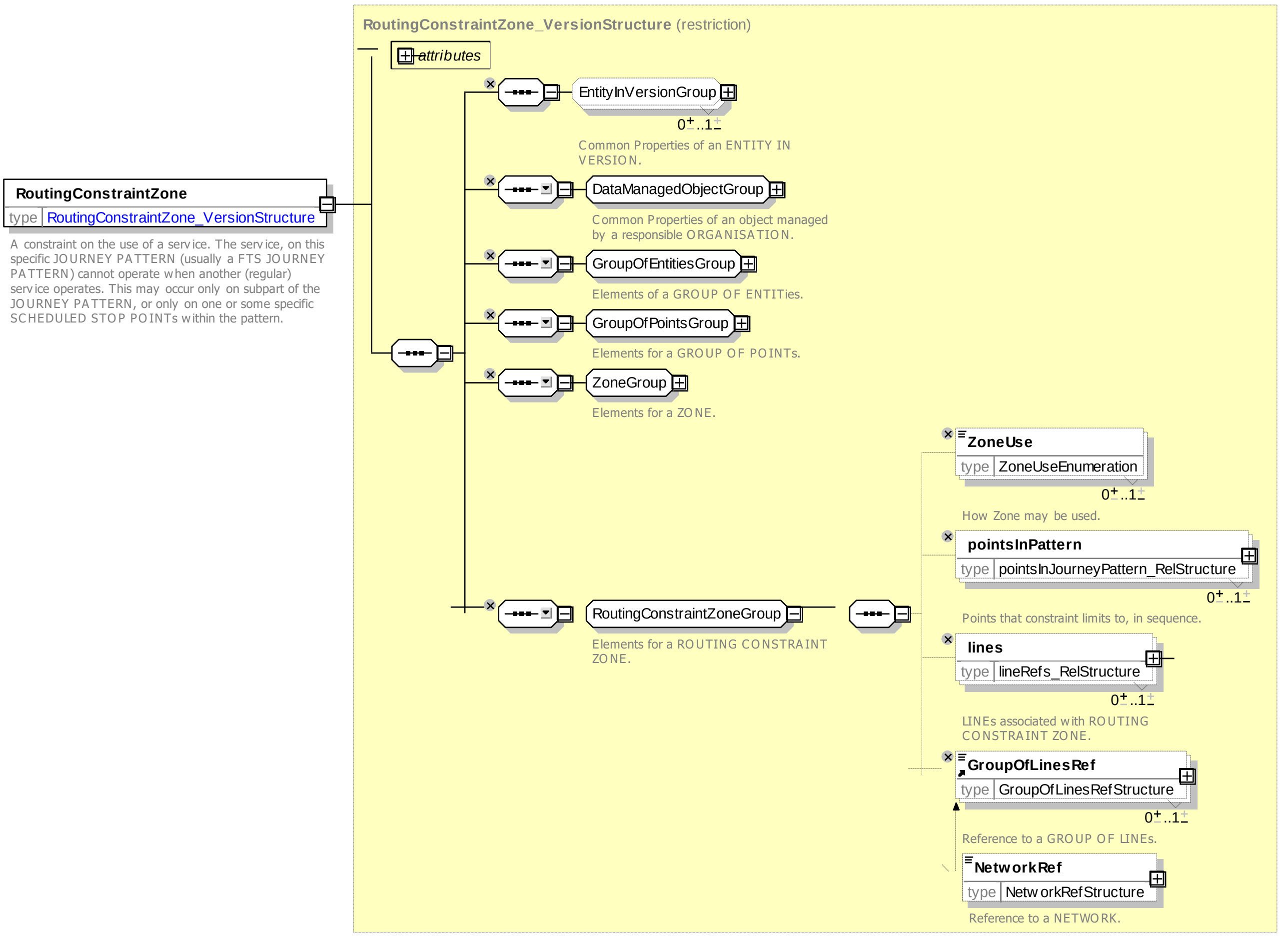


Figure 746 – *RoutingConstraintZone – XSD*

8.5.6.4.1.1 ZoneUseTypeEnum – – Allowed Values

Allowed values for ***ZoneUse*** (*ZoneUseTypeEnumeration*).

Table 631 – *ZoneUseType* – Allowed Values

Value	**Description**
cannotBoardAndAlightInSameZone	Cannot board and alight.
mustAlightInZone	Must alight in zone.
mustBoardInZone	Must board in Zone.
other	Other rule.

8.5.6.4.2 ServiceExclusion – Model Element

A constraint on the use of a service. The service, on this specific JOURNEY PATTERN (usually a FTS JOURNEY PATTERN) cannot operate when another (regular) service operates. This may occur only on subpart of the JOURNEY PATTERN, or only on one or some specific SCHEDULED STOP POINTs within the pattern.

Table 632 – *ServiceExclusion* – Element

Classification	**Name**	**Type**	**Cardinality**	**Description**
::>	*::>*	*Assignment*	*::>*	SERVICE EXCLUSION inherits from ASSIGNMENT.

«PK»	***id***	*ServiceExclusionIdType*	1:1	Identifier of SERVICE EXCLUSION.
«FK»	***ExcludingJourney PatternRef***	*JourneyPatternRef*	0:1	Excluding JOURNEY PATTERNs for the SERVICE EXCLUSION.
«FK»	***StartPointRef***	*ScheduledStopPointRef*	1:1	Starting SCHEDULED STOP POINT within JOURNEY PATTERN for exclusion.
«FK»	***EndPointRef***	*ScheduledStopPointRef*	1:1	Ending SCHEDULED STOP POINT within JOURNEY PATTERN for exclusion.
«cntd»	***excludedJourney Patterns***	*JourneyPatternRef*	0:*	Excluded JOURNEY PATTERNs for the SERVICE EXCLUSION.

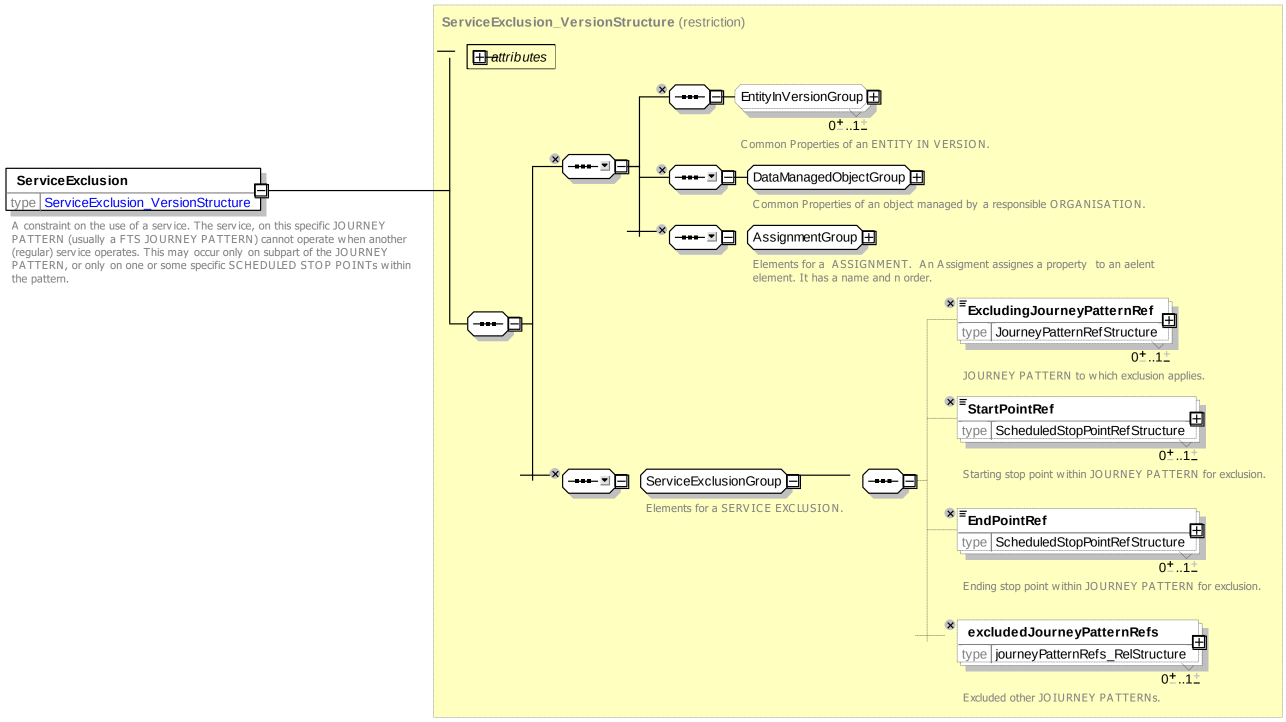


Figure 747 – *ServiceExclusion* – *XSD*

8.5.6.4.3 TransferRestriction – Model Element

CONSTRAINT that can be applied on a CONNECTION or INTERCHANGE between two SCHEDULED STOP POINTs, preventing or forbidding the passenger from making a transfer.

Table 633 – *TransferRestriction* – Element

Classification	Name	Type*Assignment*	Cardinality	Description
::>	::>	*Assignment*	::>	TRANSFER inherits from ASSIGNMENT.
«PK»	***id***	*TransferRestrictionIdType*	1:1	Identifier of TRANSFER RESTRICTION.

«FK»	***TypeOfTransfer-Ref***	*TypeOfTransferRef*	0:1	Classifcation of TRANSFER RESTRICTION.
«enum»	***RestrictionType***	*TransferRestriction-TypeEnum*	1:1	TYPE OF CONSTRAINT applying to restriction. See allowed values below.
	BothWays	*xsd:boolean*	0:1	Whether TRANSFER can be traversed in both directions.
«FK»	***FromPointRef***	*ScheduledStopPointRef*	1:1	From point of TRANSFER RESTRICTION.
«FK»	***ToPointRef***	*ScheduledStopPointRef*	1:1	To transfer point of TRANSFER RESTRICTION.

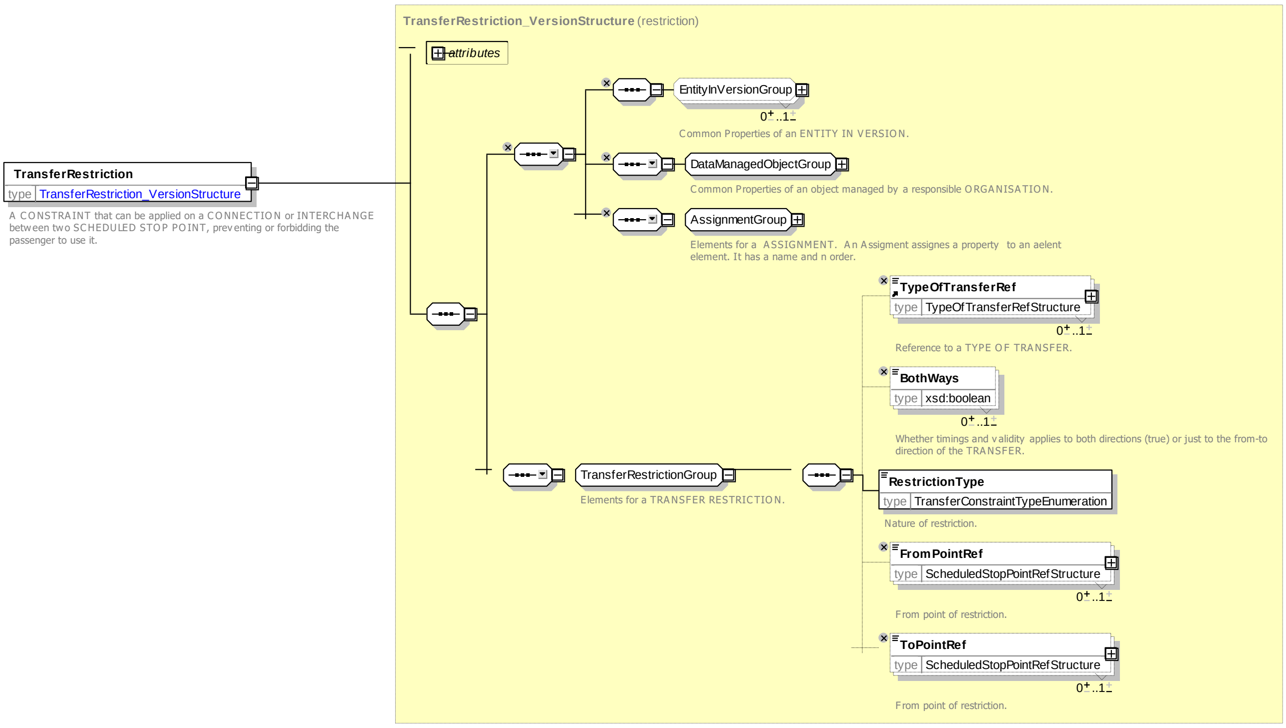


Figure 748 – *TransferRestriction – XSD*

8.5.6.4.3.1 TransferRestrictionType – Allowed Values

Allowed values for ***TransferRestriction*** (*TransferRestrictionTypeEnumeration*).

Table 634 – *TransferRestrictionType* – Allowed Values

Value	**Description**
canTransfer	Can make a transfer.
cannotTransfer	Cannot make a transfer.
other	Other restriction.

8.5.7 Time Demand Type

8.5.7.1 Introduction

The transport offer of a public transport company is provided to match different levels of demand at different times. The demand, as well as the traffic environment (such as traffic volume and flow speed) tends to vary during the course of the day, but with repeating characteristics each day. To handle these variations, public transport companies usually differentiate between dense traffic hours (such as morning and afternoon peak hours) and low traffic hours. Planning a detailed service is often based on pre-ordered service levels and parameters in these intervals.

8.5.7.2 TIME DEMAND TYPE – Conceptual MODEL

A TIME DEMAND TYPE represents the entity which encapsulates differentiation between time intervals with own demand and traffic flow characteristics (such as peak hours and off-peak hours). All time-related values – especially run times and layover times – depend on which TIME DEMAND TYPE they are used for.

To represent the function, a TIME DEMAND TYPE ASSIGNMENT relates GROUP OF TIMING LINKs to TIME DEMAND TYPEs.

A TIME DEMAND TYPE also refers to a TIME BAND in which the aforementioned characteristics apply.

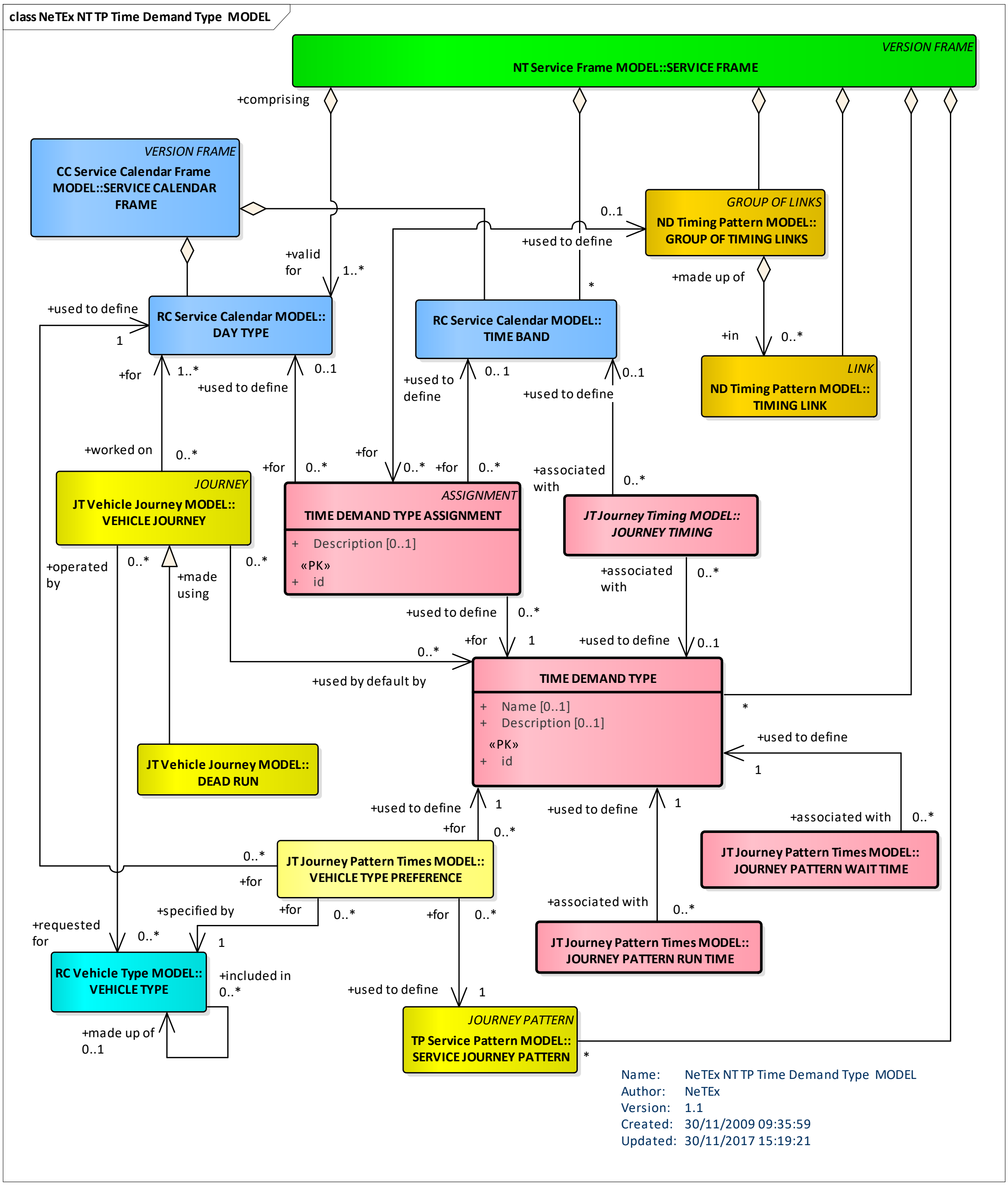


Figure 749 – Time Demand Type – Conceptual MODEL (UML)

8.5.7.3 Time Demand Type – Physical Model

Based on the conceptual model, the Time Demand Type physical model defines the implementation units. A **TimeDemandType** is assigned to a **GroupOfTimingLinks** via a **TimeDemandTypeAssigment** object.

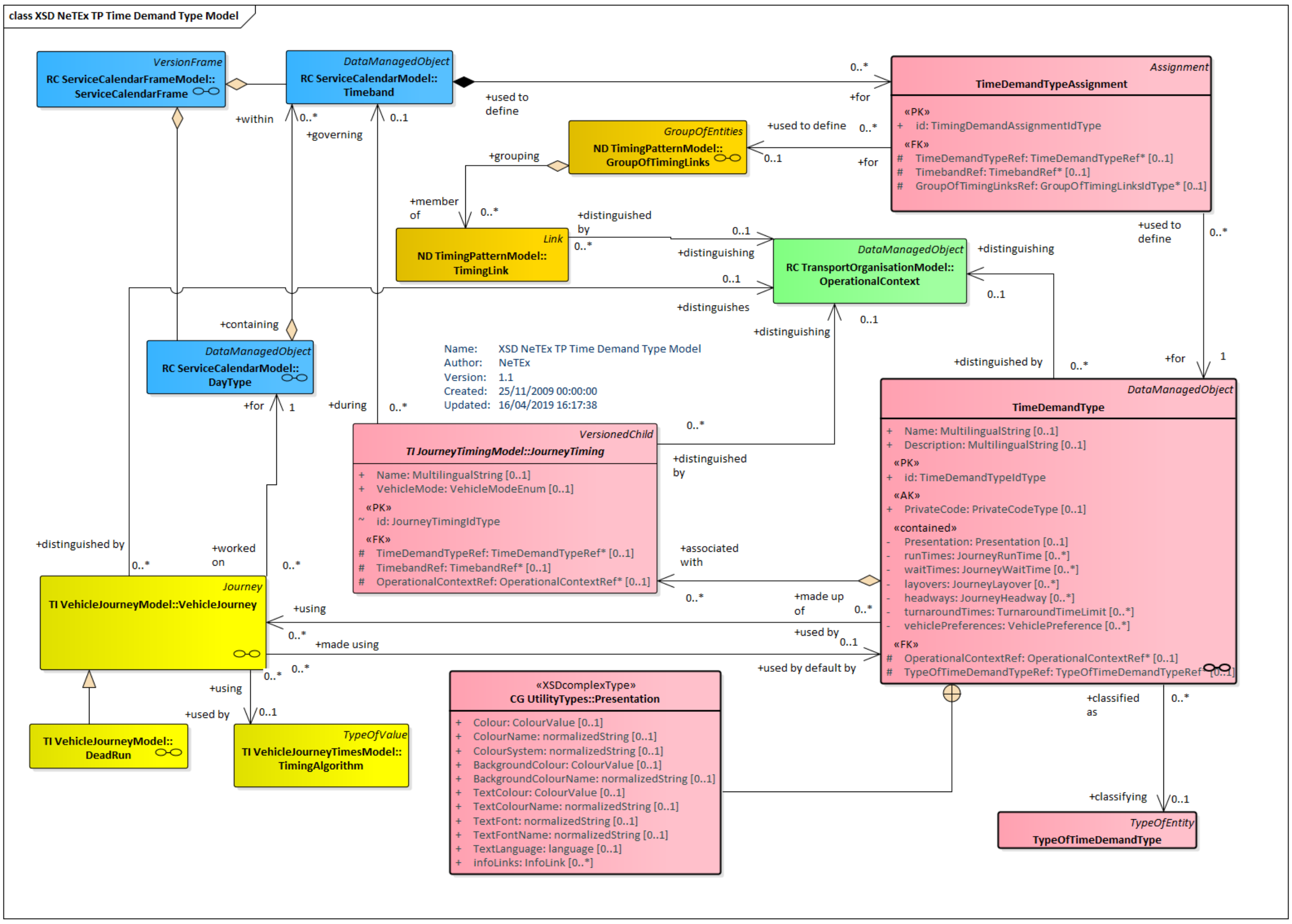


Figure 750 – Time Demand Type – Physical Model (UML)

8.5.7.4 Time Demand Type – Attributes and XSD

8.5.7.4.1 TimeDemandType – Model Element

An indicator of traffic conditions or other factors which may affect vehicle run or wait times. It may be entered directly by the scheduler or defined by the use of TIME BANDs.

Table 635 – *TimeDemandType* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	TIME DEMAND TYPE inherits from DATA MANAGED OBJECT.
«PK»	***id***	*TimeDemandTypeIdType*	1:1	Identifier of TIME DEMAND TYPE.
	Name	*MultilingualString*	0:1	Name of TIME DEMAND TYPE.
	Description	*MultilingualString*	0:1	Description of TIME DEMAND TYPE.
«AK»	***PrivateCode***	*xsd:normalizedString*	0:1	Alternative key for TIME DEMAND TYPE.

«FK»	***TypeOfTime-DemandTypeRef***	*TypeOfTimeDemandType-Ref*	0:1	Reference to a TYPE OF TIME DEMAND TYPE.
«cntd»	***Presentation***	*Presentation*	0:1	Preferred presentation attributes to use when rendering LINE on maps, etc.
«FK»	***Operational-ContextRef***	*OperationalContextRef*	0:1	OPERATIONAL CONTEXT associated with TIME DEMAND TYPE.
XGRP	***TimeDemand-TimingsGroup***	***xmlGroup***	0:*	Timing elements for TIME DEMAND TYPE.

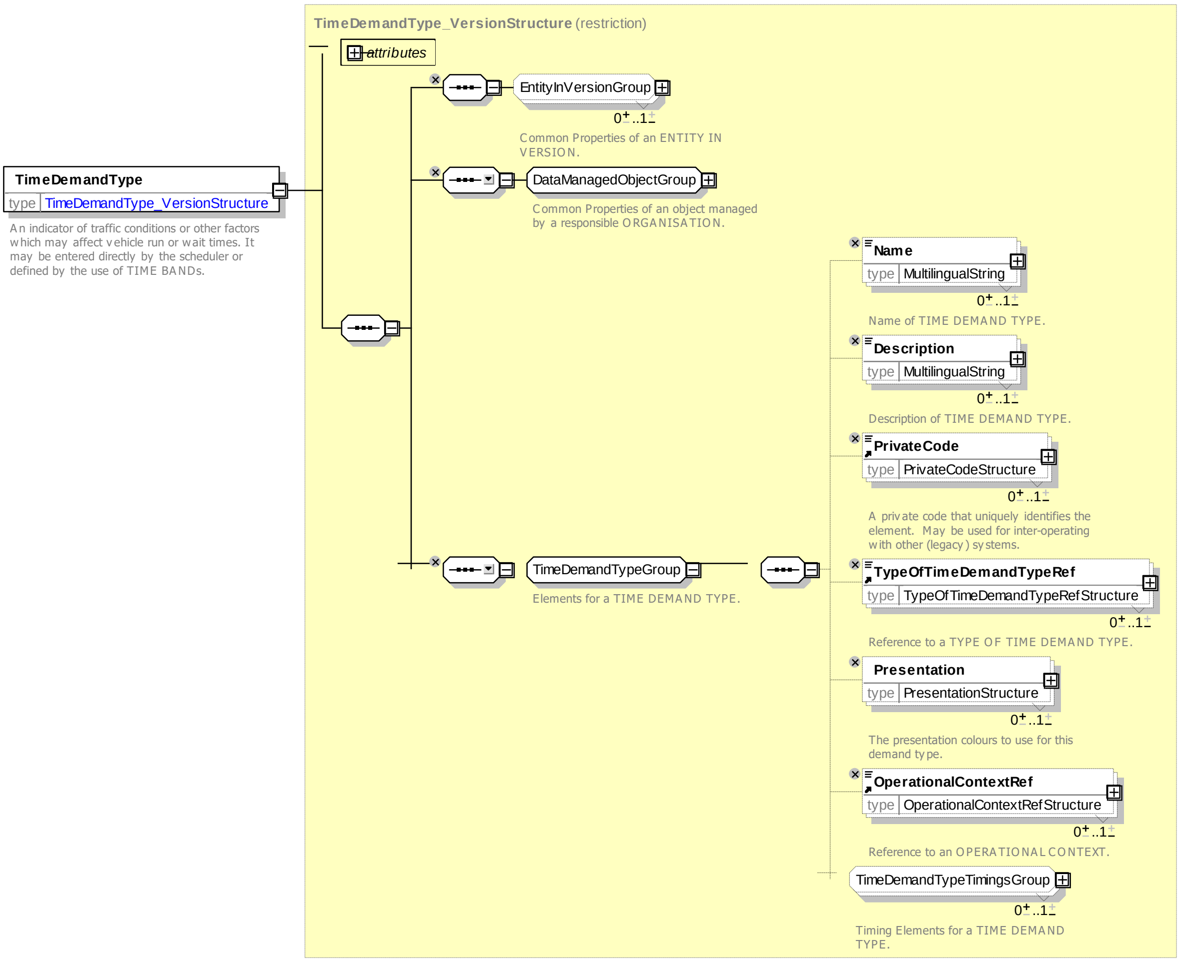


Figure 751 – *TimeDemandType – XSD*

8.5.7.4.1.1 TimeDemandTimingsGroup – XML Group

Timing elements for a TIME DEMAND TYPE.

Table 636 – *TimeDemandTimingsGroup* – Group

Classification	Name	Type	Cardinality	Description
«cntd»	***runTimes***	*JourneyRunTime*	0:*	RUN TIMEs associated with TIME DEMAND TYPE.

«cntd»	***waitTimes***	*JourneyWaitTime*	0:*	WAIT TIMEs associated with TIME DEMAND TYPE.
«cntd»	***layovers***	*JourneyLayover*	0:*	LAYOVERs associated with TIME DEMAND TYPE.
«cntd»	***headways***	*JourneyHeadway*	0:*	FREQUENCies associated with TIME DEMAND TYPE.
«cntd»	***vehicle-Preferences***	*VehiclePreference*	0:*	VEHICLE PREFERENCEs associated with TIME DEMAND TYPE.

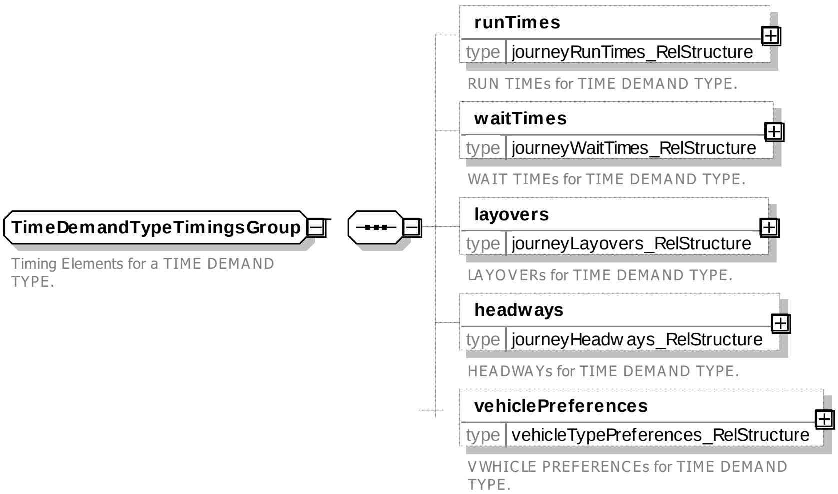


Figure 752 – *TimeDemandTimingsGroup – XSD*

8.5.7.4.2 TimeDemandTypeAssignment – Model Element

The assignment of a TIME DEMAND TYPE to a TIME BAND depending on the DAY TYPE and GROUP OF TIMING LINKs.

Table 637 – *TimeDemandTypeAssignment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Assignment*	*::>*	TIME DEMAND TYPE ASSIGNMENT inherits from ASSIGNMENT.
«PK»	***id***	*TimingDemandAssignmentIdType*	1:1	Identifier of TIME DEMAND TYPE ASSIGNMENT.
	At least one	CHOICE	1:1	
«FK»	a ***TimeDemandType-Ref***	*TimeDemandTypeRef*	0:1	Reference to a TIME DEMAND TYPE associated with of TIME DEMAND TYPE ASSIGNMENT.
«FK»	b ***TimeBandRef***	*TimeBandRef*	0:1	Reference to a TIME BAND associated with of TIME DEMAND TYPE ASSIGNMENT.
«FK»	***GroupOfTimingLinksRef***	*GroupOfTimingLinksRef*	0:1	Group of TIMING LINKs associated with of TIME DEMAND TYPE ASSIGNMENT.

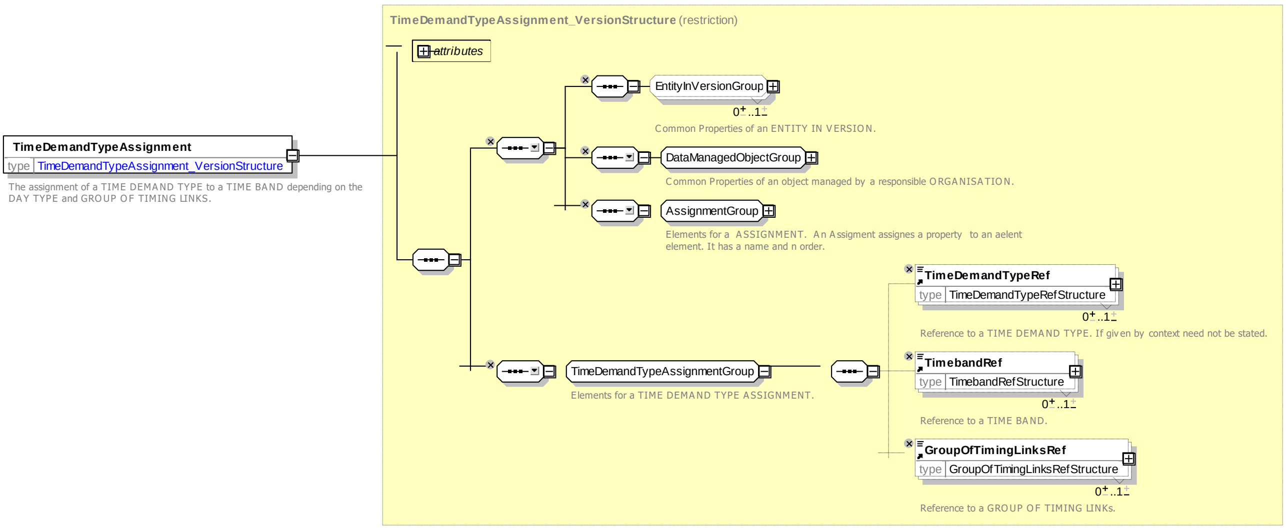


Figure 753 – *TimeDemandTypeAssignment – XSD*

8.5.7.4.3 TypeOfTimeDemandType – Model Element

A classification of TIME DEMAND TYPEs according to their functional purpose.

Table 638 — *TypeOfTimeDemandType – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*TypeOfEntity*	::>	TYPE OF TIME DEMAND TYPE inherits from TYPE OF ENTITY.
«PK»	***id***	*TypeOfTimeDemandTypeIdType*	1:1	Identifier of TYPE OF TIME DEMAND TYPE.

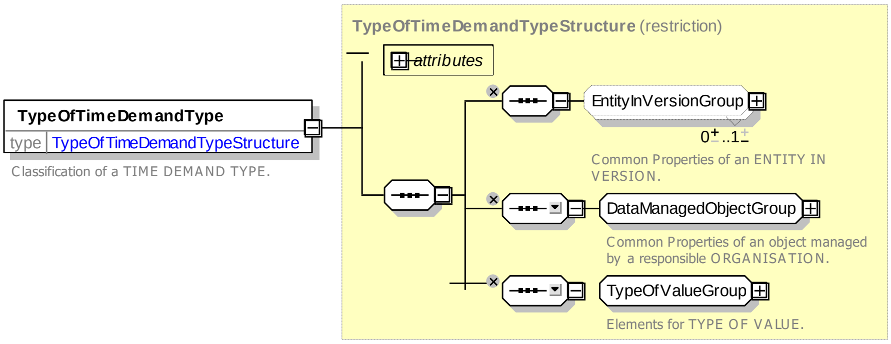


Figure 754 — *TypeOfTimeDemandType– XSD*

8.5.7.4.4 JourneyTiming – Model Element

An element such as a journey timing whose value depends on the time of use and so can be associated with a TIME DEMAND TYPE or TIME BAND. This is an abstract element that is specialised to create different types of timing.

Table 639 – *JourneyDemand* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*VersionedChild*	*::>*	JOURNEY TIMING inherits from VERSIONED CHILD
«PK»	***id***	*JourneyDemandIdType*	1:1	Identifier of JOURNEY TIMING.
	Description	*MultilingualString*	0:1	Description of JOURNEY TIMING.
«enum»	***Mode***	*VehicleModeEnum*	0:1	TRANSPORT MODE of JOURNEY TIMING. See TRANSPORT MODE for allowed values.
«FK»	***TimeDemandType-Ref***	*TimeDemandTypeRef*	0:1	TIME DEMAND TYPE associated with JOURNEY TIMING.
«FK»	***TimebandRef***	*TimebandRef*	0:1	Reference to a TIME BAND.
«FK»	***OperationalContext Ref***	*OperationalContextRef*	0:1	OPERATIONAL CONTEXT associated with JOURNEY TIMING.

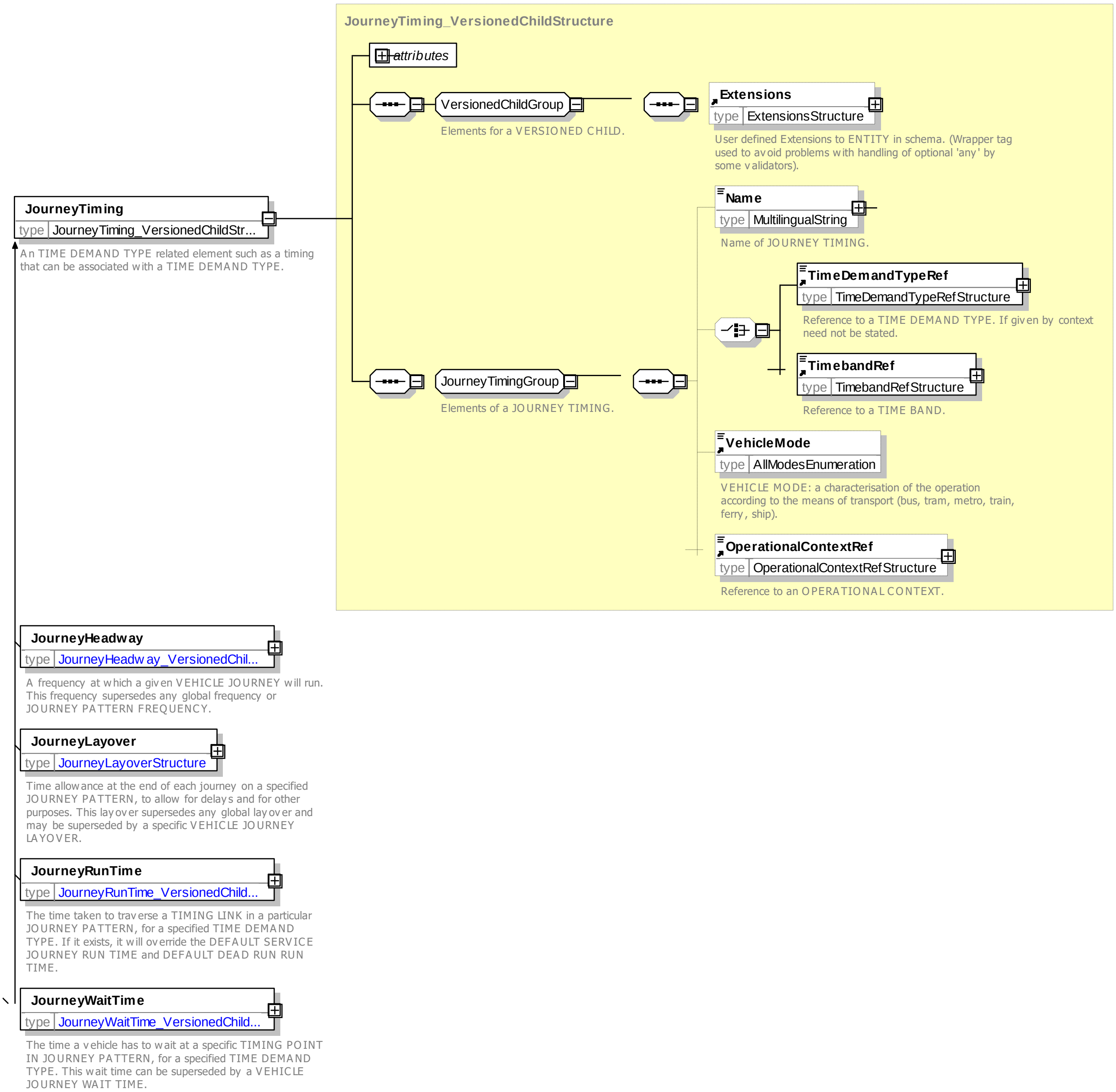


Figure 755 – *JourneyTIming – XSD*

8.5.7.5 XML Example of Time Demand Type

The following XML code fragment shows definitions for two TIME DEMAND TYPEs.

EXAMPLE

```
<TimeDemandType version="any" id="hde:TDT0001">
    <Name>Peak Time demand </Name>
</TimeDemandType>
<TimeDemandType version="any" id="hde:TDT0002">
    <Name>Time demand off Peak</Name>
</TimeDemandType>
```

8.5.8 Passenger Stop Assignment

8.5.8.1 Introduction

For the scheduling process, but also in the passenger information domain, approximate locations for boarding/alighting are determined in order either to plan timetables or to inform passenger about the availability of public transport at some specific places (e.g. points of interest, specific addresses, etc.). Frequently, these locations represent the centroid of a cluster of points, close to each other, where public transport vehicles stop and where the users have the opportunity to board/alight. These "points" in space are, geometrically speaking, areas. In the case of trains or metro, such locations may be quays with several boarding positions. However, for advertisement or planning purposes, only one single location is taken into account: the term "logical stop" is often used to describe this situation. The Reference Data Model for Public Transport (EN12896) uses in this context the concept of (SCHEDULED) STOP POINT.

In reality, i.e. physically, such a SCHEDULED STOP POINT may correspond either to a whole STOP PLACE or to a QUAY, or to a BOARDING POSITION. Conversely a STOP PLACE, QUAY or BOARDING POSITION may be viewed in different functional contexts and be subject to several assignments to SCHEDULED STOP POINTs.

8.5.8.1.1 Further Discussion

Historically there has been some confusion in transport models between *a stop as identified in the timetable* (a logical construct, for example, that a timetabled service going in a particular direction stops at a station at a particular time regardless of platform); *a stop as a physical point* (i.e. an actual pole beside the road, or platform within a station), and the *stop as a point on a line regardless of a timetable or direction (*for example, a rail or metro station, or a pair of physical stops either side of a road on a bus route that are depicted as being a single "stop" on a route map).

The following figures attempt to further convey this distinction by showing the same three stops (a) as points on a line in a typical schematic map of a line; (b) as the stopping points of some journeys of a timetable; and (c) as physical points at which the vehicles may visit.

Figure 756 – Stops as places on a line – Example

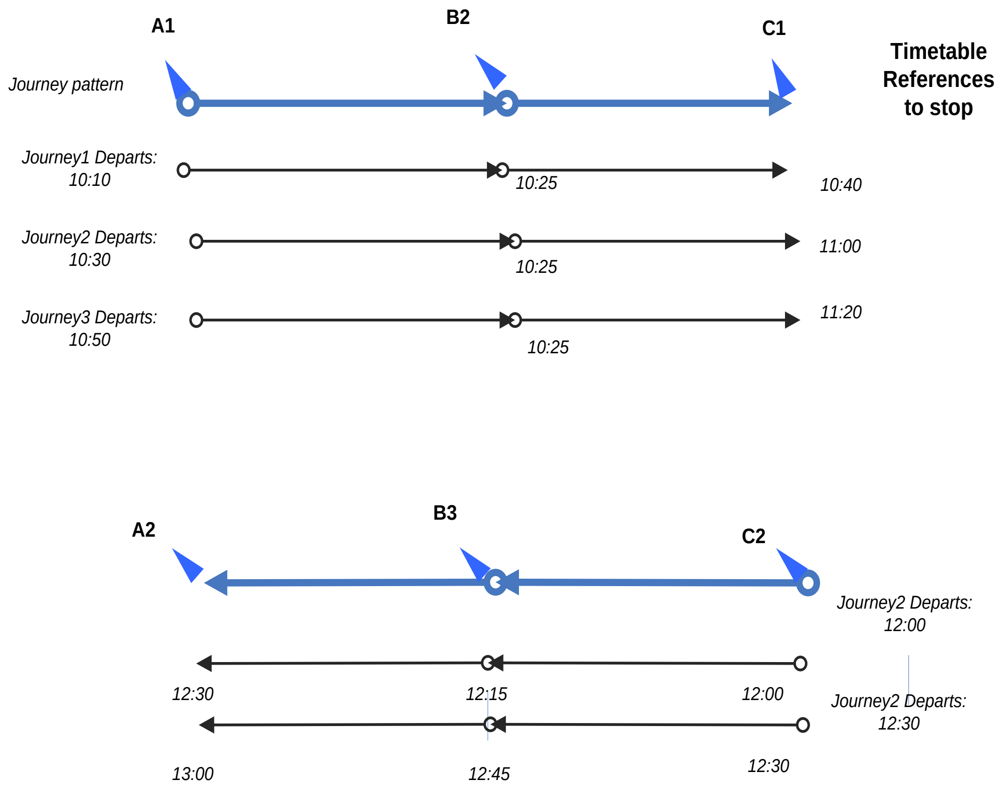


Figure 757 – Stops as stopping points in the timetable – Example

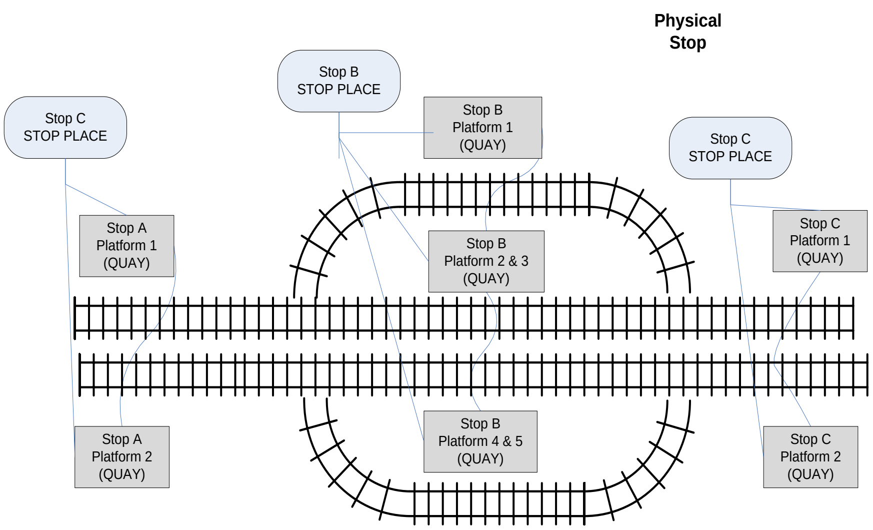


Figure 758 – Stops as physical places – Example

NeTEx clarifies these various possible relationships. It represents the logical stop in the timetable as a distinct concept, the SCHEDULED STOP POINT, independent of its physical nature. It separately represents the physical point of access as a QUAY, i.e. platform or pole, ferry landing or airline gate. It adds a STOP PLACE as a named representation of a physical interchange that may group QUAYs – for example, a station, or a pair of bus stops on a street with the same name. Then to associate a timetable or real-time data for a particular service with a physical stop, *NeTEx* uses the concept of a STOP ASSIGNMENT, which associates a SCHEDULED STOP POINT with a STOP PLACE. An assignment can be just to the whole station (STOP

PLACE), or to a specific platform (QUAY) within the station (thus allowing for detailed platform allocation and also platform changes).

In the trivial case where the SCHEDULED STOP POINT has the same identifier as the STOP PLACE or QUAY, the assignment can be implicit (i.e. because they have been given the same codes, the association between the SCHEDULED STOP POINT and the QUAY or STOP PLACE can be inferred). In other cases, where the code is different an explicit assignment needs to be used.

There can potentially be multiple assignments of the same STOP PLACE. A condition quite often found in the real world is that different operators or different modes use different codes for either the same SCHEDULED STOP POINT, or STOP PLACE and QUAY, or both: the NeTEx model allows this to be represented.

8.5.8.2 PASSENGER STOP ASSIGNMENT – Conceptual MODEL

SCHEDULED STOP POINTs are related to the physical stop through the concept of a PASSENGER STOP ASSIGNMENT, defined as the allocation of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) to a specific STOP PLACE, a QUAY or a BOARDING POSITION. For the reasons explained above, there may be several such assignments for a STOP PLACE, a QUAY or a BOARDING POSITION.

This assignment may be subject to AVAILABILITY CONDITIONs.

In general a fixed assignment of a SCHEDULED STOP POINT to an exact STOP PLACE, QUAY or BOARDING POSITION will be planned but may sometimes change dynamically. Often such dynamic assignments represent a change of platform and are a consequence of a control action of an AVM system.

A DYNAMIC STOP POINT ASSIGNMENT is used to represent this situation. It may indicate also a scheduled set of STOP POINTs from which the real-time allocation may be made.

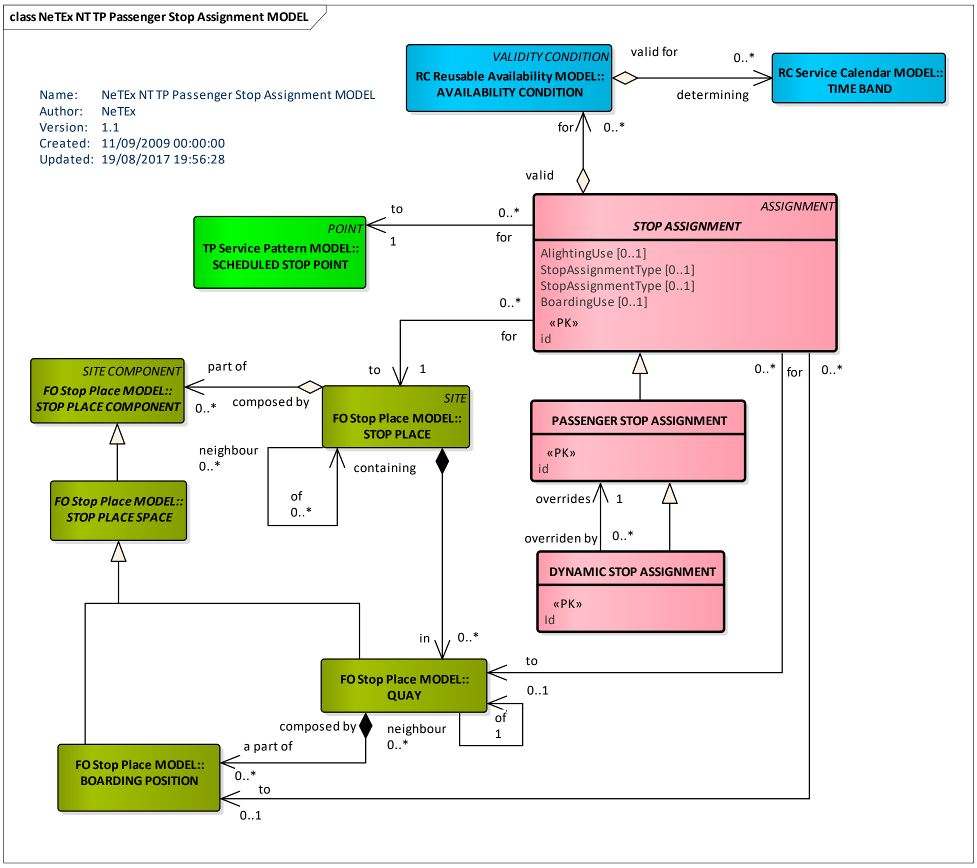


Figure 759 – Passenger Stop Assignment - Conceptual MODEL (UML)

8.5.8.2.1 Example of a Stop Assignment

The following figure shows some of the PASSENGER STOP ASSIGNMENTs for Wimbledon station, which has four different modes (Rail, Metro, Tram and Bus). The Tram Link and bus stop SCHEDULED STOP POINTs correspond to specific QUAYs. The Rail SCHEDULED STOP POINTs correspond to the STATION as a whole (but could potential be assigned in more detail to a specific platform, i.e. QUAY within the station. The Tram shares a specific rail platform. The metro SCHEDULED STOP POINT alternates between either platform.

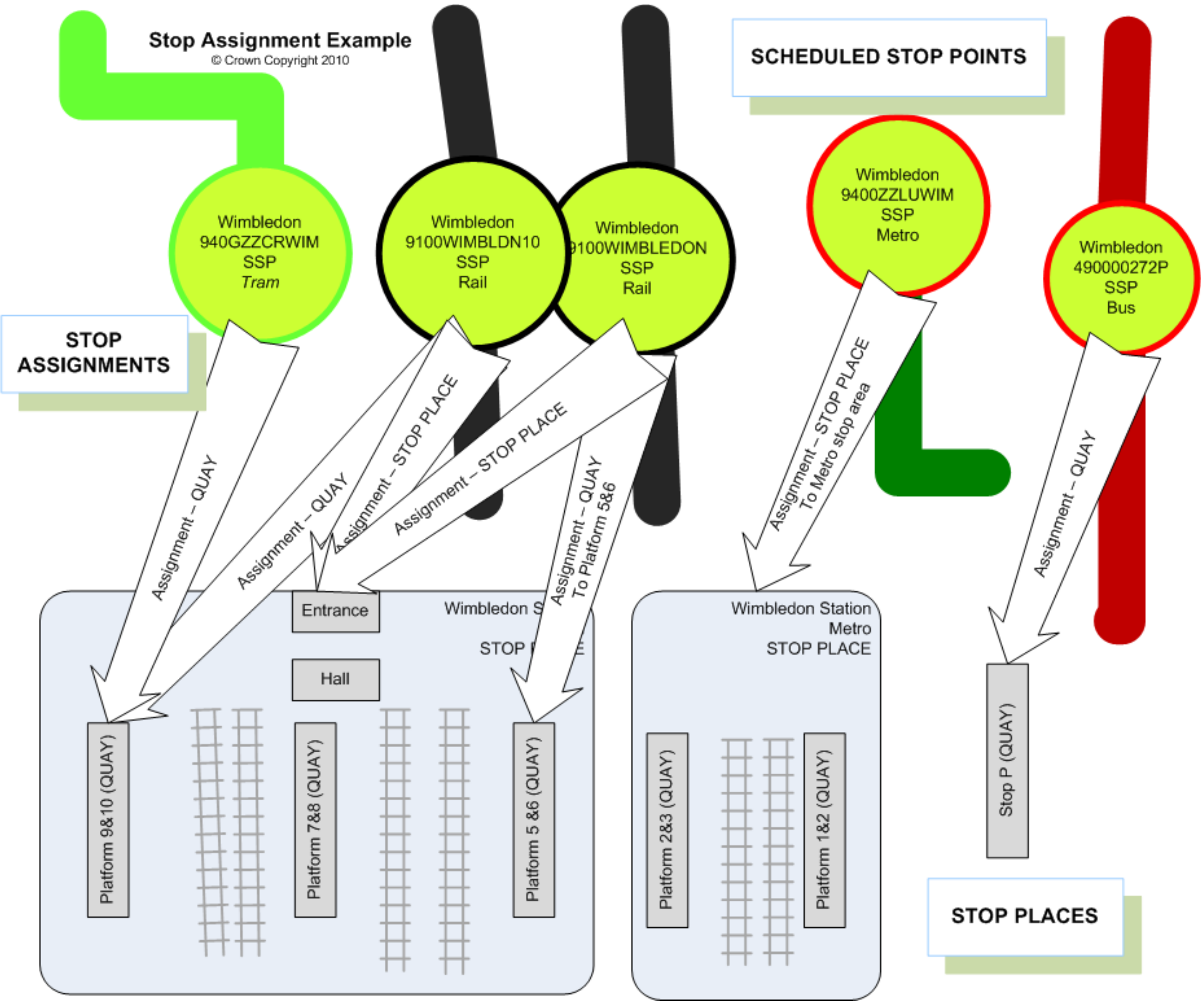


Figure 760 – Some Stop Assignments for the Wimbledon – Example

8.5.8.3 Passenger Stop Assignment – Physical Model

The following figure shows the attributes of the PASSENGER STOP ASSIGNMENT model.

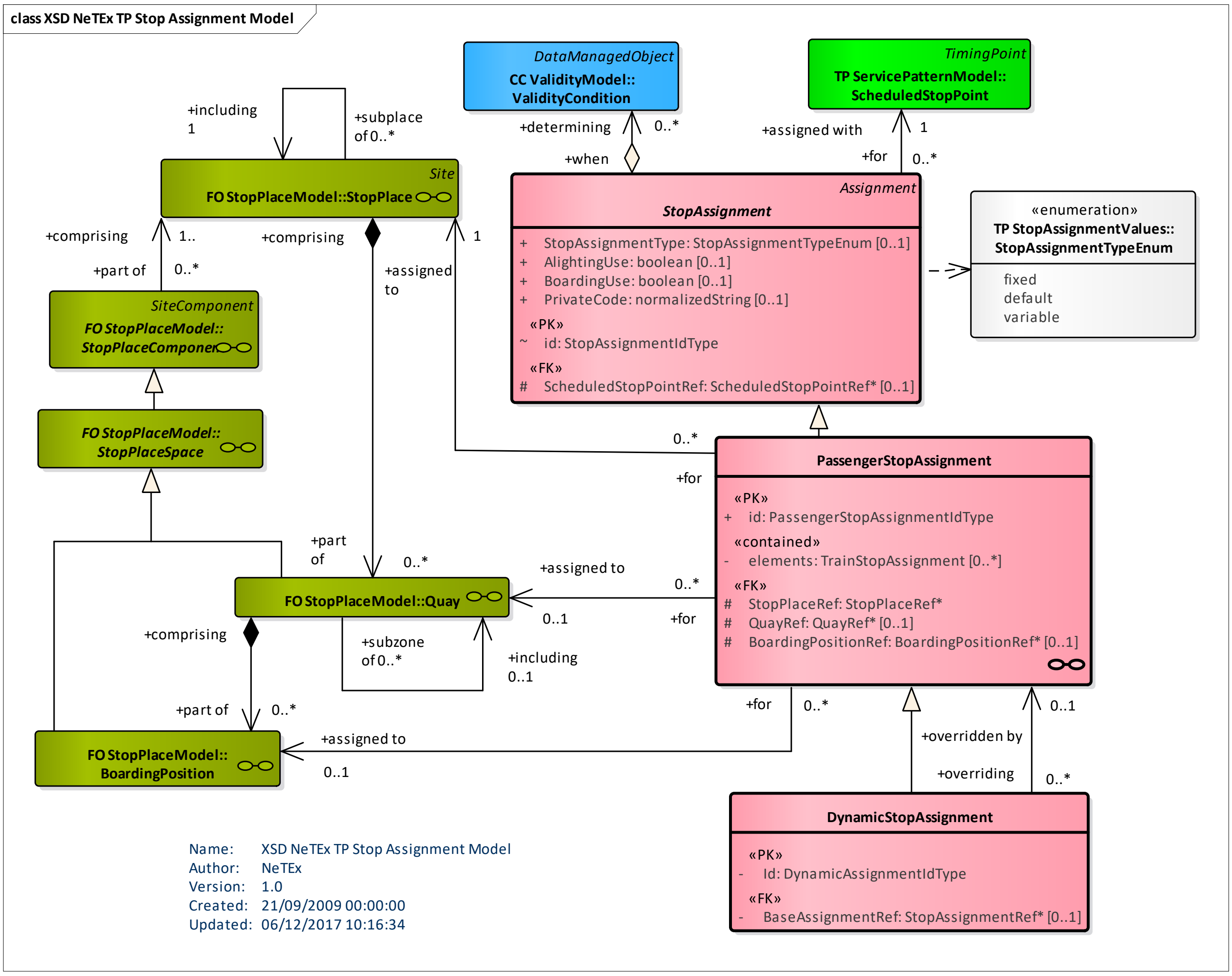


Figure 761 – Passenger Stop Assignment – Physical Model (UML) – Detail

8.5.8.4 Stop Assignment – Attributes and XSD

8.5.8.4.1 StopAssignment – Model Element

The allocation of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) to a specific STOP PLACE, for either a Passenger JOURNEY or VEHICLE SERVICE.

Table 640 — *StopAssignment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	<u>*Assignment*</u>	::>	STOP ASSIGNMENT inherits from ASSIGNMENT.
«PK»	***id***	*StopAssignmentIdType*	1:1	Identifier of a STOP ASSIGNMENT.
	BoardingUse	*xsd:boolean*	0:1	Whether STOP ASSIGNMENT allows boarding at the stop.

	AlightingUse		*xsd:boolean*	0:1	Whether STOP ASSIGNMENT allows alighting at the stop.
	PrivateCode		*PrivateCode*	0:1	Private code for STOP ASSIGNMENT.
			CHOICE	0:1	Can be reference or inline.
«FK»	***a***	***ScheduledStop PointRef***	*ScheduledStopPointRef*	0:1	Reference to a SCHEDULED STOP POINT to be assigned.
«FV»	***b***	***ScheduledStop Point***	*ScheduledStopPoint*	0:1	SCHEDULED STOP POINT to be assigned. +v1.1

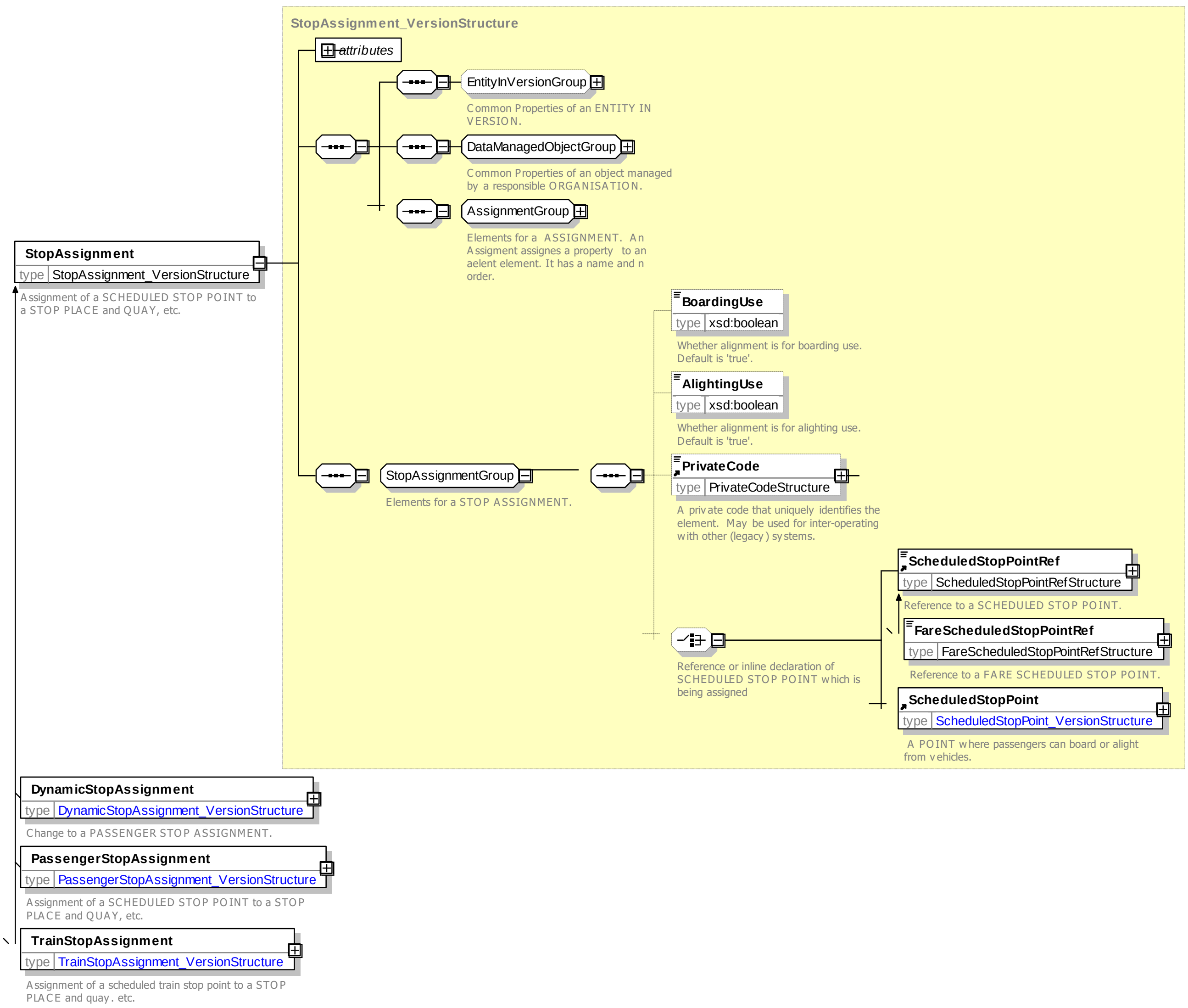


Figure 762 –*StopAssignment – XSD*

8.5.8.4.2 PassengerStopAssignment – Model Element

The allocation of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) to a specific STOP PLACE for a PASSENGER SERVICE, and also possibly a QUAY and BOARDING POSITION.

Table 641 — *PassengerStopAssignment – Element*

Classification	Name		Type	Cardinality	Description
::>	*::>*		*StopAssignment*	*::>*	PASSENGER STOP ASSIGNMENT inherits from STOP ASSIGNMENT.
«PK»	***id***		*PassengerStopAssignment-IdType*	1:1	Identifier of PASSENGER STOP ASSIGNMENT.
			CHOICE	0:1	Can be reference or inline.
«FK»	***a***	***StopPlaceRef***	*StopPlaceRef*	1:1	Reference to STOP PLACE to which the SCHEDULED STOP POINT is assigned.
«FV»	***b***	***StopPlace***	*StopPlace*	1:1	STOP PLACE to which the SCHEDULED STOP POINT is assigned.
			CHOICE	0:1	Can be reference or inline.
«FK»	***a***	***QuayRef***	*QuayRef*	0:1	Reference to a QUAY within the STOP PLACE to which the SCHEDULED STOP POINT is assigned.
«FV»	***b***	***Quay***	*Quay*	0:1	QUAY within the STOP PLACE to which the SCHEDULED STOP POINT is assigned.
			CHOICE	0:1	Can be reference or inline.
«FK»	***a***	***Boarding-PositionRef***	*BoardingPositionRef*	0:1	Reference to a BOARDING POSITION with QUAY at STOP PLACE to which the SCHEDULED STOP POINT is assigned.
«FV»	***b***	***Boarding-Position***	*BoardingPosition*	0:1	BOARDING POSITION with QUAY at STOP PLACE to which the SCHEDULED STOP POINT is assigned.
«cntd»	***trainElements***		*TrainStopAssignment*	0:*	TRAIN STOP ASSIGNMENTs associated with PASSENGER STOP ASSIGNMENT. These can provide more detailed information about individual BOARDING POSITION and carriage alignments.

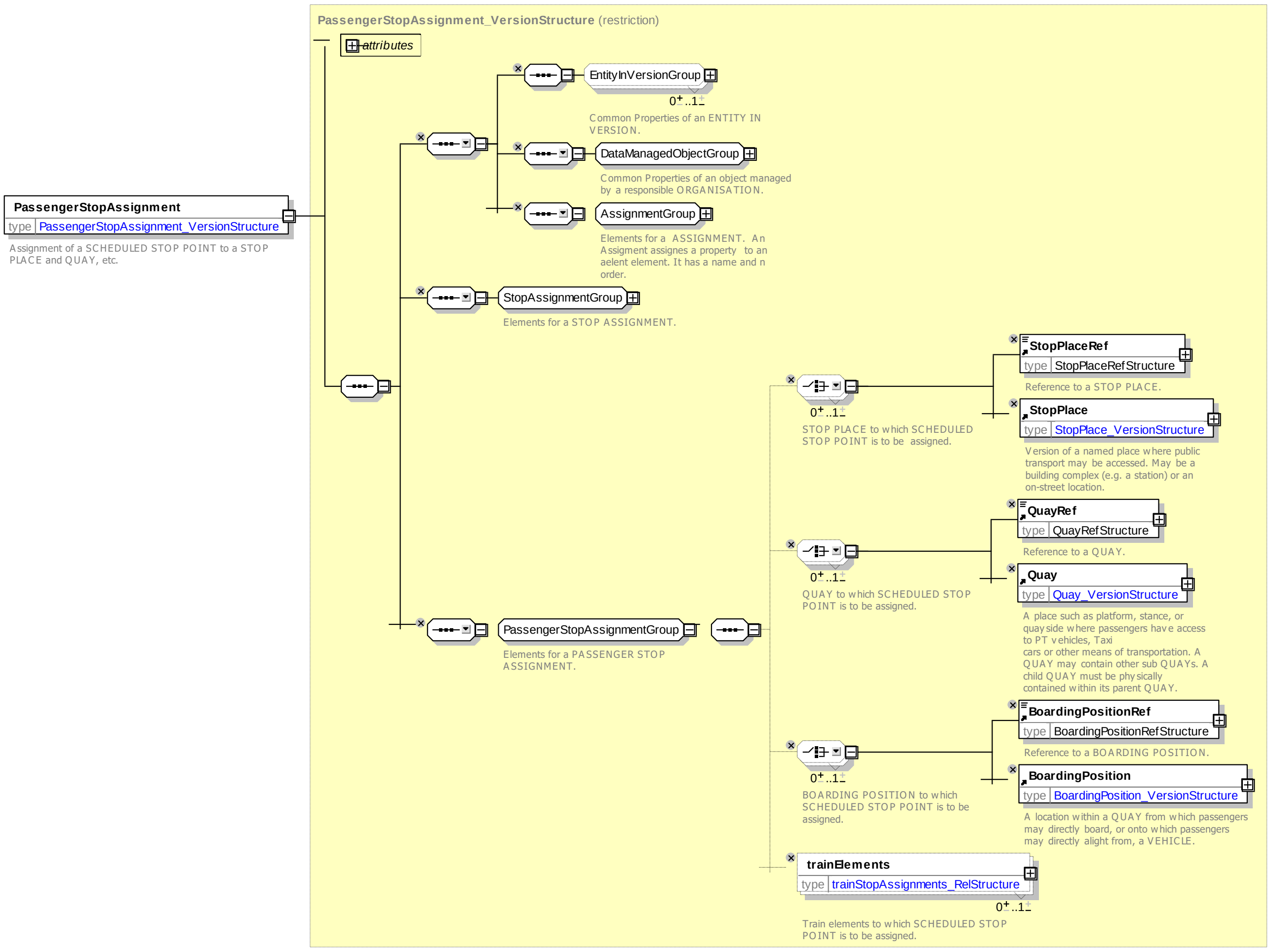


Figure 763 – *PassengerStopAssignment – XSD*

8.5.8.4.3 DynamicStopAssignment – Model Element

The dynamic association of a SCHEDULED STOP POINT (i.e. a STOP POINT of a SERVICE PATTERN or JOURNEY PATTERN) with the next available STOP PLACE, QUAY or BOARDING POSITION within a STOP PLACE.

Table 642 — *DynamicStopAssignment – Element*

Classification	Name	Type	Cardinality	Description
::>	::>	*DynamicStopAssignment*	::>	DYNAMIC STOP ASSIGNMENT inherits from PASSENGER STOP ASSIGNMENT.
«PK»	***Id***	*DynamicAssignment-IdType*	1:1	Identifier of DYNAMIC STOP ASSIGNMENT.
«FK»	***JourneyPattern-Ref***	*JourneyPatternRef*	0:1	JOURNEY PATTERN that DYNAMIC STOP ASSIGNMENT applies to + v1.1.
«FK»	***PassengerStop-AssignmentRef***	*PassengerStop-AssignmentRef*	0:1	PASSENGER STOP ASSIGNMENT that DYNAMIC STOP ASSIGNMENT overrides.

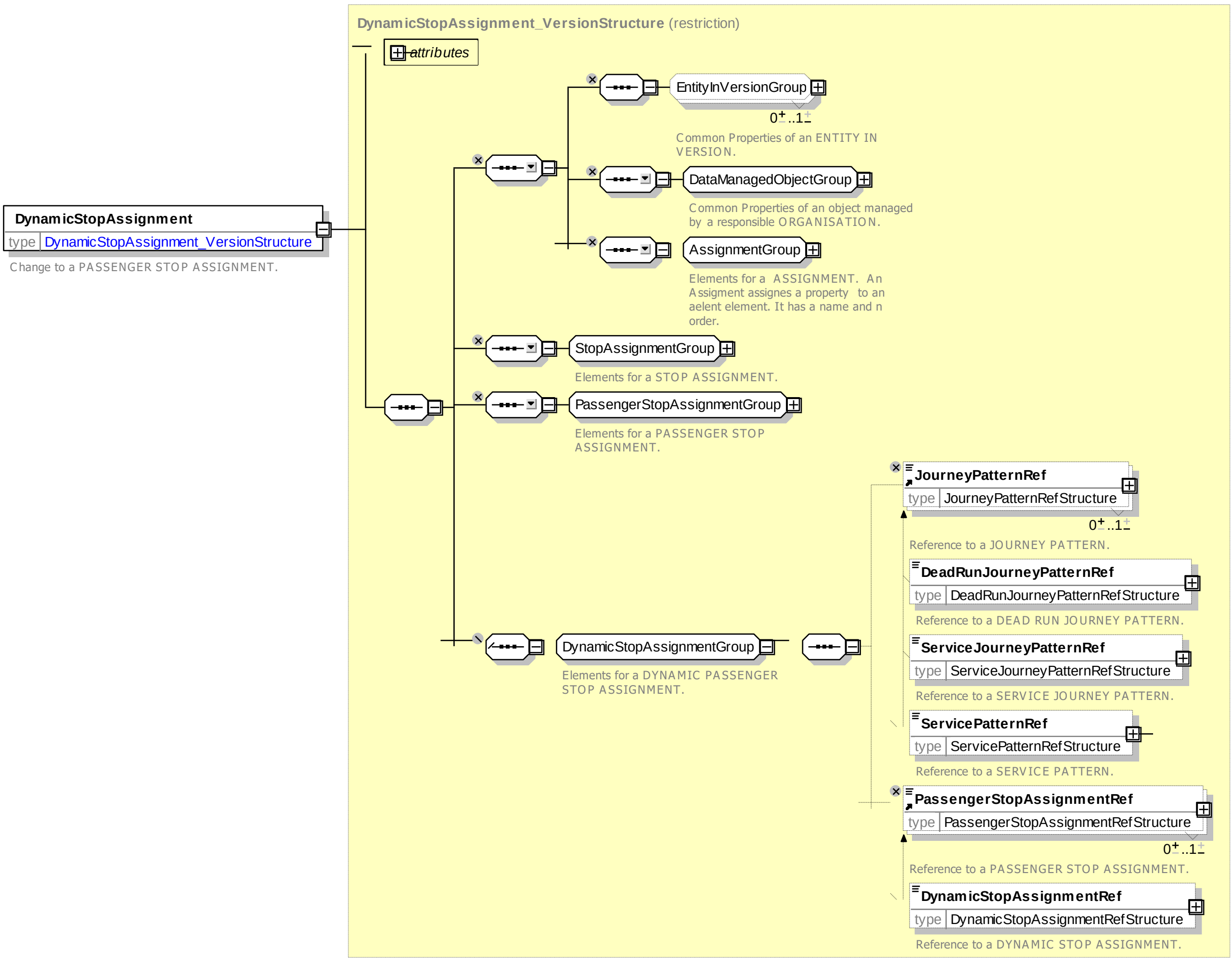


Figure 764 – *DynamicStopAssignment – XSD*

8.5.8.5 XML Example of StopAssignment

The following XML code fragment shows part of a STOP ASSIGNMENT assigning a SCHEDULED STOP POINT to a platform i.e. QUAY within a STOP PLACE.

EXAMPLE XML Example of StopAssignment – Rail Station with Platforms

```
<PassengerStopAssignment version="any" id="hde:PSA_40016_A1">
    <Description lang="en">Assigns Timetable timetable ref to physical stop</Description>
    <ScheduledStopPointRef version="032" ref="hde:SSP0042A"/>
    <StopPlaceRef version="any" ref="mybus:SSP_02456A"/>
    <QuayRef version="any" ref="mybus:Q_SSP_02456A_1"/>
</PassengerStopAssignment>
```

8.5.9 Train Stop Assignment

8.5.9.1 TRAIN STOP ASSIGNMENT – Conceptual MODEL

A TRAIN STOP ASSIGNMENT describes the alignment of the carriages of a train with the BOARDING POSITIONs of a QUAY so that, for example, exact guidance can be given to passengers as to where to stand on the platform to access a particular carriage or part of the train.

Each TRAIN COMPONENT of a TRAIN can be assigned to a SCHEDULED STOP POINT and a specific QUAY and BOARDING POSITION of a STOP PLACE.

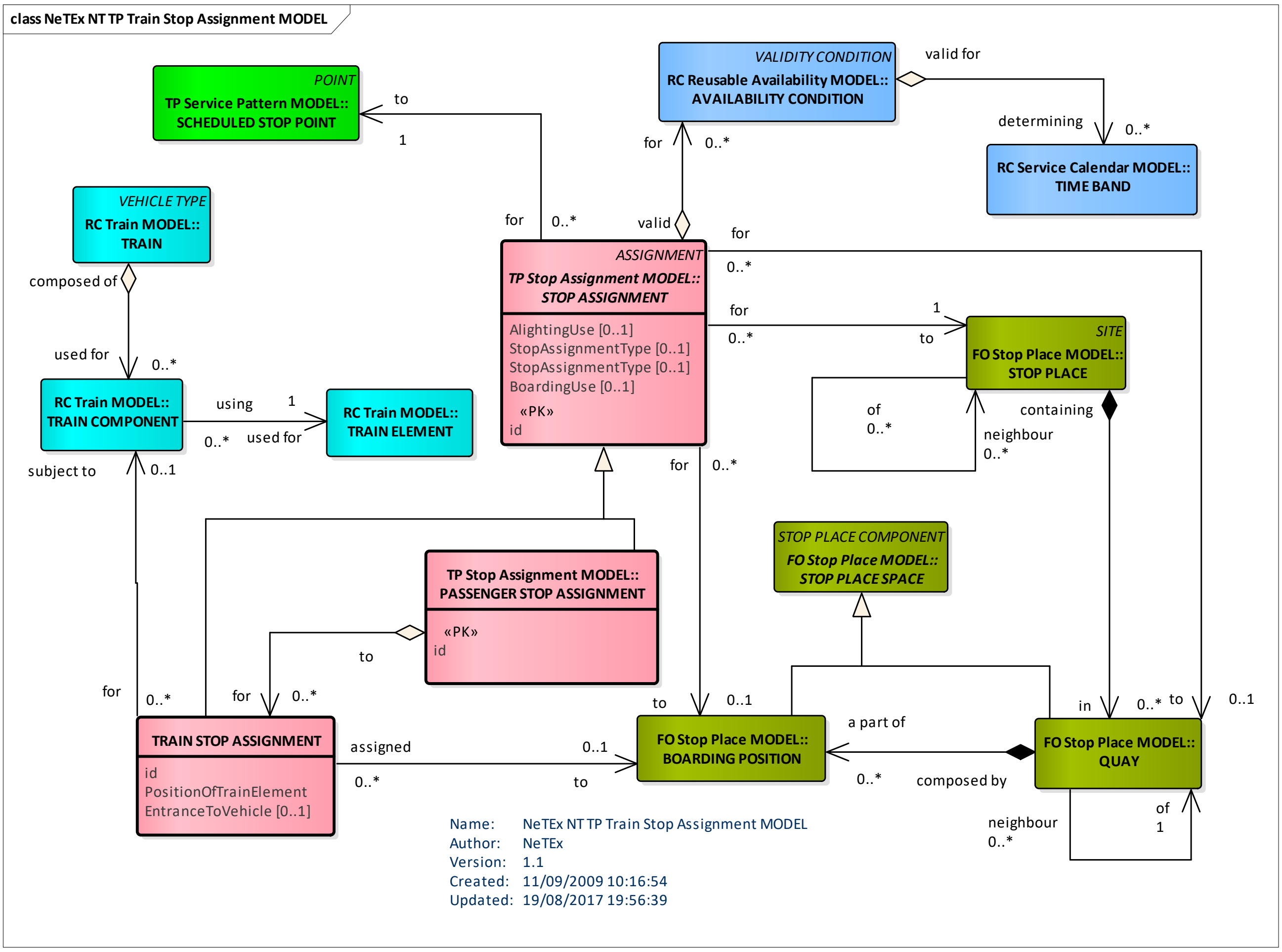


Figure 765 – Train Stop Assignment – Conceptual MODEL (UML)

8.5.9.2 Train Stop Assignment – Example

The following figure shows how a TRAIN STOP ASSIGNMENT can be used to relate specific TRAIN COMPONENTs to specific BOARDING POSITIONs.

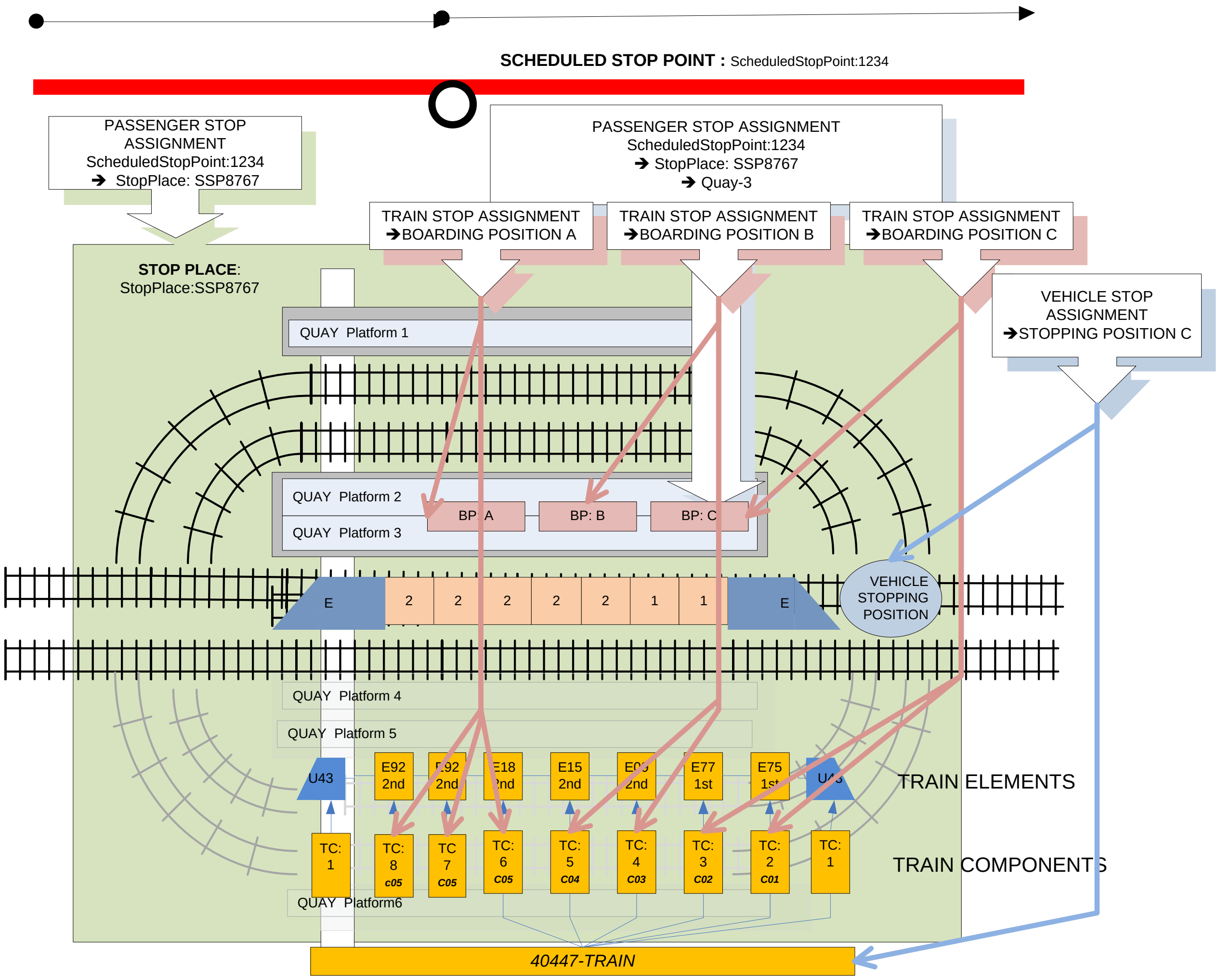


Figure 766 – Train Assignment – Example

8.5.9.3 Train Stop Assignment – Physical Model

The following figure shows detailed attributes of the TRAIN STOP ASSIGNMENT model.

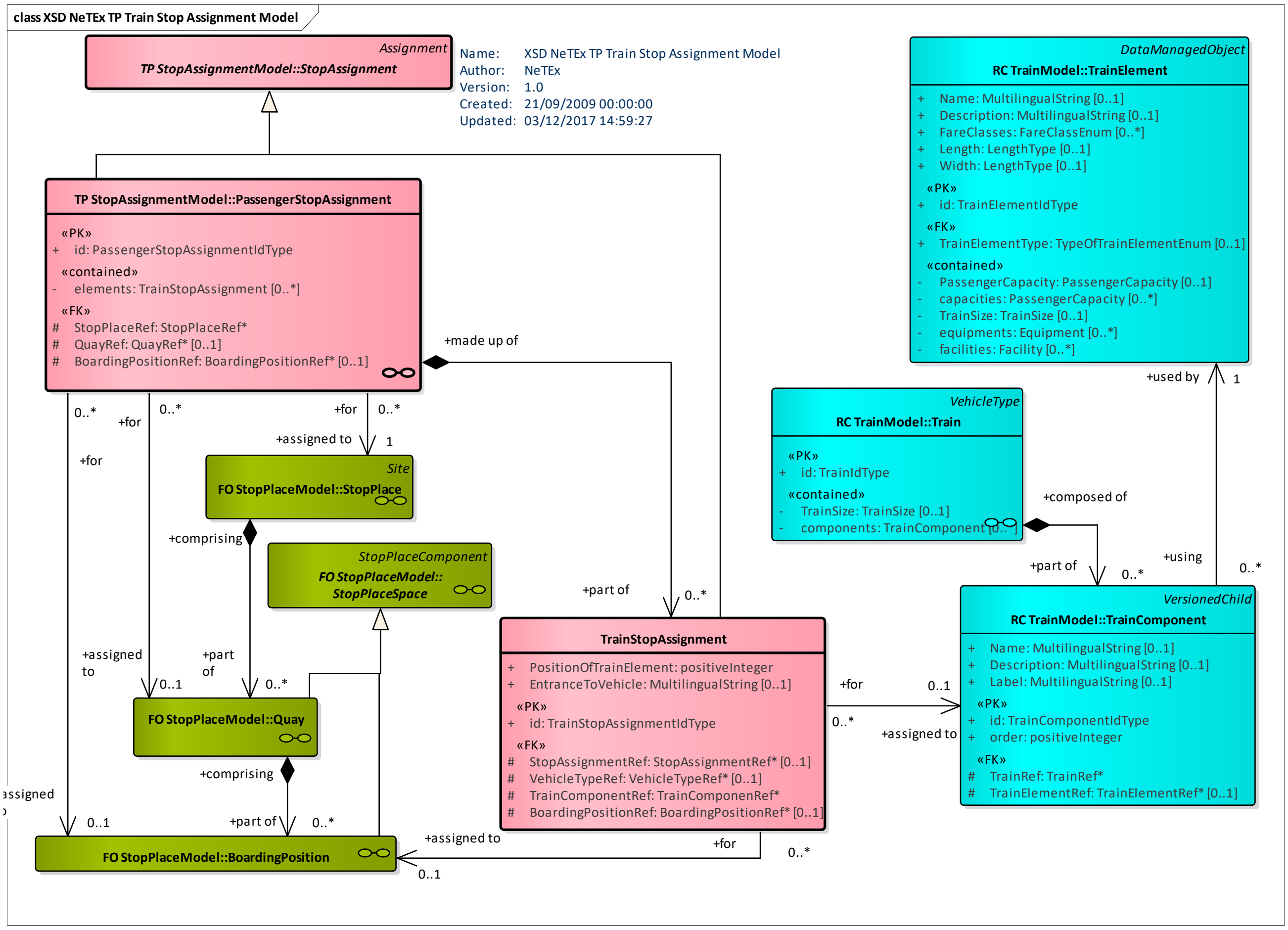


Figure 767 – Train Stop Assignment – Physical Model (UML)

8.5.9.4 Train Stop Assignment – Attributes and XSD

8.5.9.4.1 TrainStopAssignment – Model Element

The association of a TRAIN COMPONENT at a SCHEDULED STOP POINT with a specific STOP PLACE and also possibly a specific QUAY and BOARDING POSITION.

Table 643 – *TrainStopAssignment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*StopAssignment*	::>	TRAIN STOP ASSIGNMENT inherits from STOP ASSIGNMENT.
«PK»	***id***	*TrainStopAssignment\-IdType*	1:1	Identifier of TRAIN STOP ASSIGNMENT.
«FK»	***PassengerStop-AssignmentRef***	*PasssengerStopAssignmentRef*	0:1	PASSENGER STOP ASSIGNMENT that this is part of
«FK»	***TrainRef***	*TrainRef*	0:1	TRAIN to which TRAIN STOP ASSIGNMENT applies.

		CHOICE	0:1	Can be reference or view.
«FK»	***a*** ***TrainComponent-Ref***	*TrainComponenRef*	0:1	TRAIN COMPONENT to which TRAIN STOP ASSIGNMENT applies.
«FV»	***b*** ***TrainComponent-Viewf***	*TrainComponenView*	0:1	View of TRAIN COMPONENT to which TRAIN STOP ASSIGNMENT applies. (Can contain additioal derived data)
	PositionOfTrain-Element	xsd:*positiveInteger*	0:1	Relative position of TRAIN ELEMENT.
«FK»	***BoardingPositionRef***	*BoardingPositionRef*	0:1	Reference to a BOARDING POSITION with QUAY at STOP PLACE to which the SCHEDULED STOP POINT is assigned.
	EntranceToVehicle	*MultilingualString*	0:1	Description of a specific ENTRANCE to the VEHICLE. E.g. Front, rear.

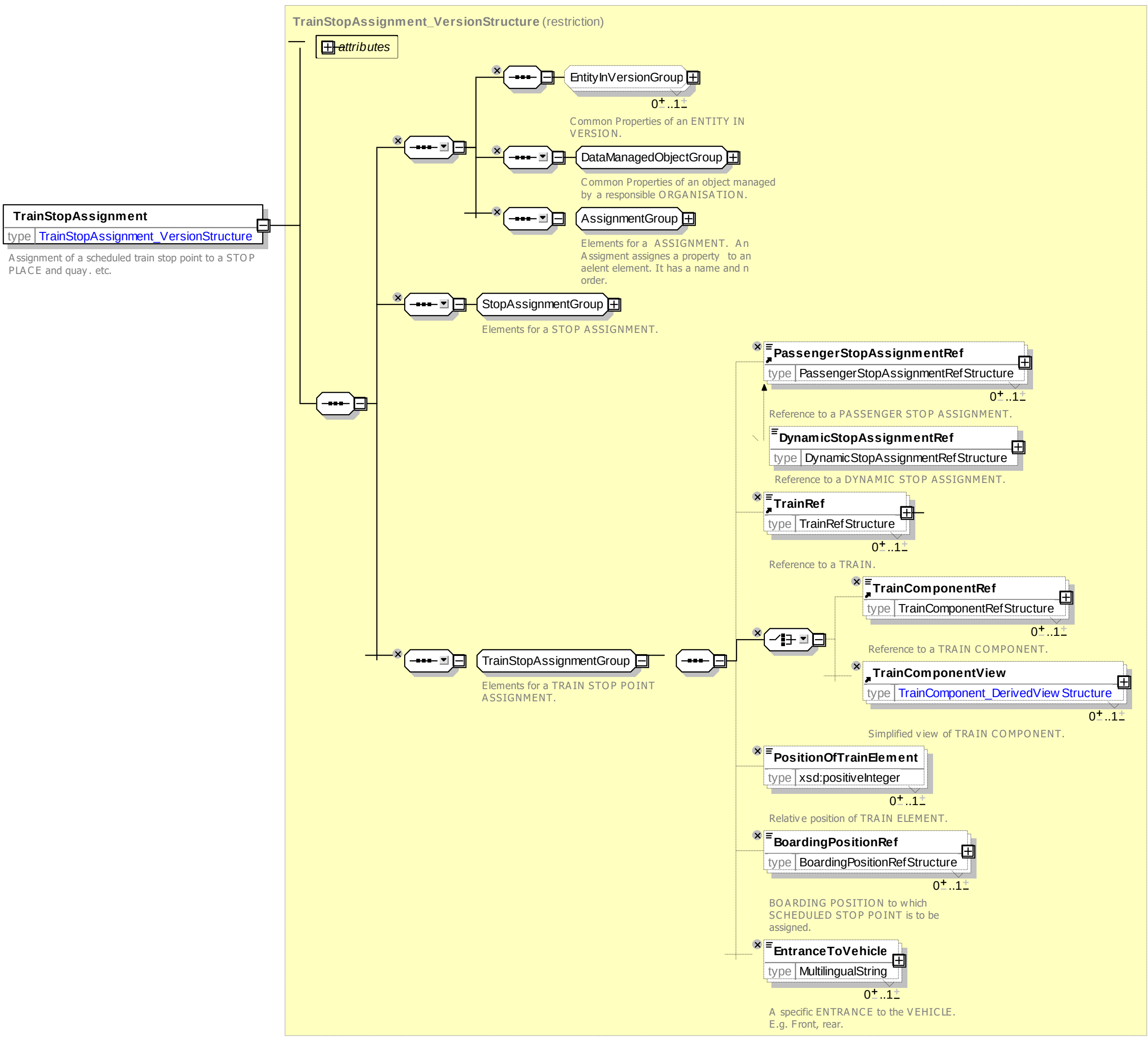


Figure 768 – *TrainStopAssignment – XSD*

8.5.9.5 XML Example of TrainStopAssignment

The following XML code fragment shows part of a TRAIN STOP ASSIGNMENT assigning sections of a QUAY to parts of a train.

EXAMPLE


```xml
<PassengerStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02">
    <Description lang="en">London St Pancras International Platform 5 Coaches 1 -6</Description>
    <ScheduledStopPointRef version="any" ref="eos:SSP_01_uk_StPancras"/>
    <StopPlaceRef version="1" ref="napt:9100STPADOM"/>
    <QuayRef version="any" ref="napt:Quay:QY_01_9100STPADOM_5"/>
    <BoardingPositionRef version="any" ref="napt:BP_01_9100STPADOM_021_A"/>
    <trainElements>
        <TrainStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02_01">
            <Description>Assign Coach 1 to Boarding Area A (Coaches 1 to 6)</Description>
            <TrainComponentView>
                <TrainComponentRef version="any" ref="est:TRNCMP_LP01_C01"/>
                <Label>Carriage 1</Label>
                <Description>Standard Family</Description>
                <FareClasses>standardClass</FareClasses>
                <TrainElementType>carriage</TrainElementType>
            </TrainComponentView>
        </TrainStopAssignment>
        <TrainStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02_02">
            <Description>Assign Coach 2 to Boarding Area A (Coaches 1 to 6)</Description>
            <TrainComponentView>
                <TrainComponentRef version="any" ref="est:TRNCMP_LP01_C02"/>
                <Label>Carriage 2</Label>
                <FareClasses>standardClass</FareClasses>
                <TrainElementType>carriage</TrainElementType>
            </TrainComponentView>
        </TrainStopAssignment>
        <TrainStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02_03">
            <Description>Assign Coach 3 to Boarding Area A (Coaches 1 to 6)</Description>
            <TrainComponentView>
                <TrainComponentRef version="any" ref="est:TRNCMP_LP01_C03"/>
                <Label>Carriage 3</Label>
                <FareClasses>standardClass</FareClasses>
                <TrainElementType>carriage</TrainElementType>
            </TrainComponentView>
        </TrainStopAssignment>
        <TrainStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02_04">
            <Description>Assign Coach 4 to Boarding Area A (Coaches 1 to 6)</Description>
            <TrainComponentView>
                <TrainComponentRef version="any" ref="est:TRNCMP_LP01_C04"/>
                <Label>Carriage 4</Label>
                <FareClasses>standardClass</FareClasses>
                <TrainElementType>carriage</TrainElementType>
            </TrainComponentView>
        </TrainStopAssignment>
        <TrainStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02_05">
            <Description>Assign Coach 5 to Boarding Area A (Coaches 1 to 6)</Description>
            <TrainComponentView>
                <TrainComponentRef version="any" ref="est:TRNCMP_LP01_C05"/>
                <Label>Carriage 5</Label>
                <FareClasses>standardClass</FareClasses>
                <TrainElementType>carriage</TrainElementType>
            </TrainComponentView>
```

```
        </TrainStopAssignment>
        <TrainStopAssignment version="any" id="eos:PSA_01_uk_StPancras_02_06">
            <Description>Assign Coach 6 to Boarding Area A (Coaches 1 to 6)</Description>
            <TrainComponentView>
                <TrainComponentRef version="any" ref="est:TRNCMP_LP01_C06"/>
                <Label>Carriage 6</Label>
                <FareClasses>standardClass</FareClasses>
                <TrainElementType>buffetCar</TrainElementType>
            </TrainComponentView>
        </TrainStopAssignment>
    </trainElements>
</PassengerStopAssignment>
```

8.5.10 Path Assignment

A NAVIGATION PATH ASSIGNMENT associates a physical NAVIGATION PATH with a CONNECTION link as one of the recommended ways to make a transfer between the points of the connection.

8.5.10.1 PATH ASSIGNMENT – Conceptual MODEL

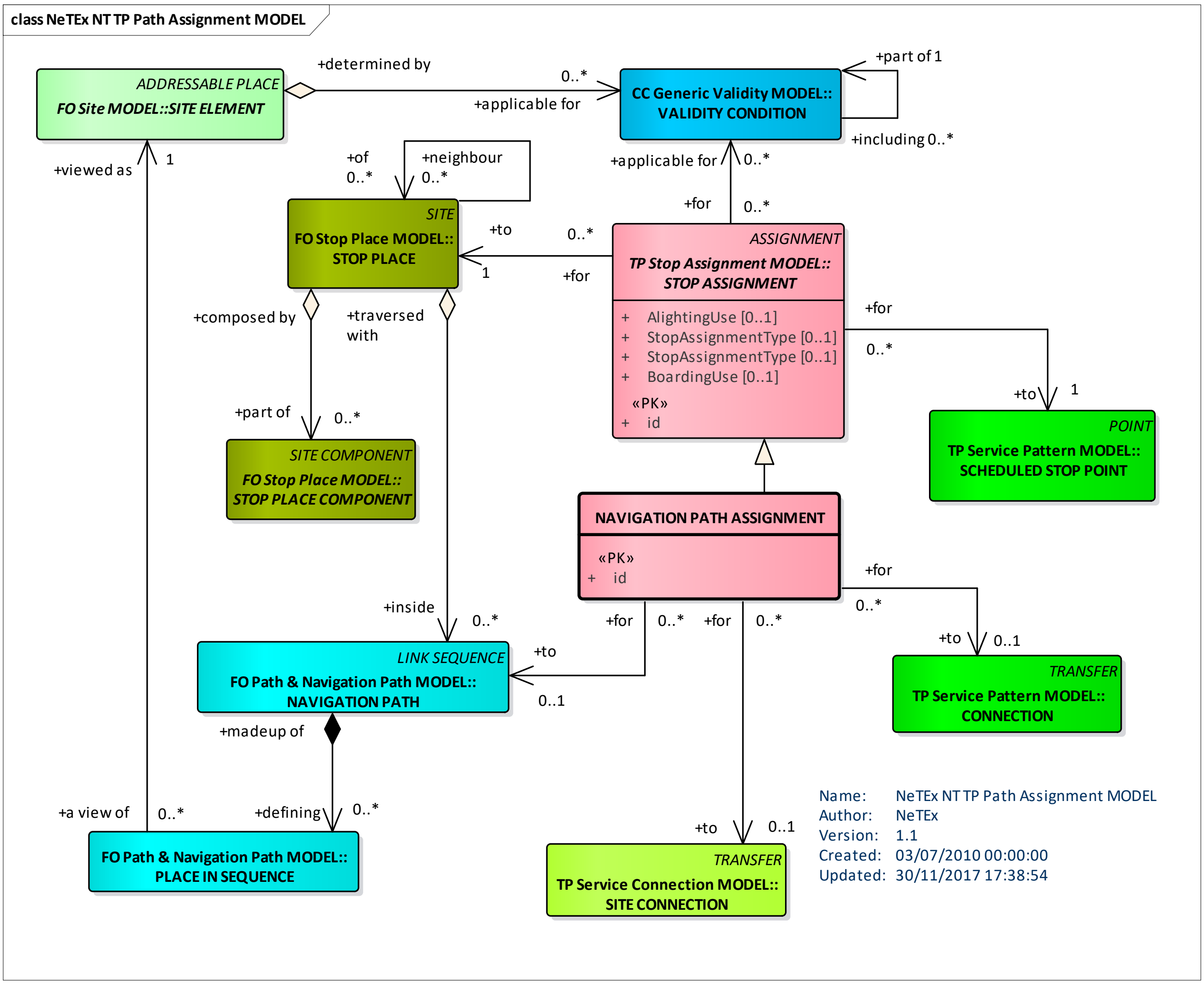


Figure 769 – Path Assignment – Conceptual MODEL (UML)

8.5.10.2 Path Assignment – Physical Model

The following figure shows detailed attributes of the PATH ASSIGNMENT model.

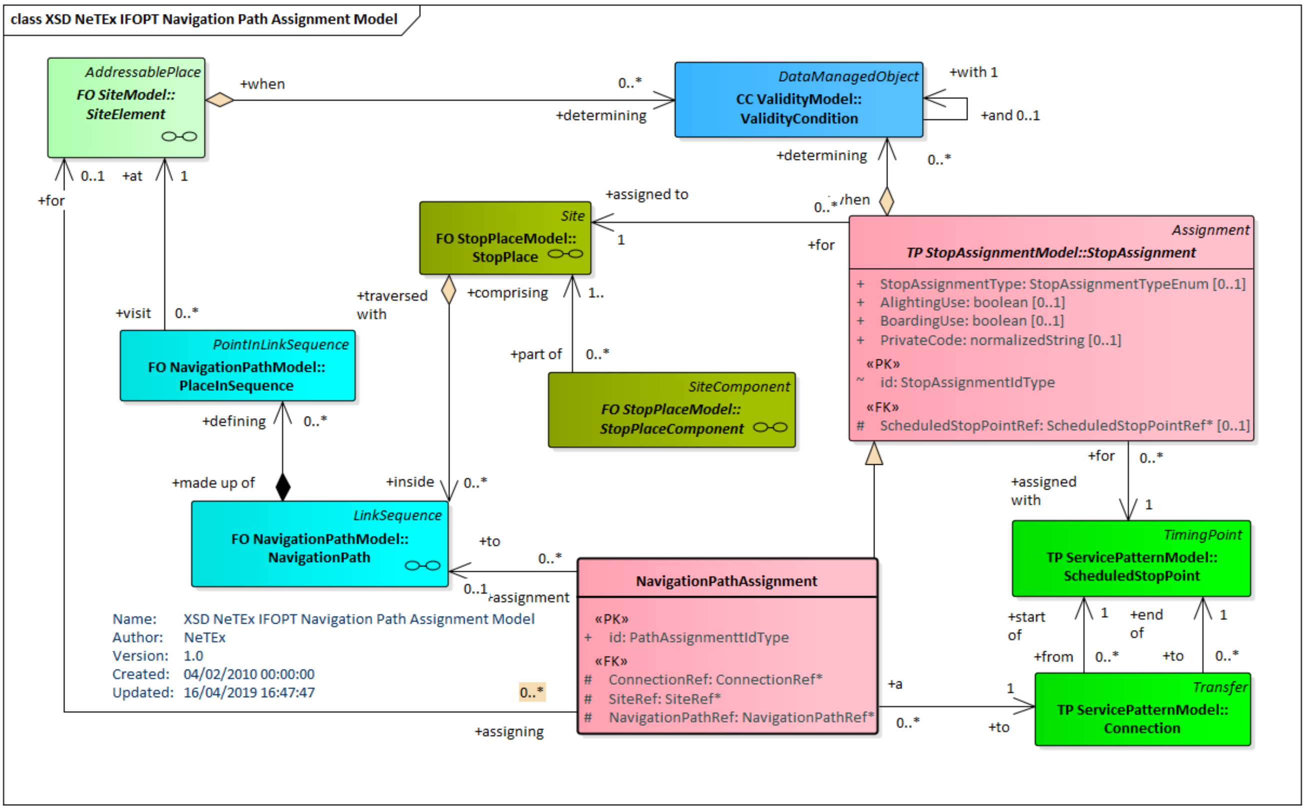


Figure 770 – Path Assignment – Physical Model (UML)

8.5.10.3 Path Assignment – Attributes and XSD

8.5.10.3.1 NavigationPathAssignment – Model Element

The NavigationPathAssignment models the allocation of a NAVIGATION PATH to a specific STOP POINT ASSIGNMENT, for example to indicate the path to be taken to make a CONNECTION.

Table 644 – *NavigationPathAssignment* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*Assignment*	::>	NAVIGATION PATH ASSIGNMENT inherits from ASSIGNMENT.
«PK»	***id***	*PathAssignmenttIdType*	0:1	Identifier of a NAVIGATION PATH ASSIGNMENT.
«FK»	***ConnectionRef***	*ConnectionRef*	0:1	Reference to a CONNECTION to which NAVIGATION PATH is assigned.
FK»	***SiteRef***	*(SiteRef)*	0:1	Reference to a SITE to which NAVIGATION PATH is assigned.
«FK»	***NavigationPath-Ref***	*NavigationPathRef*	0:1	Reference to a NAVIGATION PATH that is to be assigned.

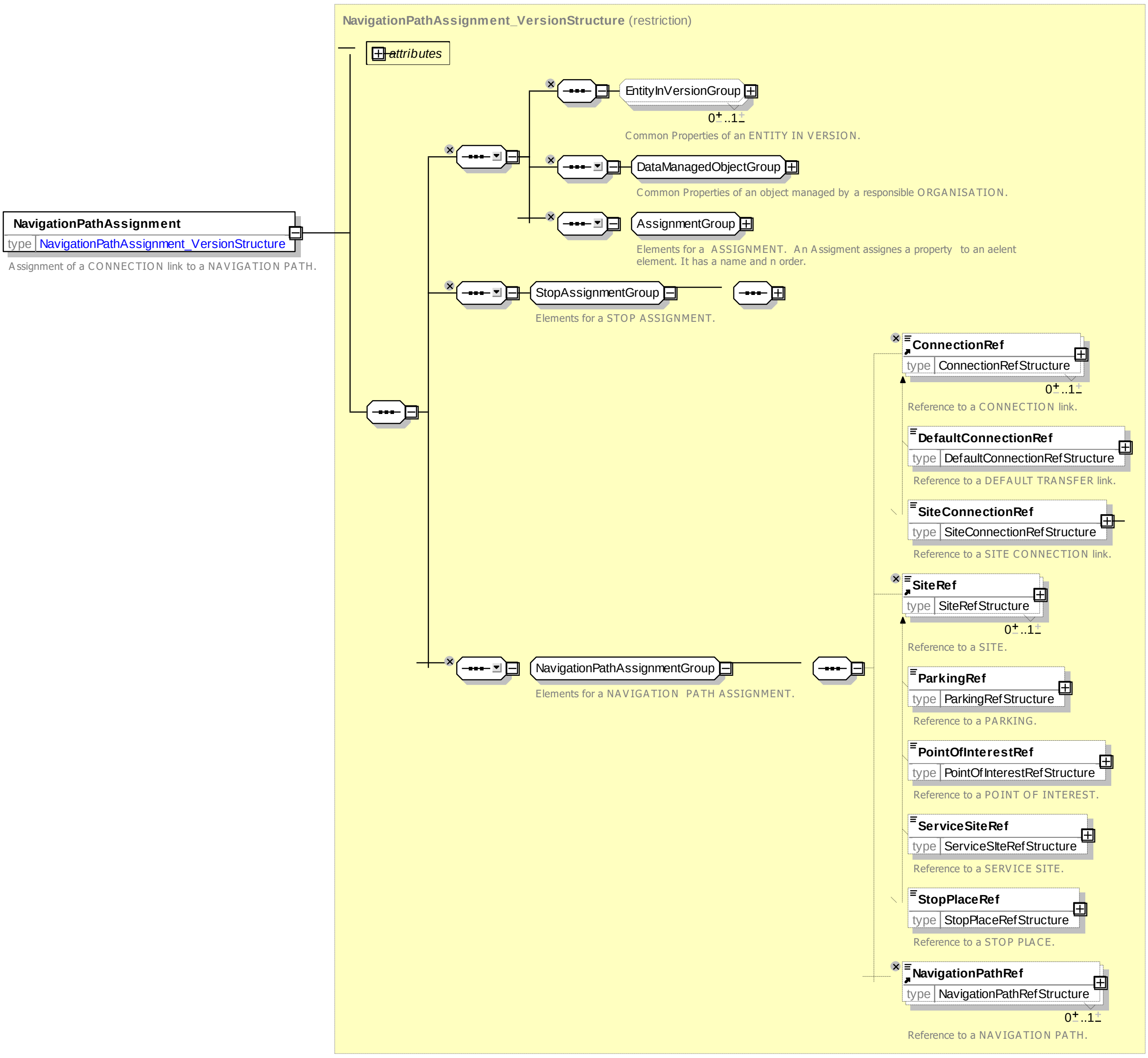


Figure 771 – *NavigationPathAssignment – XSD*

8.5.11 Passenger Information Equipment

8.5.11.1 Passenger Information Equipment – Conceptual MODEL

Displays at stops or on board of vehicles, kiosk terminals or information desks are examples for passenger information facilities. PASSENGER INFORMATION EQUIPMENT is therefore classified by the generic entity TYPE OF PASSENGER INFORMATION EQUIPMENT. The PASSENGER INFORMATION EQUIPMENT can be associated with a LOGICAL DISPLAY; using a DISPLAY ASSIGNMENT a LOGICAL DISPLAY can be assigned to SCHEDULED STOP POINTs and JOURNEY PATTERNs. A JOURNEY PATTERN itself can be associated with a DESTINATION DISPLAY entity that describes the passenger information displays on the vehicle.

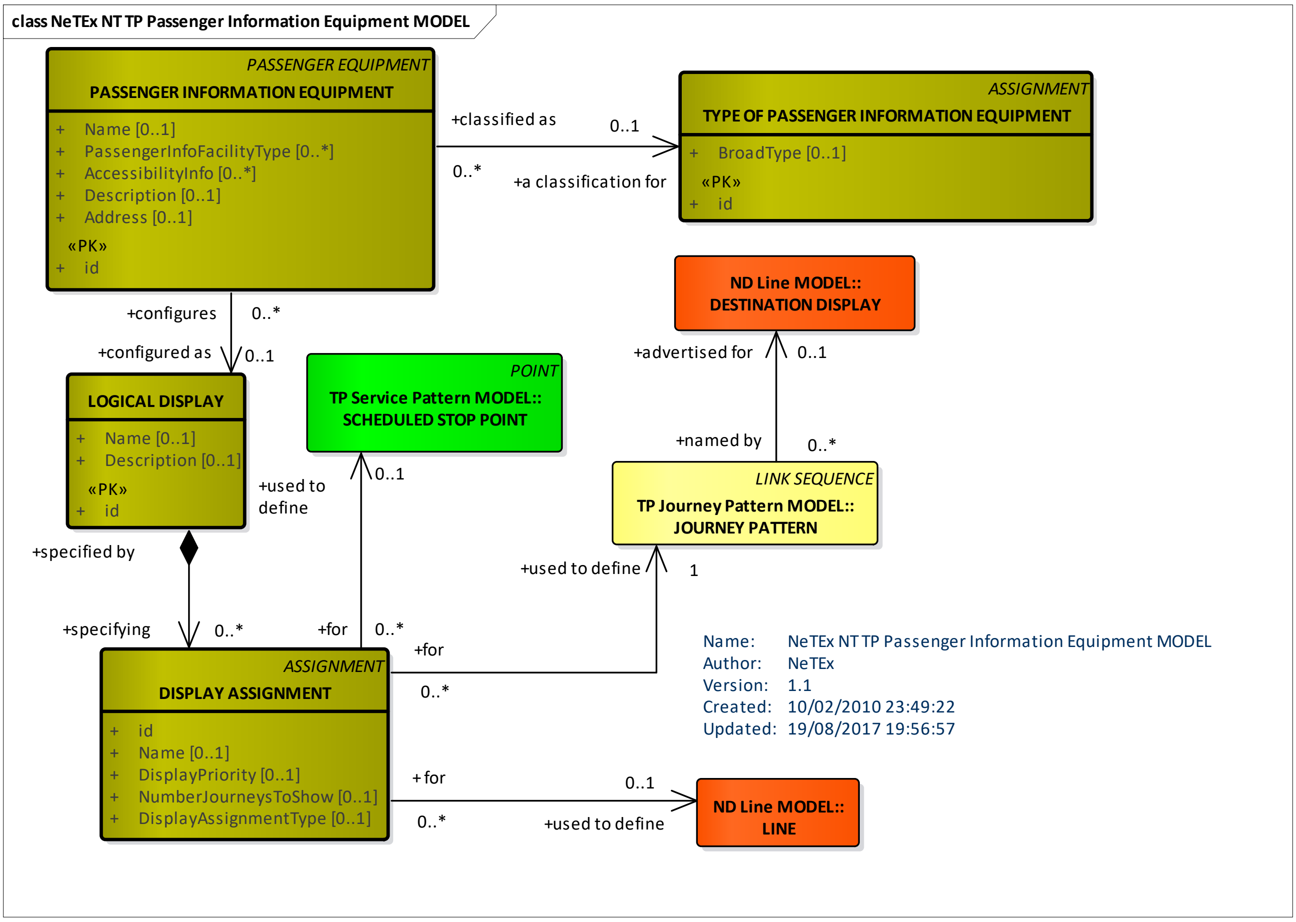


Figure 772 – Passenger Information Equipment – Conceptual MODEL (UML)

8.5.11.2 Passenger Information Equipment – Physical Model

The following figure shows detailed attributes of the PASSENGER INFORMATION EQUIPMENT model.

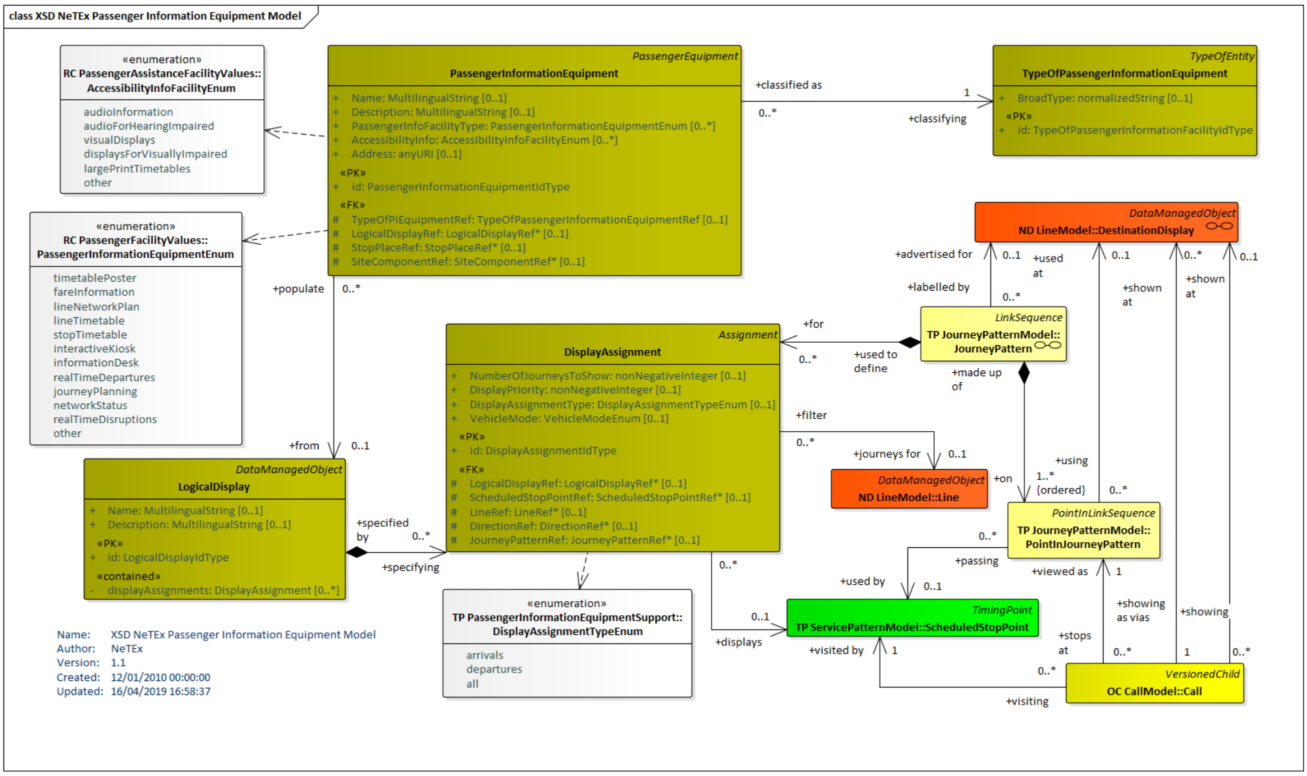


Figure 773 – *Passenger Information Equipment* – Physical Model (UML)

8.5.11.3 Passenger Information Equipment – Attributes and XSD

8.5.11.3.1 PassengerInformationEquipment – Model Element

A public transport information piece of equipment, as for instance terminals (on street, at information desks, telematic, etc) or printed material (leaflets displayed at stops, booklets, etc.).

Table 645 – *PassengerInformationEquipment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*DataManagedObject*	*::>*	PASSENGER INFORMATION EQUIPMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*PassengerInformation-EquipmentIdType*	1:1	Identifier of PASSENGER INFORMATION EQUIPMENT.
	Image	*xsd:anyURI*	0:1	Default image for EQUIPMENT.
«FK»	***LogicalDisplayRef***	*LogicalDisplayRef*	0:1	LOGICAL DISPLAY defining contents of PASSENGER INFORMATION EQUIPMENT.
«FK»	***StopPlaceRef***	*StopPlaceRef*	0:1	STOP PLACE associated with PASSENGER INFORMATION EQUIPMENT.
«FK»	***SiteComponent-Ref***	*SiteComponentRef*	0:1	SITE COMPONENT associated with PASSENGER INFORMATION EQUIPMENT.

«FK»	***TypeOfPassengerInformation-EquipmentRef***	*TypeOfPassenger-InformationEquipmentRef*	0:1	Type of PASSENGER INFORMATION EQUIPMENT.
«enum»	***PassengerInfo-FacilityListe***	*PassengerInformation-FacilityEnum*	0:*	Types of Passenger Travel Information available. See Facility Model for allowed values.
«enum»	***AccessibilityInfo-FacilityList***	*AccessibilityInfoFacility-Enum*	0:*	Types of Accessibility Info available on PASSENGER INFORMATION EQUIPMENT. See Facilities Model for allowed values.

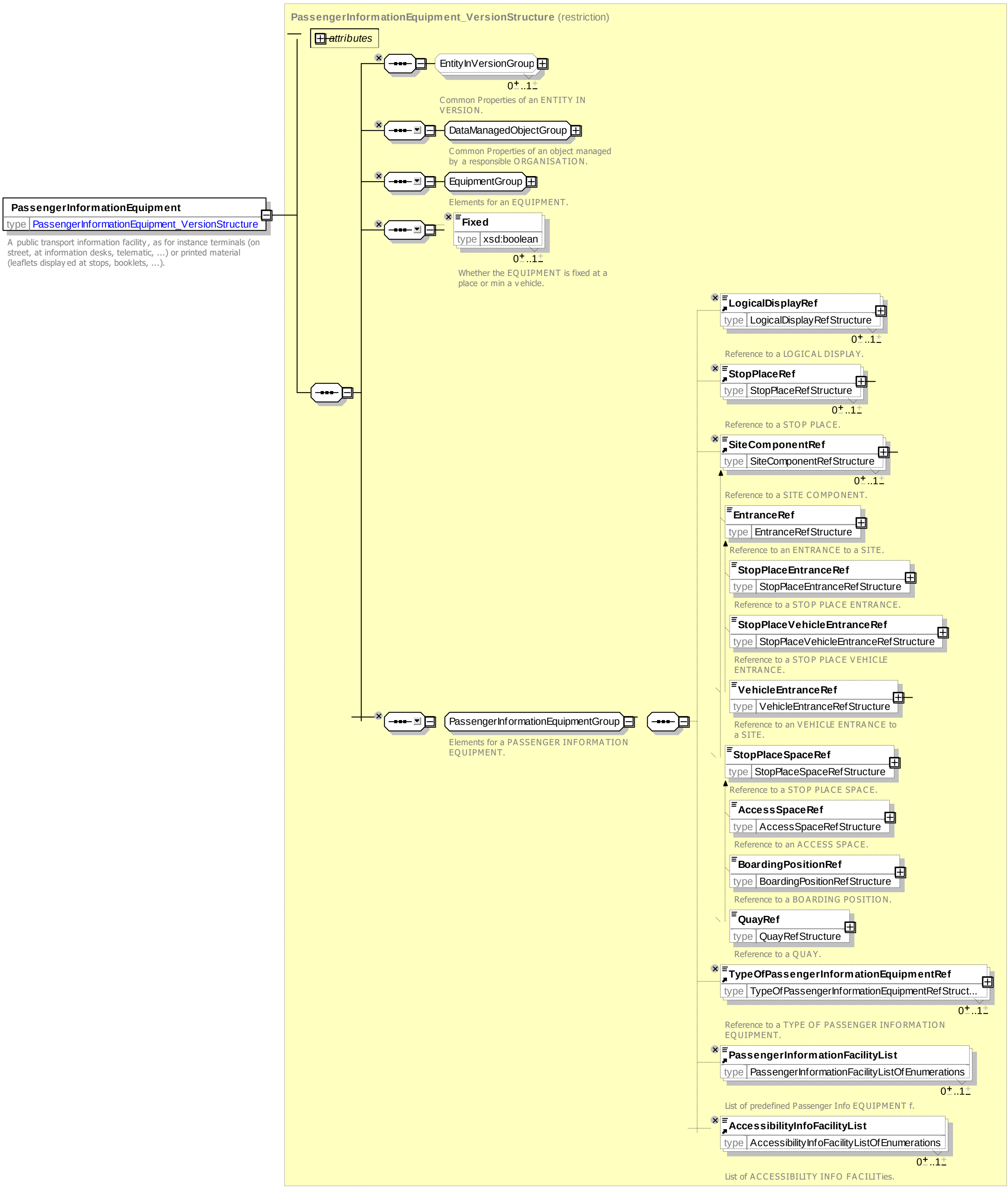


Figure 774 – *PassengerInformationEquipment – XSD*

8.5.11.3.2 DisplayAssignment – Model Element

The assignment of one STOP POINT and one JOURNEY PATTERN to a PASSENGER INFORMATION EQUIPMENT, specifying that information on this STOP POINT and this JOURNEY PATTERN will be provided (e.g. displayed, printed).

Table 646 – *DisplayAssignment* – Element

Classification	Name	Type	Cardinality	Description
::>	*::>*	*Assignment*	*::>*	DISPLAY ASSIGNMENT inherits from ASSIGNMENT.
«PK»	***id***	*DisplayAssignmentIdType*	1:1	Identifier of DISPLAY ASSIGNMENT.
«FK»	***LogicalDisplayRef***	*LogicalDisplayRef*	0:1	LOGICAL DISPLAY to which DISPLAY ASSIGNMENT applies.
«FK»	***ScheduledStopPoint-Ref***	*ScheduledStopPointRef*	0:1	Stop for which Data is to be shown on LOGICAL DISPLAY.
«enum»	***VehicleMode***	*VehicleModeEnum*	0:1	MODE for which Data is to be shown on LOGICAL DISPLAY.
«FK»	***LineRef***	*LineRef+*	0:1	LINE for which data is to be shown on LOGICAL DISPLAY.
«FK»	***DirectionRef***	*DirectionRef*	0:1	DIRECTION for which data is to be shown on LOGICAL DISPLAY.
«FK»	***JourneyPatternRef***	*JourneyPatternRef+*	0:1	JOURNEY PATTERN for which data is to be shown on LOGICAL DISPLAY.
«enum»	***DisplayAssignment-Type***	*DisplayAssignmentType*	1:1	Type Of DISPLAY ASSIGNMENT. See allowed values below.
	NumberJourneys-ToShow	*xsd:nonNegativeInteger*	1:1	The number of journeys of a given LINE to show.
	DisplayPriority	*xsd:nonNegativeInteger*	1:1	Relative priority of assignment for displaying.

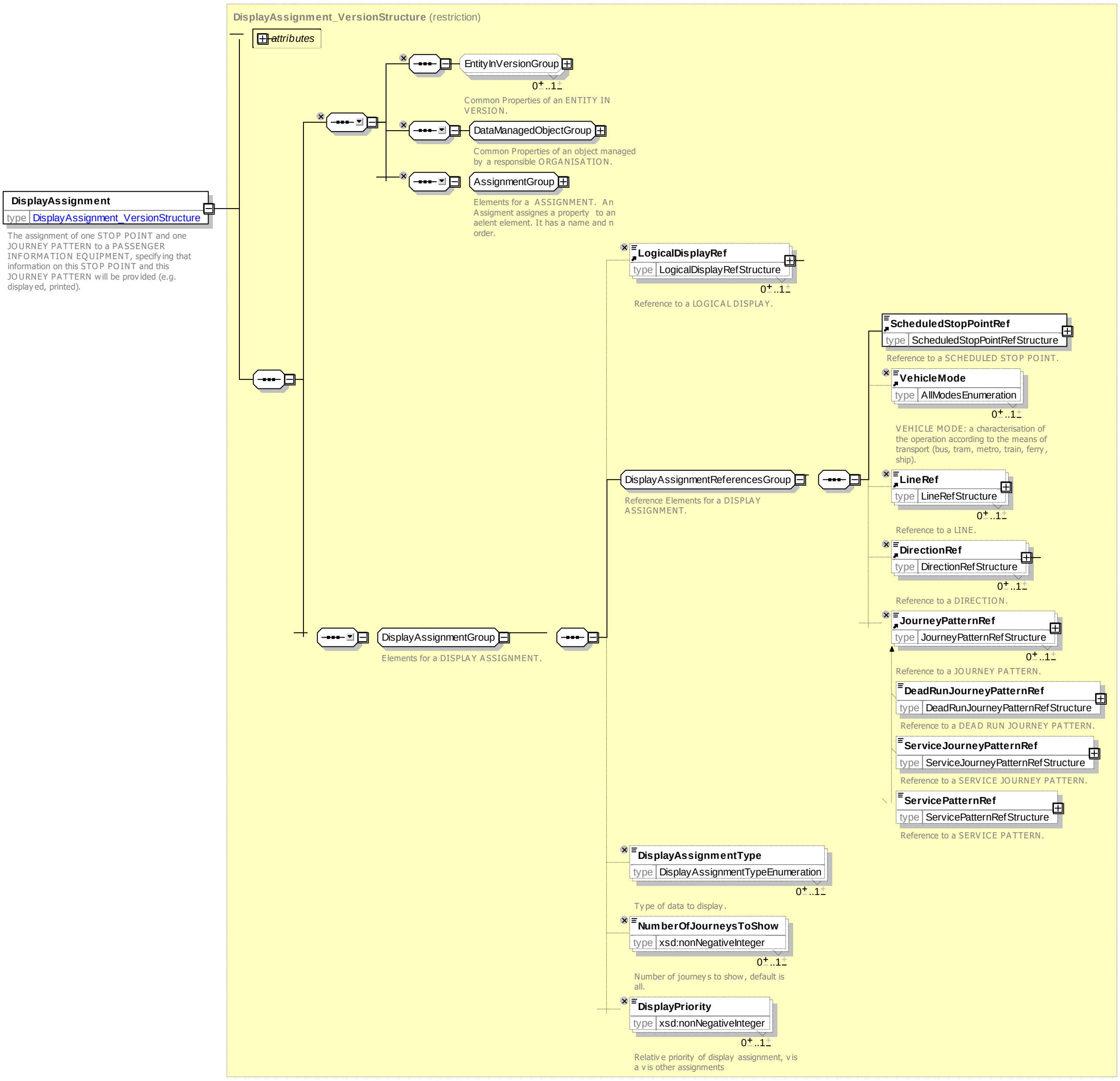


Figure 775 – *DisplayAssignment* – XSD

DisplayAssignmentType – Allowed Values

Allowed values for ***DisplayAssignmentType*** (*DisplayAssignmentTypeEnumeration*)

Table 647 – *DisplayAssignmentType* – Allowed Values

Value	Description
arrivals	Arrivals only
departures	Departures only
all	Both

8.5.11.3.3 *LogicalDisplay – Model Element*

A set of data that can be assembled for assignment to a physical PASSENGER INFORMATION EQUIPMENT or to a logical channel such as web or media. It is independent of any physical embodiment.

A LOGICAL DISPLAY may have a set of DISPLAY ASSIGNMENTs each of which associates a JOURNEY PATTERN whose journeys are to be shown at the LOGICAL DISPLAY. It may also be associated with a SCHEDULED STOP POINT.

LOGICAL DISPLAY corresponds to a SIRI STOP MONITORING point.

Table 648 – *LogicalDisplay* – Element

Classification	Name	Type	Cardinality	Description
::>	::>	*DataManagedObject*	::>	LOGICAL DISPLAY inherits from DATA MANAGED OBJECT.
«PK»	***id***	*LogicalDisplayIdType*	1:1	Identifier of LOGICAL DISPLAY.
	Name	*MultilingualString*	0:1	Name of LOGICAL DISPLAY.
	Description	*MultilingualString*	0:1	Description of LOGICAL DISPLAY.
«cntd»	***display-Assignments***	*DisplayAssignment*	0:*	Assignments of services to LOGICAL DISPLAY.

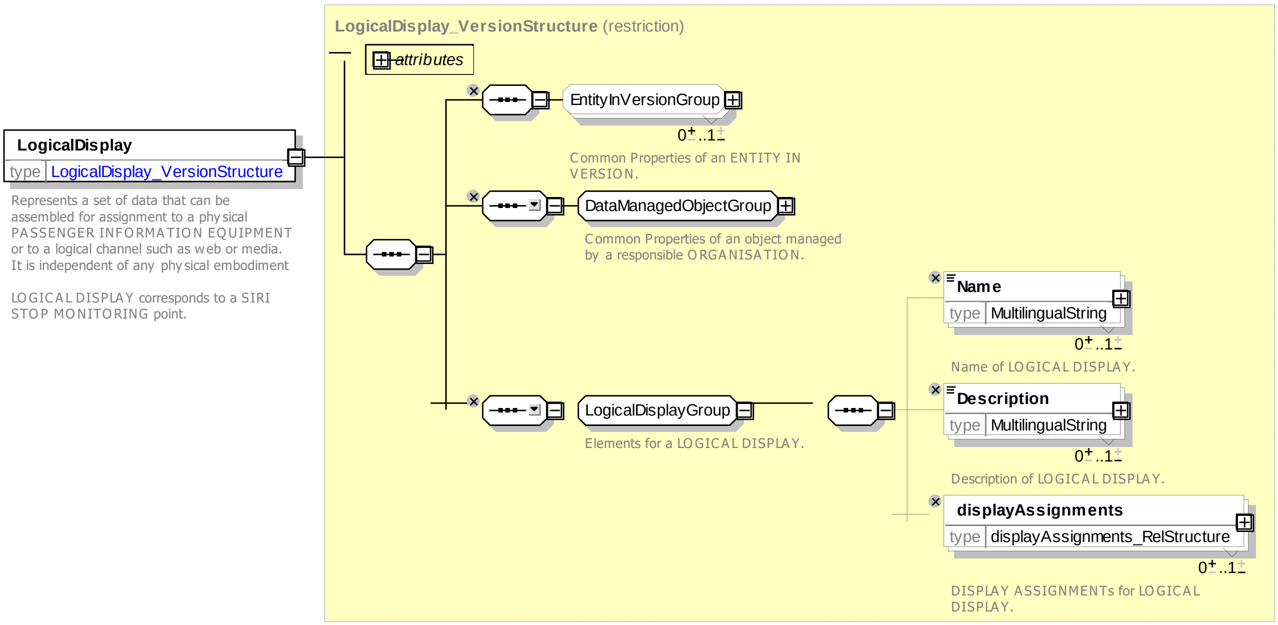


Figure 776 – *LogicalDisplay – XSD*

8.5.11.3.4 TypeOfPassengerInformationEquipment – Model Element

A Classification for PASSENGER INFORMATION EQUIPMENT (e.g. next stop indicator, stop announcements, passenger information facility).

Table 649 – *TypeOfPassengerInformationEquipment*– Element

Classification	Name	Type	Cardinality	Description

::>	::>	*DataManagedObject*	::>	TYPE OF PASSENGER INFORMATION EQUIPMENT inherits from DATA MANAGED OBJECT.
«PK»	***id***	*TypeOfPassengerInformation FacilityIdType*	1:1	Identifier of TYPE OF PASSENGER INFORMATION EQUIPMENT.
	BroadType	*xsd:normalizedString*	1:1	Classification of PASSENGER INFORMATION EQUIPMENT.

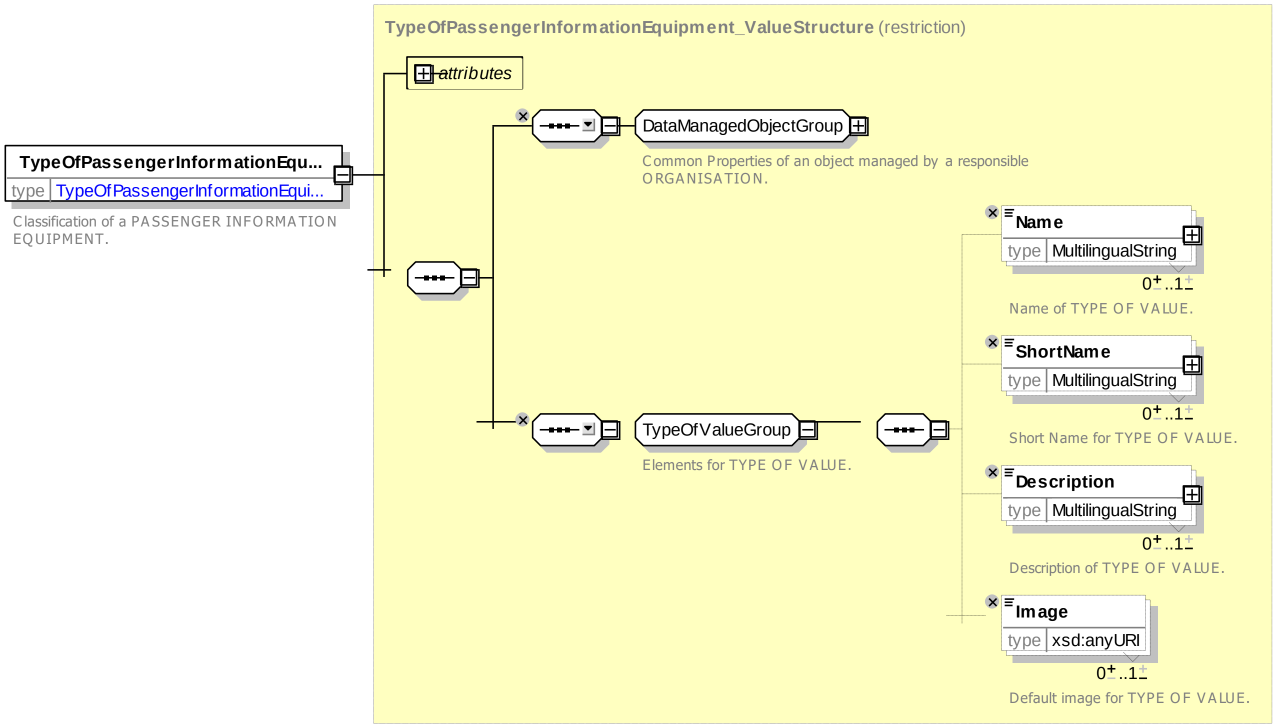


Figure 777 – *TypeOfPassengerInformationEquipment – XSD*

8.5.11.4 XML Examples for Passenger Information Equipment

8.5.11.4.1 Logical Display – XML Example

The following XML code fragment the definitions of two LOGICAL DISPLAYs that assign particular journeys to a LOGICAL DISPLAY.

EXAMPLE XML Example of LogicalDisplay

```
<LogicalDisplay version="any" id="mybus:LD001_TramBus_Q1_EastboundSouthbound">
    <Description>Departures Display for use on Bravo Bus/Tram Quay 1 Outbound</Description>
    <displayAssignments>
        <DisplayAssignment version="any" id="LD001_SSP002_24o_departures">
            <Name>Assign ScheduledStopPoint 002 (Bravo) route 24 Eastbound Departures to Display
LD001 </Name>
            <ScheduledStopPointRef version="any" ref="mybus;SSP_002"/>
            <ServiceJourneyPatternRef version="any" ref="hde:jp_24o"/>
            <DisplayAssignmentType>departures</DisplayAssignmentType>
            <NumberOfJourneysToShow>2</NumberOfJourneysToShow>
            <DisplayPriority>1</DisplayPriority>
```

```
            </DisplayAssignment>
        <DisplayAssignment version="any" id="LD001_SSP032_46o_departures">
                <Name>Assign ScheduledStopPoint 032 (Bravo) route 46 Southbound Departures to Display
LD001 </Name>
                <ScheduledStopPointRef version="any" ref="SSP_032"/>
            <ServiceJourneyPatternRef version="any" ref="hde:jp_46o"/>
                <DisplayAssignmentType>departures</DisplayAssignmentType>
                <NumberOfJourneysToShow>2</NumberOfJourneysToShow>
                <DisplayPriority>1</DisplayPriority>
                </DisplayAssignment>
    </LogicalDisplay>
```

8.5.11.4.2 Passenger Information Display – XML Example

The following XML code fragment the definitions of two PASSENGER INFORMATION EQUIPMENTs that that assign particular LOGICAL DISPLAYs to particular physical equipment located on particular platforms.

EXAMPLE XML Example of Passenger Information Display

```
        <PassengerInformationEquipment version="any" id="pif_SP002B_TramBus_Q1">
            <Name>Display on Bravo Tram bus departures Platform 1 </Name>
            <LogicalDisplayRef version="any" ref="LD001_TramBus_Q1_EastboundSouthbound"/>
            <StopPlaceRef version="any" ref="SP002B"/>
            <QuayRef version="any" ref="Q_SP002B_1"/>
            <TypeOfPassengerInformationEquipmentRef version="any" ref="PlatDisp"/>
        </PassengerInformationEquipment>
        <PassengerInformationEquipment version="any" id="pif_SP002B_TramBus_Q2">
            <Name>Display on Bravo Tram bus departures Platform 2 </Name>
            <LogicalDisplayRef version="any" ref="LD002_TramBus_Q2_WestboundNorthbound"/>
            <StopPlaceRef version="any" ref="StPlace:SP002B"/>
            <QuayRef version="any" ref="Q_SP002B_2"/>
            <TypeOfPassengerInformationEquipmentRef version="any" ref="mybus;PlatDisp"/>
        </PassengerInformationEquipment>
```

9 NeTEx Service Interface

9.1 Introduction

The core NeTEx schema describes a set of public transport data objects independently of any specific communication protocol and which can be grouped inside a NeTEx **publication**. The same content model, serialized as an XML document, can be transmitted electronically using a variety of different transport protocols, for example as a file by FTP; as an email attachment by SMTP; over HTTP as an attachment; or as a stream over a TCPIP socket.

In addition to the serializable content model, NeTEx provides a **service interface** with which to make programmatic requests to a server to return a set of the NeTEx data elements that meet some given criterion. The service interface allows the scope of requests and responses also to be described in terms of NeTEx elements, using an extended schema that defines a set of messages also described in XML. This meets two main design requirements ***DELIVERY-003: Filtered delivery*** and ***DELIVERY-05: Periodic Delivery, Scheduled delivery at regular intervals***.

The NeTEx web service makes use of the SIRI communication layer which provides common mechanisms for conducting both request/response and publish/subscribe interactions to exchange data. The NeTEx SIRI service can thus be regarded as an additional SIRI service that complements the existing SIRI real-time services and can be used to provision real-time systems with reference data. Reusing a common communication layer for all the various technical services for transport data exchange allows implementers a useful degree of reuse of code and know-how, making it easier and cheaper to build interfaces and facilitating the interoperation of common standards.

9.2 Protocols versus payload

A protocol defines the sequence of messages that need to be exchanged in order for two systems to communicate including behaviour to recover from errors, loss of communication, etc.

It is desirable to separate the concerns of a protocol (such as addresses, method of communication, time of communication, currency of data, error codes, recovery behaviour) from those of the semantics of the payload being exchanged as this gives greater adaptability over time and allows different protocols to be harnessed in different circumstances to exchange the same underlying data. A well-known example of this separation is seen in the http protocol which can be used to exchange a wide variety of mime types and content.

However for large and complex data sets, certain aspects of the organisation of the payload have at least some bearing on the choice of messages and the query semantics required. It is usually neither desirable nor viable to exchange the whole dataset – and where only parts of a model are exchanged, additional considerations apply to the selection, aggregation, validity and coherence of data. Different sequences of messages may be needed to meet different requirements, for example to distribute a baseline set of timetables, changes to a known timetable, or updates to an information about connecting journeys for those changes. It may first be necessary to discover certain data elements in order to request further details, for example “Give me the areas you know about”, then “Give me the stops in area Z2”, then “”Give me the routes that connect to stop 21”, etc.

The NeTEx specification includes two standard protocols designed to allow both bulk and fine grained exchange of data sets and flexible query semantics. They are intended to provide a standard method for data exchange using a uniform XML based technology - but do not preclude the use of the NeTEx content model in other protocols in future.

- Simple bulk **publication** – An exporting system outputs as an XML document a file containing all the data elements for a given purpose for a specified area or areas and period, with a header indicating the date of production and other metadata. This is very similar to current systems that regularly output an export file according to a given prescription, say the stops and localities provided in the UK NaPTAN service - or a VDV452 timetable export. **Filter** objects describe what has been included in the delivery along with its export and validity dates, but the request process and the actual semantics of the request are not part of the protocol.

- Exchange over http as a **SIRI service**: the same content model can be embedded as an XML document in a SIRI data service. A SIRI ***DataObjectRequest*** requests a specific set of NeTEx objects. A SIRI ***DataObjectResponse*** returns a set of data elements (also a ***PublicationDelivery***) that meet the request criteria, grouped in one or more frames. Filters on the request specify the scope in terms of NeTEx model elements. The NeTEx SIRI Service can be used with WSDL bindings or as plain http attachments.

There are two separate root schemas, sharing common subpackages that describe the payload model. The SIRI schema in effect just adds the generic SIRI service elements and a request. The schemas include referential integrity constraints to enforce key uniqueness of most elements.

Table 650 – NeTEx XSD schemas

Name	Schema	Description
NeTEx Publication schema	***NeTEx_publication.xsd***	Simple publication of NeTEx data as an XML document.
NeTEx SIRI protocol Schema	***NeTEx_siri.xsd***	A series of messages Documents with SIRI wrappers

9.3 NeTEx Publication XSD schema

The NeTEx publication schema describes an XML document suitable for exchange either as a stand-alone document, or embedded in a SIRI-NX Delivery, (that is the response to a SIRI request for NETEX data). A publication wraps one or more NeTEx VERSION FRAMEs inside a ***PublicationDelivery*** element. Each

PublicationDelivery acts as a container and is annotated with a time stamp. It can also embed a ***PublicationRequest*** element that records filter parameters describing the delivery contents.

9.3.1 NeTEx Publication Delivery – Physical Model

9.3.1.1 NeTEx Publication Delivery - Overview of Physical Model

The primary element in the publication schema is a ***PublicationDelivery,*** comprising a simple header, followed by a payload of VERSION FRAMES containing NeTEx data objects as described in NeTEx parts 1, 2 and 3. The delivery may also include some optional metadata indicating the classes present in the payload.

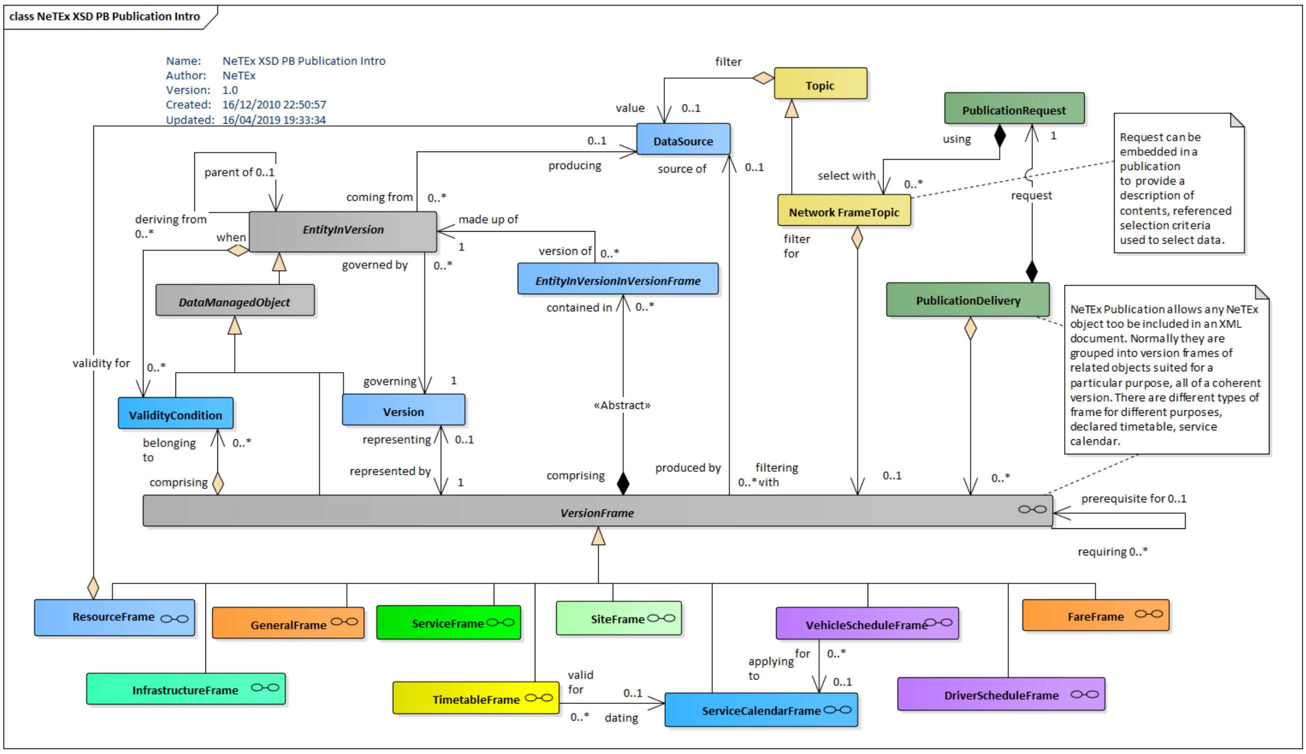


Figure 778 – PublicationDelivery – Physical Model (UML) – Intro

9.3.1.2 NeTEx PublicationDelivery - Details

The ***PublicationRequest*** may specify a variety of filter criteria using NeTEx model elements***.***

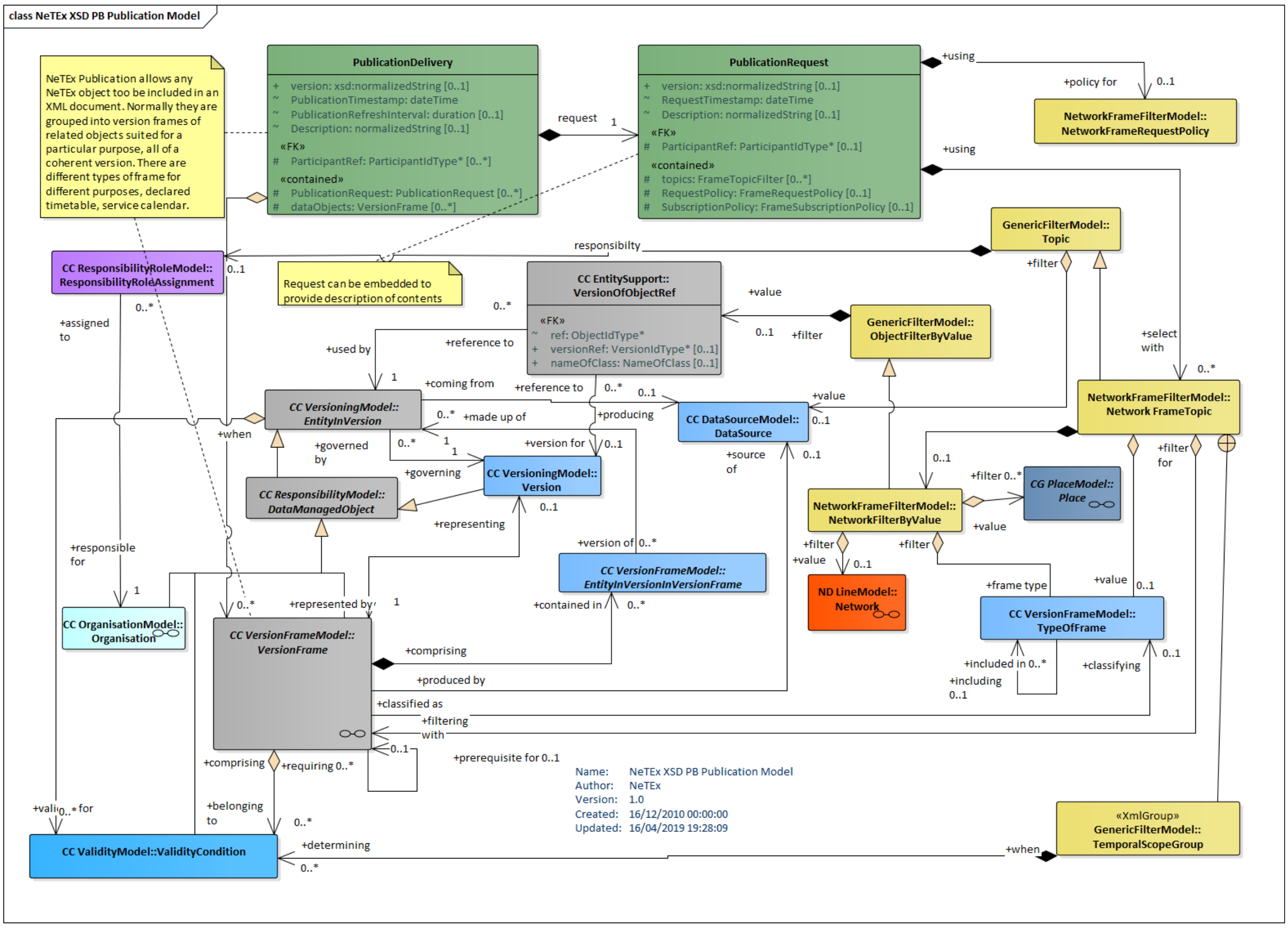


Figure 779 – PublicationDelivery – Physical Model (UML) – Detail

Table 651 –*PublicationDelivery* – Service Element

Classification	Name	Type	Cardinality	Description
	version	*xsd:NMTOKEN*		Version identifier of schema. Fixed.
PublicationHeaderGroup	***PublicationTimestamp***	*xsd:dateTime*	1:1	Time of output of data.
	ParticipantRef	*ParticipantCodeType*	1:1	Identifier of system providing data. Often this will be the same as the DATA SOURCE but it is possible there may be multiple participant systems belonging to a single data source.
	PublicationRequest	*PublicationRequestStructure*	0:1	Echo Request used to create bulk response.
	PublicationRefreshInterval	*xsd:duration*	0:1	How often data in publication is normally refreshed. This is only relevant for regular period batch exports, for example a weekly timetable update.

	Description	*xsd:normalizedString*	0:1	Description of contents.
Payload Group	***dataObjects***	*dataObjects_RelStructure*	0:1	NeTEx VERSION FRAMES making up publication.
	classes	*ClassInRepository*	0:1	Information about classes included in the publication.

The following figure introduces the ***PublicationDelivery*** element.

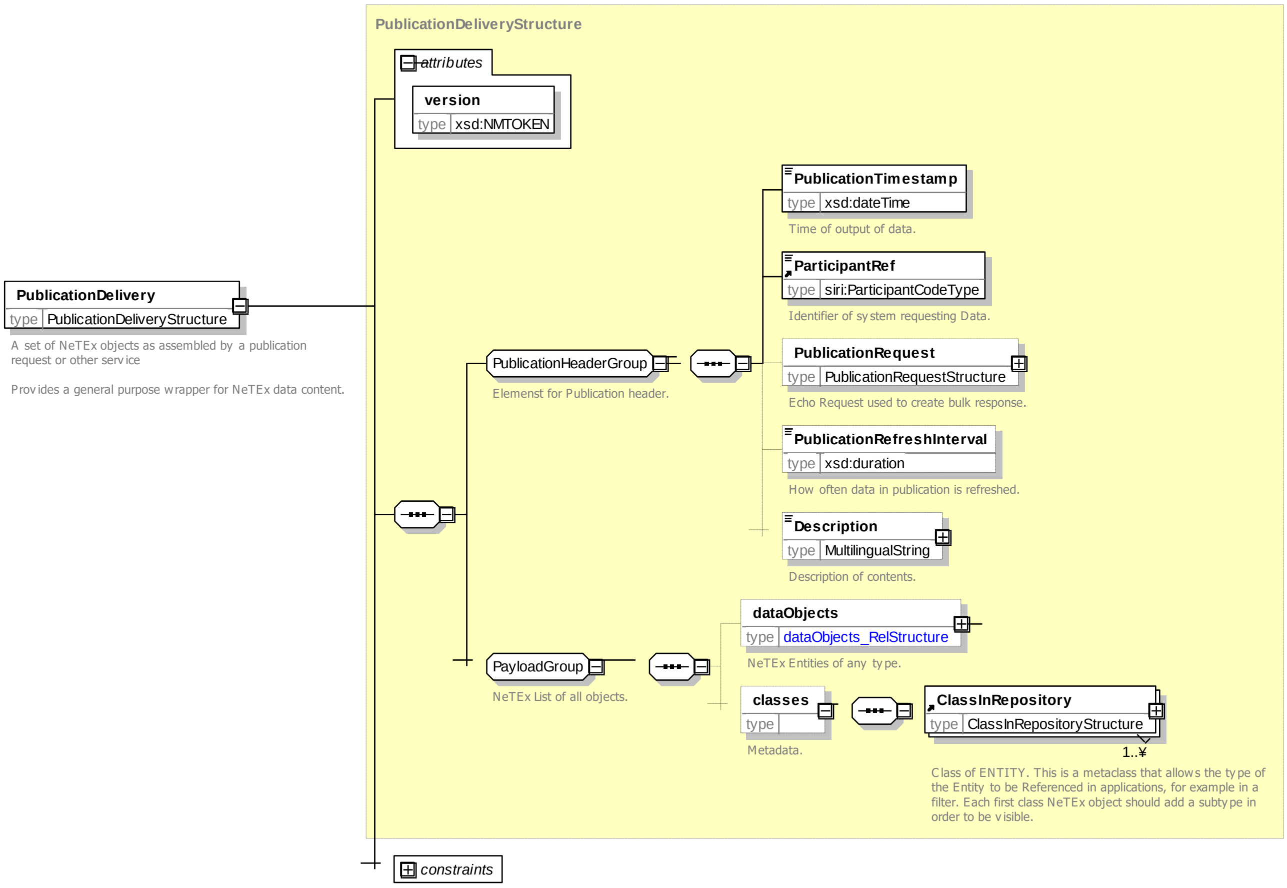


Figure 780 – *Publication Delivery - XSD*

All the elements in a ***PublicationDelivery*** element are contained in a VERSION FRAME. There are a number of different types of VERSION FRAME in NeTEx.

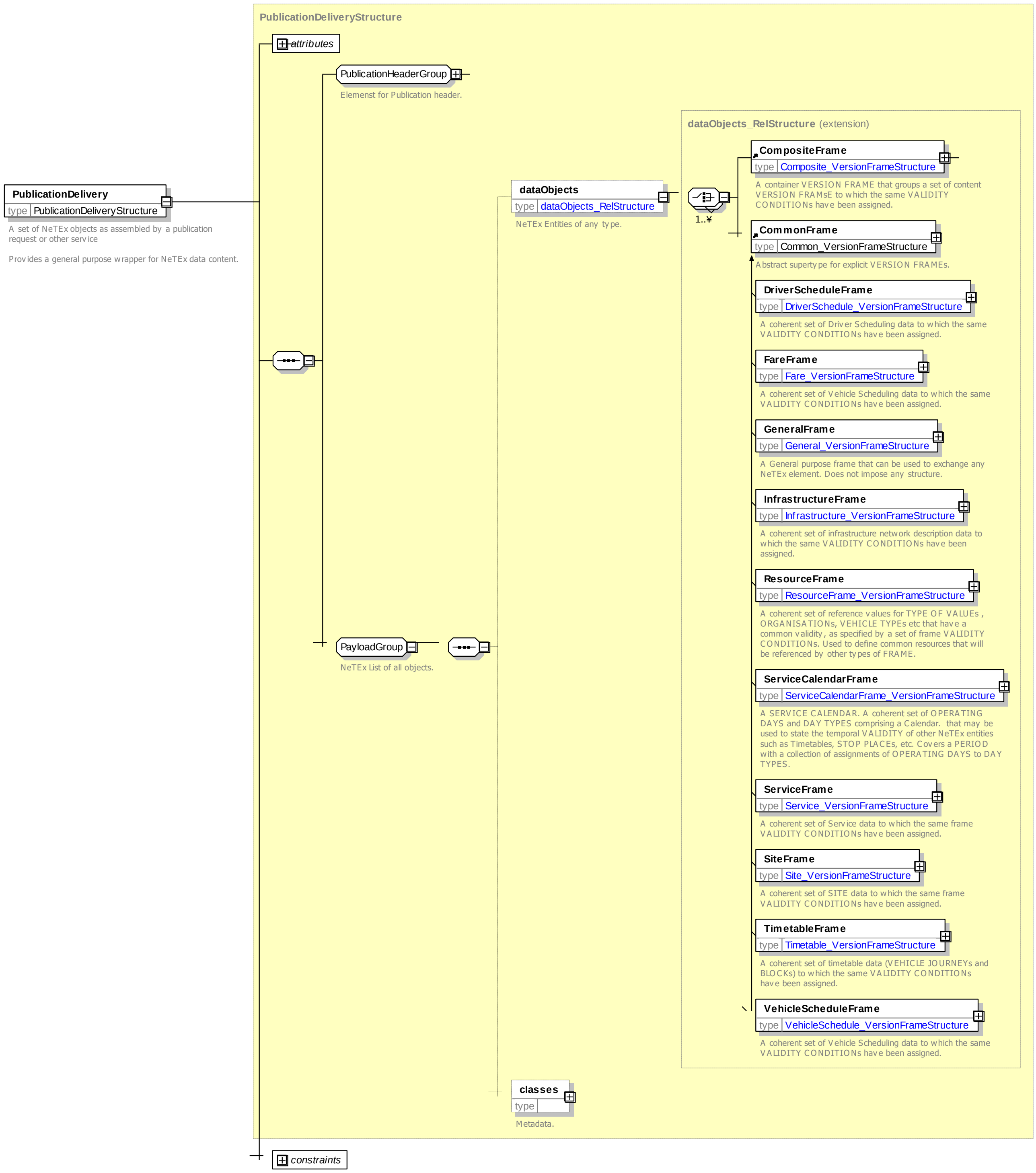


Figure 781 – *Publication Delivery content frames – XSD*

9.3.2 Publication Delivery – Attributes and XSD

9.3.2.1 Publication Request – Service Element

The ***PublicationRequest*** records which NeTEx objects were requested for inclusion in the document along with a timestamp, and an identification of the requestor.

It has two components – the ***topics,*** describing what model values are to be selected by the filter and the ***policies,*** describing how these values are to be processed; for example, the maximum number of returned objects, the level of detail to include, etc.

Table 652 – *PublicationRequest* – Service Element

Classification	Name	Type	Cardinality	Description
	RequestTimestamp	*xsd:dateTime*	1:1	Time of request.
«FK»	***ParticipantRef***	*ParticipantCodeType*	0:1	Identifier of system requesting data.
	Description	*MultiingualString*	0:1	Text description of the filter.
«cntd»	***topics***	*NetworkFrameTopic*	0:*	One or more request filters that describe selection criteria for data to be included in output.
«cntd»	***RequestPolicy***	*NetworkFrameRequest PolicyStructure*	0:1	Policies to apply when fetching data specified by Topics.
«cntd»	***SubscriptionPolicy***	*FrameSubscription-PolicyStructure*	0:1	Policies to apply when subscribing to data specified by Topics.

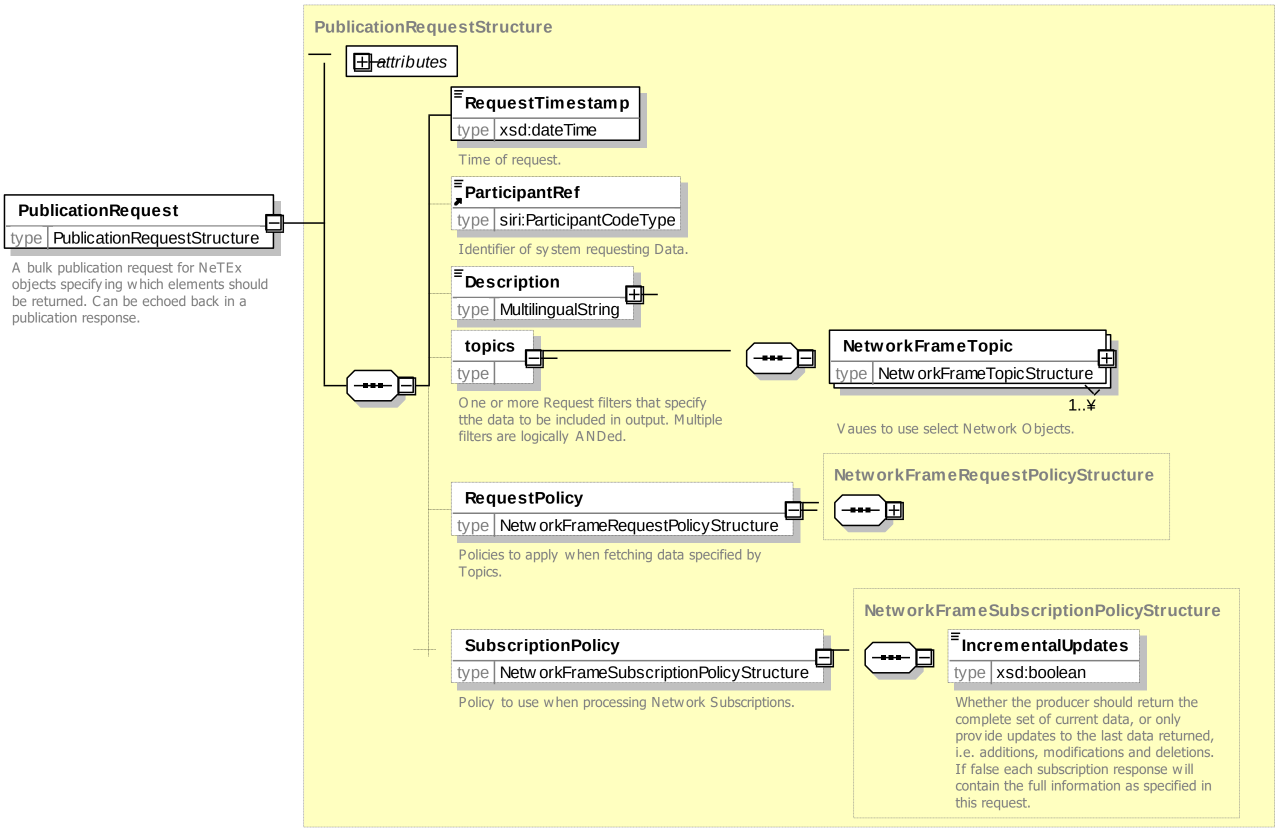


Figure 782 – *Publication Request – XSD*

9.3.2.1.1 *NetworkFrameTopic* – Service Element

The ***NetworkFrameTopic*** specifies the selection criteria for including NeTEx objects in the publication.

It provides several categories of filtering:

- By DATA SOURCE: one or more DATA SOURCEs can be selected (no DATA SOURCE filter means all).
- By temporal information (current, current at, filter on availability conditions, etc.).
- By VALIDITY CONDITIONS, specifying attribute values of specific elements.
- By geographic properties defined by bounding boxes.
- By direct object references to get specific NeTEx objects having a known identifier (the version of the object can also be selected).
- By generic object reference to get all objects of a given class.
- By direct object references to get specific FRAMEs (generic or specific types of VERSION FRAMEs for each functional area of NeTEx, like SITE FRAME, TIMETABLE FRAME, etc.), all the available object in that frame will be returned (with possible additional filters, of course).
- By generic attributes such as NETWORK (all the objects related to a NETWORK), TRANSPORT MODE (elements relating to specific TRANSPORT MODEs), and PLACEs (all the object associated with a given PLACE, such as a STOP PLACE, a SERVICE SITE, a POINT OF INTEREST or a PARKING).

9.3.2.1.2 Using multiple topics

If more than one topic is specified for a filter, the topic values are logically ANDed with all the other topic values. For example if ***DataSource***='123', and ***Operator***="AB' and ***TopographicPlace***='456' then only topics that ***DataSource***, ***Operator*** and

Table 653 – *NetworkFrameTopic* – Service Element

Classification	Name	Type	Cardinality	Description
	Description	*MultiingualString*	0:1	Text description of the filter.
«cntd»	***sources***	*DataSourceRef*	0:*	DATA SOURCE for data.
«FK»	***CodeSpaceRef***	*CodeSpaceRef*	0:1	CODE SPACE for data to select.
«cntd»	***ResponsibilityRole-Assignment***	*ResponsibilityRole-Assignment*	0:1	RESPONSIBILITY ROLE ASSIGNMENT to use or selection criteria for data to be included in output.
XGRP	***TopicTemporalScope-Group***	*Xml_Group*	0:1	Elements specifying temporal conditions for selection.
«cntd»	***TypeOfFrameRef***	*TypeOfFrame-RefStructure*	0:1	Reference to a TYPE OF VERSION FRAME.
		CHOICE		
«FK»	a ***VersionFrameRef***	*VersionFrame-RefStructure*	1:*	Reference to a VERSION FRAME.

«cntd»	b ***NetworkFilterByValue***	*NetworkFilterByValueStructure*	1:0	Values for selecting by value

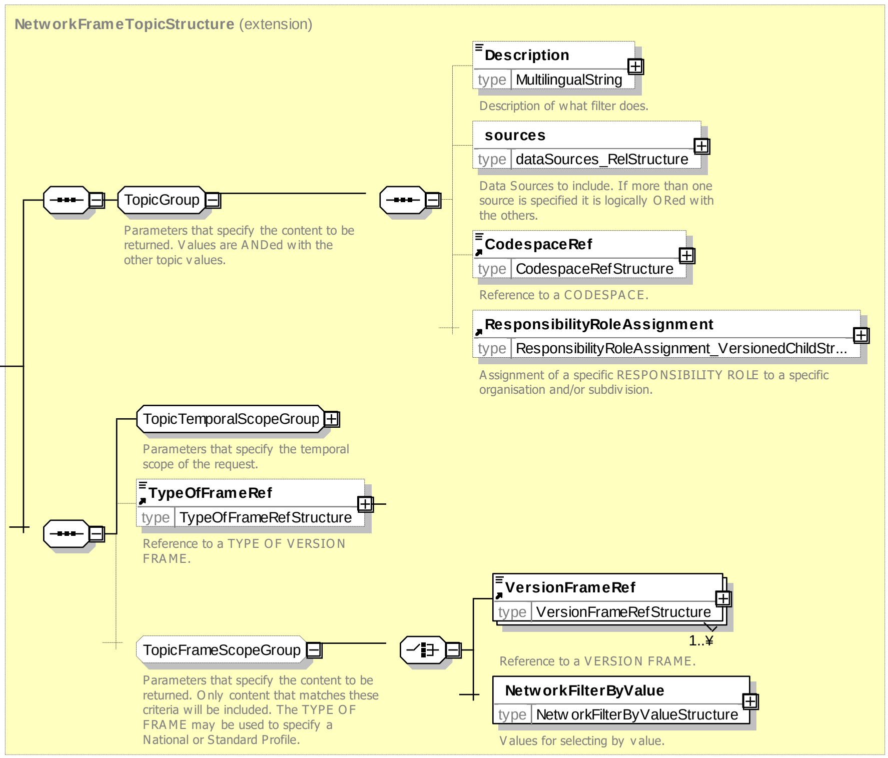


Figure 783 – *Frame Request Topic Structure* – XSD

9.3.2.1.3 *TopicTemporalScopeGroup*

The ***TopicTemporalScopeGroup*** specifies the selection criteria for including NeTEx objects in the publication.

Table 654 – *TopicTemporalScopeGroup – XML Group*

Classification	Name	Type	Cardinality	Description
		CHOICE		
	a ***Current***	*EmptyType*	1:1	Return just the values from the producer current at the time the response is created.
»	b ***ChangedSince***	*dateTime*	1:1	Return the values that have changed since the specified point in time, including current and superseded values. Values that are no longer current will be flagged with the date they were superseded (using the various cleared / deleted /historic dates on different element types).
	c ***CurrentAt***	*dateTime*	1:1	Return just the values that were current at the specified point in time.

«cntd»	d	***HistoricBetween***	*ClosedTimestampRangeStructure*	1:1	Return all values created or updated between the specified period, including current and historic values. Values that are no longer current will be flagged with the date they were superseded.
«cntd»	e	***selectionValidity-Conditions***	*ValidityCondition*	1:*	Validity conditions to apply when selecting data. Applies to frame.

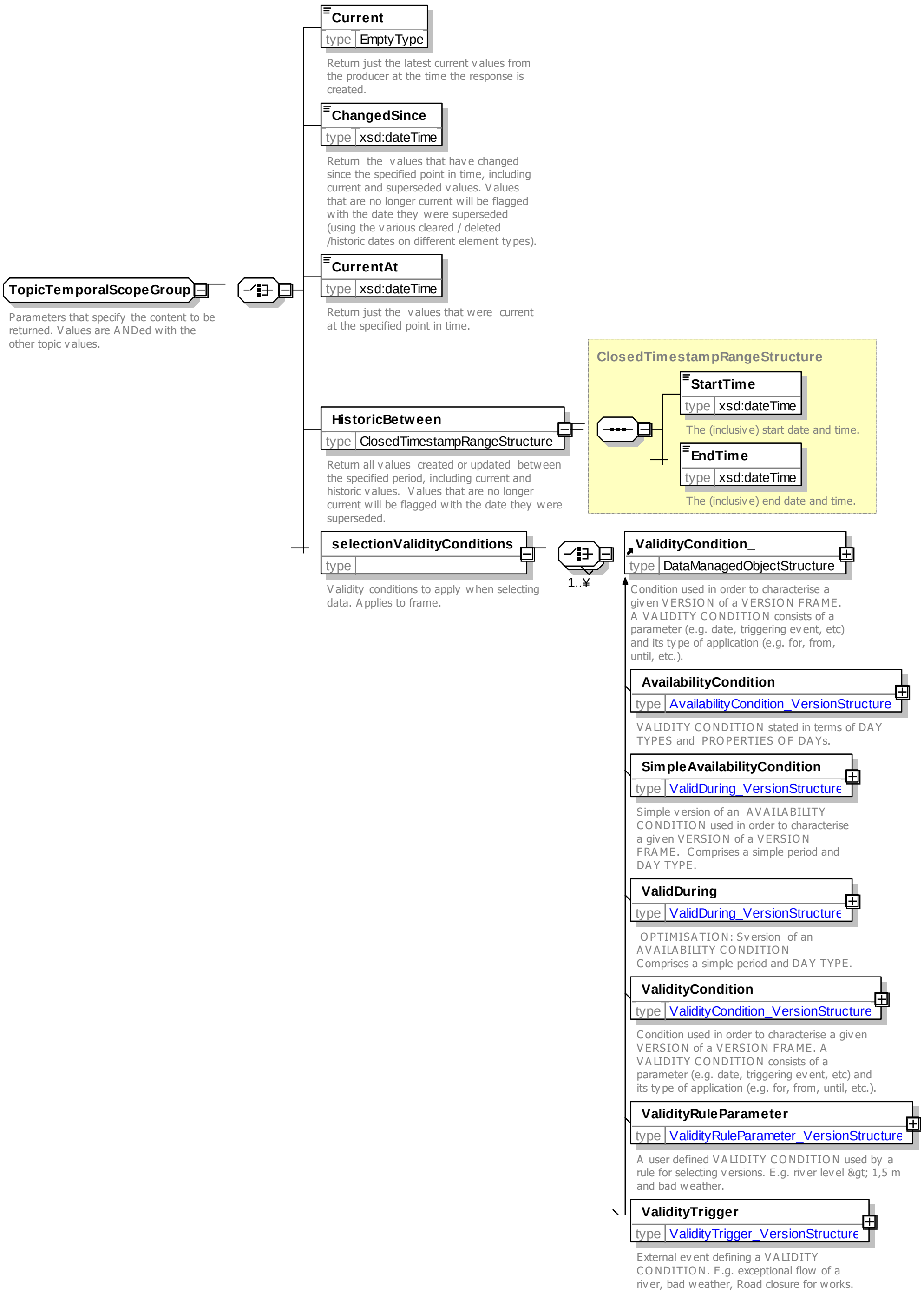


Figure 784 – *TopicTemporalScopeGroup* – XSD

9.3.2.1.4 *NetworkFilterByValue* – Service Element

The ***NetworkFilterByValue*** specifies the selection criteria for including NeTEx objects in the publication.

Table 655 – *NetworkFilterByValue* – Element

Classification	Name	Type	Cardinality	Description
«cFK»	***LayerRef***	*LayerRef*	0:1	Area containing objects to be returned.
«cntd»	***BoundingBox***	*BoundingBoxStructure*	0:1	Area containing objects to be returned.
«cntd»	***objectReferences***	*(VersionOfObjectRef)*	0:*	Reference to ENTITIES to be returned. Use REF with specific value to identify an object. Use REF with blank value to specifiy all object of the object type. Use TypeofValueRef to specify categories of object to be included.
cntd»	***NetworkRef***	*NetworkRefStructure*	0:1	Reference to a NETWORK.
cntd»	***places***	*(PlaceRef)*	0:*	Return all elements conerned with given place.

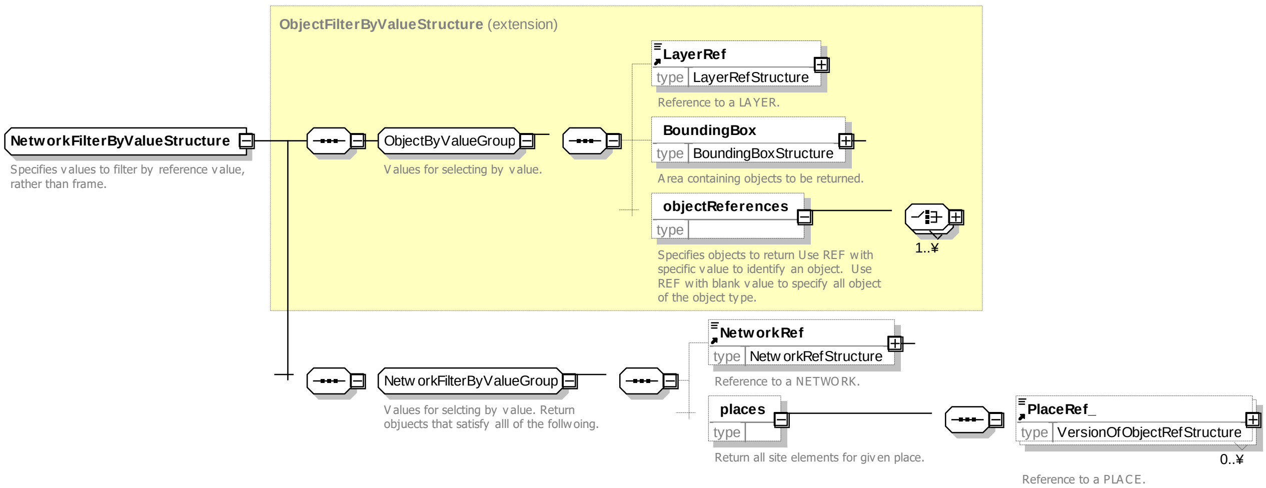


Figure 785 – Frame Request Topic Structure – XSD

9.3.2.2 Frame Request Policy

The ***FrameRequestPolicy*** optionally specifies additional principles to be used when selecting and filtering data.

Table 656 – *FrameRequestPolicy* – Element

Classification	Name	Type	Cardinality	Description
	MaximumNumberOfElements	*xsd:integer*	0:1	Maximum number of objects to include in response. If absent, include all instances that meet the selection criteria.
	IncludeDeleted	*xsd:boolean*	0:1	Whether to include in response elements flagged as deleted. By default this will be '*false'*.
	Urgency	*xsd:duration*	0:1	Allows a requestor to indicate a relative urgency of request. A longer period can be specified for non-urgent requests, e.g. to get historic data. If not specified assume best possible response desired, preferably immediate.
	MustHaveBy	*xsd:dateTime*	0:1	Allows requestor to indicate that if data cannot be supplied by a given time it will not be useful. Can be used to prioritize data on systems working at full load.
	Language	*xsd:language*	0:1	Preferred language to use for text elements in returned data.
«enum»	***RequestDetail***	*OutputDetailEnum*	1:1	Level of detail to return. The default is '*all'*. See allowed values below.

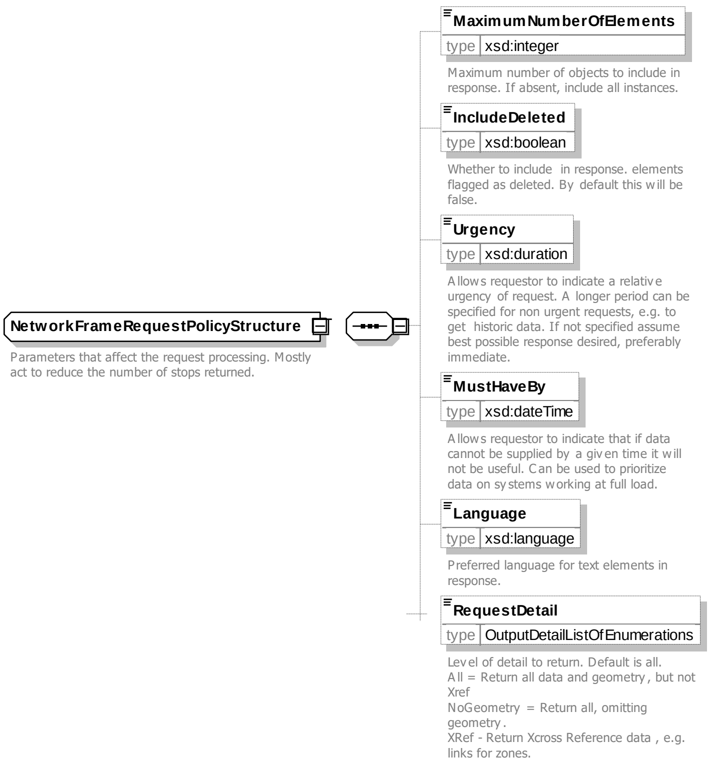


Figure 786 – *Frame Request Policy* – XSD

9.3.2.2.1 RequestDetail – Allowed Values

Allowed values for ***RequestDetail*** (*OutputRequestDetailEnumeration*).

Table 657 — *RequestDetai* – Allowed Values

Value	Description
All	Return all data and geometry, but not Xref.
Basic	Return basic data only.
NoGeometry	Return all, omitting geometry.
XRef	Return cross reference data, e.g. links for zones

9.3.2.3 Frame Request Subscription Policy

The ***FrameRequestSubscription Policy*** optionally specifies additional principles to be used when subscribing to a feed of data.

Table 658 – *FrameRequestPolicy* – Element

Classification	Name	Type	Cardinality	Description
	IncrementalUpdates	*xsd:boolean*	0:1	Whether the producer should return the complete set of current data at the agreed intervals, or only provide updates to the last data returned, i.e. additions, modifications and deletions. If false each subscription response will contain the full information as specified in this request.

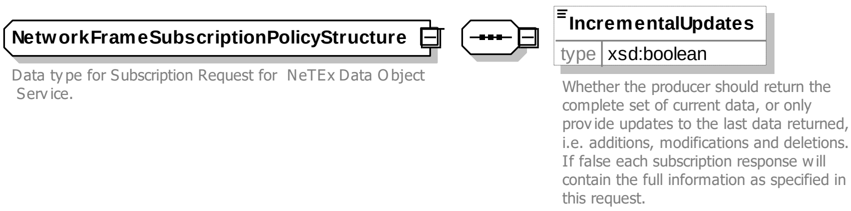


Figure 787 – *NetworkFrameSubscriptionPolicy* – XSD

9.3.3 XML Examples of Publications

9.3.3.1 PublicationDelivery – XML Fragment Example

The following XML fragment shows PublicationDelivery to return the current SCHEDULED STOP POINTs within a specified bounding box.

EXAMPLE

```
<?xml version="1.0" encoding="iso-8859-1"?>
<PublicationDelivery          version="1.0"          xsi:schemaLocation="http://www.netex.org.uk/netex
../../../xsd/NeTEx_publication.xsd"                              xmlns="http://www.netex.org.uk/netex"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
    <PublicationTimestamp>2001-12-17T09:30:47.0Z</PublicationTimestamp>
    <ParticipantRef>SYS001</ParticipantRef>
<!--- ======WHAT WAS REQUESTED ========== -->
    <PublicationRequest version="1.0">
        <RequestTimestamp>2001-12-17T09:30:47.0Z</RequestTimestamp>
```

```
        <ParticipantRef>SYS002</ParticipantRef>
        <filters>
            <FrameRequestFilter>
                <Description>All current STOP POINTS in a specified area</Description>
                <Topics>
                    <Current/>
                    <NetworkFilterByValue>
                        <BoundingBox>
                            <UpperLeft>
                                <Longitude>-0.2071300000</Longitude>
                                <Latitude>51.4217400000</Latitude>
                            </UpperLeft>
                            <LowerRight>
                                <Longitude>-0.2071400000</Longitude>
                                <Latitude>51.4217500000</Latitude>
                            </LowerRight>
                        </BoundingBox>
                        <objectReferences>
                            <ScheduledStopPointRef ref=""/>
                        </objectReferences>
                    </NetworkFilterByValue>
                </Topics>
            </FrameRequestFilter>
        </filters>
    </PublicationRequest>
    <PublicationRefreshInterval>P1M</PublicationRefreshInterval>
    <Description>Monthly stop update example</Description>
    <!--- =============== RESULTS =========== -->
    <dataObjects>
        <ServiceFrame version="any" id="SVF004">
        <!--- ======= CODESPACEs======== -->
            <codespaces>
                <Codespace id="mybus">
                    <Xmlns>mybus</Xmlns>
                    <XmlnsUrl>http://www.mybus.fr/</XmlnsUrl>
                    <Description>My buses</Description>
                </Codespace>
            </codespaces>
            <!--- =======FRAME DEFAULTS======== -->
            <FrameDefaults>
                <DefaultCodespaceRef ref="mybus"/>
            </FrameDefaults>

            <scheduledStopPoints>
                <ScheduledStopPoint version="032 id="SSP0042A">

                    <Name>Poste, St Jean</Name>
                    <Location>
                        <Longitude>-0.2071397147</Longitude>
                        <Latitude>51.4217482061</Latitude>
                    </Location>
                    <VehicleModes>bus</VehicleModes>
                </ScheduledStopPoint>
                <ScheduledStopPoint version="032" id="SNCF0047">
                    <Name>Gare, St Jean</Name>
                    <Location>
                        <Longitude>-0.2071397153</Longitude>
                        <Latitude>51.4217482054</Latitude>
                    </Location>
                    <VehicleModes>rail</VehicleModes>
                </ScheduledStopPoint>
…….
            </scheduledStopPoints>
        </ServiceFrame>
    </dataObjects>
</PublicationDelivery>
```

9.4 NeTEx SIRI-NX services XSD schema

The NeTEx SIRI service provides an http based protocol for exchanging NeTEx content dynamically between servers.

SIRI uses a consistent set of general communication protocols to exchange information between client and server. The same pattern of message exchange may be used to implement different specific **functional interfaces** as sets of concrete message content types. The core SIRI standard defines a number of such functional services, for example SIRI-SM for real-time stop departures, SIRI-VM for real-time vehicle positions, etc.). All SIRI services assume a content model and terminology based on Transmodel.

NeTEx adds an additional functional service **SIRI-NX**, allowing NeTEx and SIRI to share the same communication layer. The service can be used to request reference data in the form of any NeTEx data object. A filtering mechanism allows a client to choose the desired subset of data from a server.

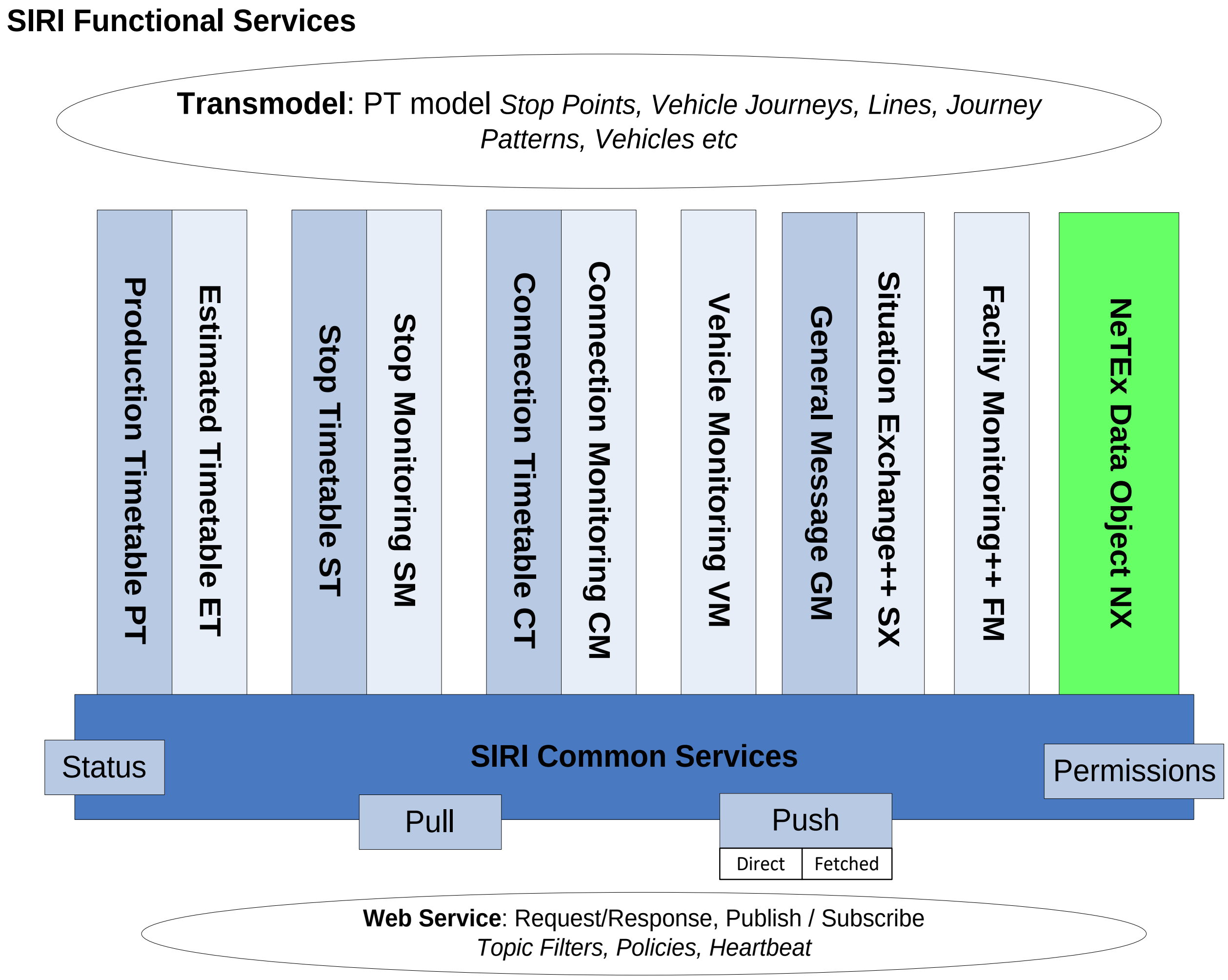


Figure 788 – NeTEx *service as a SIRI functional service*

As NeTEx describes hundreds of different object types it is more practical a single request with powerful filtering mechanism, rather than a large number of specialised requests to fetch different types of NeTEx data. The filters may apply on FRAMEs (generic or specific types of VERSION FRAMEs for each functional area of NeTEx, like SITE FRAME, TIMETABLE FRAME, etc.), on direct object references, on geographic properties, on temporal information and on generic attributes like DATA SOURCE, NETWORK, TRANSPORT MODE, etc. Multiple filters can be combined.

The SIRI-NX services are described by a separate schema that embeds a NeTEx Publication inside a SIRI functional service and that also includes the relevant schemas that define common SIRI functions.

9.4.1 Brief overview of SIRI communication layer

The SIRI communication layer is formally described in SIRI Part 2, EN 15531-2*, 'Service interface for real-time information relating to public transport operations — Part 2: Communications infrastructure*'; refer to this documentation for further details. Here we provide a brief introduction.

9.4.1.1 SIRI interaction patterns

The SIRI protocol is designed to allow the exchange of data between a client system acting as a Consumer of data and a server system acting as a Producer. The protocol comprises a number of messages which are distinct for the Consumer and Producer roles.

SIRI supports two well-known specific patterns of client server interaction for data exchange between Consumer and Producer, Request/Response and Publish/Subscribe.

— **Request/Response** allows for the ad hoc exchange of data on demand from the client. NeTEx uses this pattern.

— **Publish/Subscribe** allows for the repeated asynchronous push of notifications and data to distribute events and Situations detected by a Real-time Service. The initial NeTEx requirements call only for Request/Response exchange and not Publish/Subscribe, but Publish/Subscribe is routinely available through the SIRI communication functions should it be needed in future.

For the delivery of data in responses (to both requests and subscriptions), the SIRI communication layer supports two common patterns of message exchange, as realised in existent national systems:

— A one step '**Direct Delivery'**, as per the classic client-server paradigm, and normal WS-PubSub publish subscribe usage; and;

— A two-step '**Fetched Delivery'** which elaborates the delivery of messages into a sequence of successive messages pairs to first notify the client, and then to send the data when the client is ready. Fetched Delivery is a stateful pattern in its own right.

For a given SIRI Functional Service type (Connection Monitoring, Stop Monitoring etc.), the message payload content is the same regardless of whether information is exchanged with a Request/Response or Publish/Subscribe pattern, or whether it is returned by Direct or Fetched Delivery.

9.4.1.2 SIRI framework messages

The SIRI Communication layer includes a set of messages to manage the communication services:

— **ServiceRequest**: Request from a Consumer to a Producer for a data. Embeds a Functional service.

— **SubscriptionRequest**: Request from a Consumer to a Producer for a subscription. Embeds a Functional service.

— ***TerminateSubscription***: Request from Subscriber to Subscription Manager to terminate a subscription.

— ***DataReady***: Notification from Producer to Consumer that data is ready.

— ***DataSupply***: Request from Consumer to Producer to fetch update previously notified by a Data ready message.

— ***CheckStatus***: Request from Consumer to Producer to check whether service is working.

— ***Heartbeat***: Notification from Producer to Consumer that system is working.

9.4.1.3 SIRI Endpoint Addresses

For each different SIRI Functional Service, SIRI identifies different Logical Service endpoints, which may be distinct addresses, or all be mapped to the same concrete URI. The following table shows the endpoints for server functions. These addresses can be configured in the ***ServiceRequestContext***.

Table 659 – *SIRI* Producer Logical Endpoints

Server Logical Endpoint Name	Messages sent to endpoint	Description
[CheckStatus]	***CheckStatusRequest***	Address to which send requests to check that the Functional Service is available.
[Subscribe]	***SubscriptionRequest***	Address to which send requests for new subscriptions to the Functional Service.
[ManageSubscriptions]	***TerminateSubscription***	Address to which send requests to change or delete subscriptions to the Functional Service. Normally the same as *Subscribe.*
[GetData]	***ServiceRequest***	Address to which send requests to fetch data, and confirmation of successful receipt.
	DataReadyResponse	
	DataSupplyRequest	
	DataReceivedAcknowledgement	

The following table shows the endpoints to access client functions, that is, the addresses to which different types of responses to client requests will be sent. If not explicitly specified, the response address may be taken from the http request.

Table 660 – *SIRI* Consumer Logical Endpoints

Client Logical Endpoint Name	Messages sent to endpoint	Description
[ReportStatus]	***CheckStatusResponse***	Address to which send responses to inform the Consumer that the Functional Service is available.
	HeartBeat	
[Subscriber]	***SubscriptionResponse***	Address to which send responses to requests to create, change or delete subscriptions.
	TerminateSubscriptionResponse	
[Notify]	***DataReadyNotification***	Address to which send notifications of data being ready.
[Consumer]	***ServiceDelivery***	Address to which send data.

9.4.1.4 SIRI error handling

The SIRI Communication protocols are designed to fail gracefully, so that a Consumer service can detect failure and recover. Error codes and other considerations for resilience and recovery are covered in the SIRI documentation.

9.4.2 SIRI ServiceRequest wrapper

Request/Response is the simplest pattern of SIRI interaction. For data exchange, the requestor sends a Service Request to a Specific SIRI Functional Service as located by the Service's Endpoint and is returned an immediate data delivery, i.e. a response message that contains application payload data.

Each specific SIRI Functional Service Request is wrapped within a general ***ServiceRequest*** element, and the corresponding delivery is similarly wrapped within a ***ServiceDelivery*** element. There is a different SIRI

Functional Service Request message type for each different SIRI Functional Service, and also a distinct SIRI Functional Service Delivery message with which to return the content for the individual service.

9.4.2.1 SIRI ServiceRequest Message — Element

The ***ServiceRequest*** is sent to the [*GetData*] endpoint of a SIRI Functional Service. The following table shows the common parameters that may be specified on a ***ServiceRequest***.

Each request may contain Endpoint information, including Endpoint Reference Properties. For Direct delivery the endpoint address is the [*Consumer*] endpoint to which the data is sent. For fetched Delivery it is the [*Notify*] endpoint to which the data ready notification is to be sent.

The Endpoint properties may include:

— The *Participant Reference*, which will be unique to the requestor in communication between the two parties

— A *Message Identifier,* with which to reference the specific request message in subsequent conversations, and which will be unique within the scope of the SIRI Functional Service Type and the Participant scope.

Note that both the ***ServiceRequest*** and the concrete Functional Service Requests contained within it can have their own specific message references.

If SIRI Access Controls are supported, the Participant Reference is used to determine if the Requestor is authorised to make the request for which it has asked. The Reference will be checked against the Access Matrix, and if the permitted access is exceeded, the ***NotAuthorised*** error condition will be returned.

Table 661 – *SIRI ServiceRequest* — Attributes

ServiceRequest			*+Structure*	Request from a Consumer to a Producer for immediate delivery of data. Answered with a ***ServiceDelivery****.* (For *Fetched Delivery* this will be after a further ***DataReadyRequest***.)
	ServiceRequestContext	0:1	*+Structure*	General request properties – typically configured rather than repeated on request.
log	***RequestTimestamp***	**1:1**	*xsd:dateTime*	Timestamp on request.
Auth.	***AccountId***	0:1	*+Structure*	Account Identifier. May be used to attribute requests to a specific user account for authentication or reporting purposes +SIRI v3.0
	AccountKey	0:1	*+Structure*	Authentication key for request. May be used to authenticate the request to ensure the user is a registered client. +SIRI v3.0
Endpoint Properties	***Address***	0:1	*EndpointAddress*	Address to which response is to be sent: [***Notify***] endpoint. If omitted, this may also be determined from ***RequestorRef*** and preconfigured data, or the http request.

	RequestorRef	**1:1**	*→Participant Code*	Identifier of Requestor. May be used to identify an individual participant system or individual device client. If used for a device client should be an anonymous token, divulged with user consent.
	MessageIdentifier	0:1	*MessageQualifier*	Arbitrary identifier that may be given to message.
Delegator Endpoint	***DelegatorAddress***	0:1	*EndpointAddress*	Address of originated system to which delegated response is to be returned If request has been proxied by an intermediate aggregating system this provides tracking information relating to the original requestor. This allows the aggregation to be stateless.
	DelegatorRef	0:1	*→Participant Code*	Identifier of delegating system that originated message.
	Concrete service subscription			If more than one, must all be same type.
Payload	*a* ***DataObjectRequest***	**-1:***	*+Structure*	Data Object Request

9.4.2.2 SIRI ServiceDelivery Message — Element

Delivery responses are sent to the *[Consumer]* endpoint for the request. The delivery may be received as a single step *Direct Delivery* or as the last step of a *Fetched Delivery*.

Each Delivery comprises a general ***ServiceDelivery*** message, containing a one or more SIRI Functional Service delivery responses, for example ***DataObjectDelivery***, ***StopMonitoringDelivery***.

9.4.3 SIRI ServiceDelivery

9.4.3.1 SIRI ServiceDelivery— Element

The ***ServiceDelivery*** contains any general parameters that are common to all delivery types.

Table 662 – *SIRI ServiceDelivery* — Attributes

ServiceDelivery			*+Structure*	Response from Producer to Consumer to deliver payload data. Either answers a direct ServiceRequest, or satisfies a subscription asynchronously. May be sent directly in one step, or be fetched in response to a Data Supply Request.
Attributes	***srsName***	0:1	*xsd:string*	Default GML coordinate format for any spatial points defined in response by Coordinates parameter.
Log	***ResponseTimestamp***	**1:1**	*xsd:dateTime*	Time individual response element was created.
Endpoint properties	***ProducerRef***	0:1	*→Participant Code*	Participant reference that identifies producer of data. May be available from context.
	Address	0:1	*EndpointAddress*	Address to which any acknowledgment should be sent. Only needed if ***ConfirmDelivery*** specified.
	ResponseMessageIdentifier	0:1	*MessageQualifier*	An arbitrary unique reference associated with the response which may be used to reference it.
	RequestMessageRef	0:1	*→Message Qualifier*	Reference to a unique message identifier associated with the request which gave rise to this response.

Delegator endpoint	***DelegatorAddress***		0:1	*EndpointAddress*	Address of originated system to which delegated response is to be returned. If request has been proxied by an intermediate aggregating system this provides tracking information relating to the original requestor. This allows the aggregation itself to be stateless.
	DelegatorRef		0:1	→*ParticipantCode*	Identifier of delegating system that originated message
Status	***Status***		0:1	*xsd:boolean*	Whether the complete request could be processed successfully or not. The default is '*true'*. If any of the individual requests within the delivery failed, should be set to *false*.
	ErrorCondition		0:1	*See below*	Description of any error or warning conditions that applies to the overall request. More Specific error conditions should be included in the error conditions attached to each functional service response that fails.
	a	***CapabilityNotSupportedError***	**1:1**	*+Error*	Capability not supported.
	b	***OtherError***		*+Error*	Error other than a well-defined category.
		Description	0:1	→*ErrorDescription*	Description of Error.
	MoreData		0:1	*xsd:boolean*	Whether there are more delivery messages making up this data supply group. The default is '*false'*. Optional SIRI Capability: *MultipartDespatch*.
Payload	*Concrete SIRI Service:*				One or more of a single type of the following:
	a	***DataObjectDelivery***	0:*	*+Structure*	See SIRI Part 3 – Production Timetable.

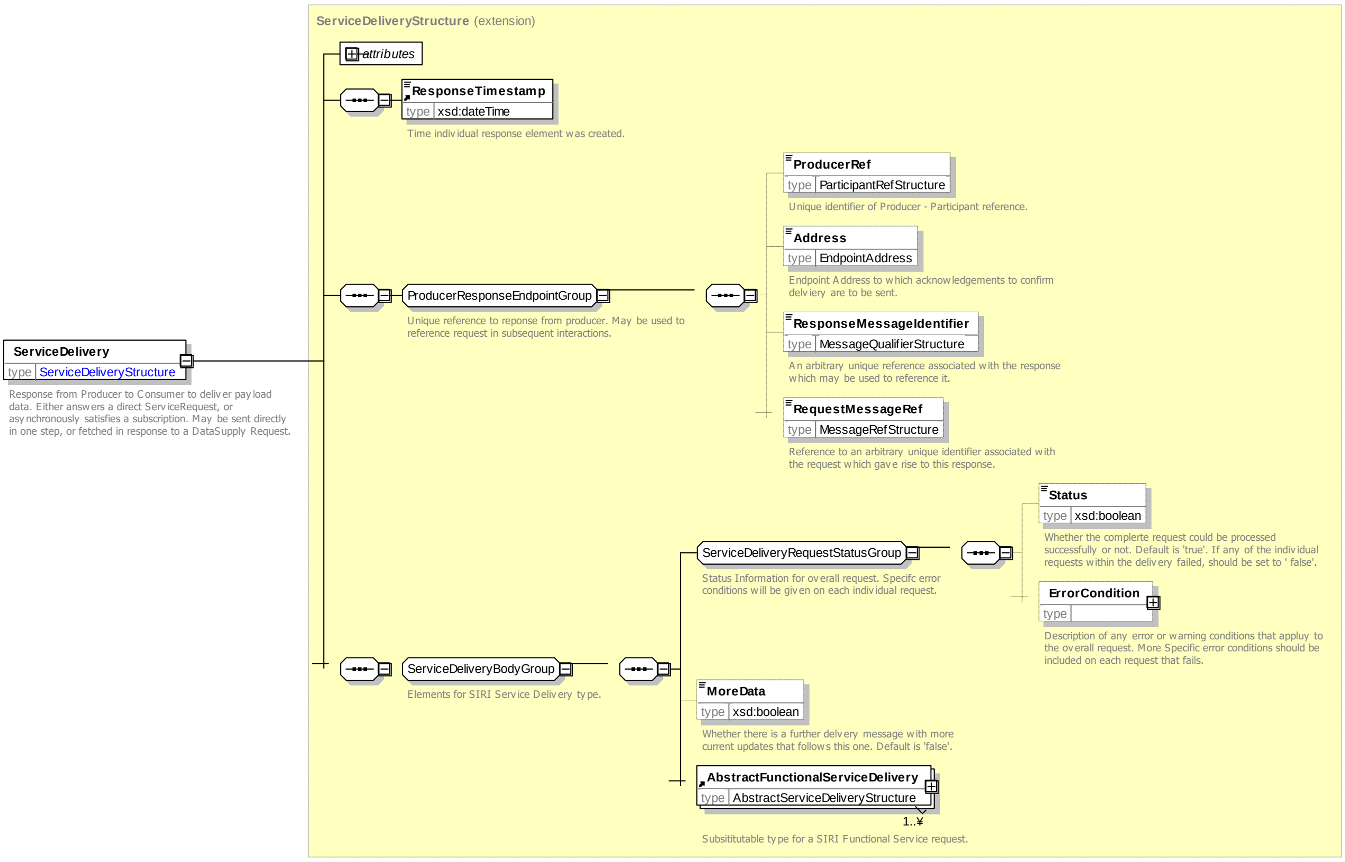


Figure 789 – *SIRI ServiceDelivery — XSD*

9.4.3.2 Common Properties of SIRI Functional Service Delivery Messages

All the individual SIRI Functional Service delivery message types, (for example ***DataObjectDelivery***, ***VehicleMonitoringDelivery***), etc., are specialisation of ***AbstractFunctionalServiceDelivery*** and have a number of common elements – see the following table.

Table 663 – *SIRI AbstractFunctionalServiceDelivery— Common* Attributes

xxxDelivery			*+Structure*	Delivery for xxx Service
Log	***ResponseTimestamp***	**1:1**	*xsd:dateTime*	Time individual response element was created.
Endpoint properties	***RequestMessageRef***	0:1	*→MessageQualifier*	For direct requests, Identifier of request that this Delivery satisfies.
	SubscriberRef	0:1	*→ParticipantCode*	Required if Delivery is for a Subscription, Participant Reference of Subscriber.
	SubscriptionFilterRef	0:1	*→Subcription FilterCode*	Unique identifier of Subscription filter to which this subscription is assigned. If there is only a single filter, then can be omitted.
	SubscriptionRef	**1:1**	*→Subscription Qualifier*	Required if Delivery is for a Subscription, Identifier of Subscription issued by Requestor. Unique within Subscriber (i.e. within ***ParticipantRef*** of Subscriber), and SIRI Functional Service type.
Delegation	***DelegatorAddress***	0:1	*Xsd:anyURI*	Address of original Consumer, i.e. requesting system to which delegating response is to be returned. +SIRI 3.0
	DelegatorRef	0:1	*→ParticipantCode*	Identifier of delegating system that originated message. +SIRI 3.0
Status	***Status***	0:1	*xsd:boolean*	Whether the complete request could be processed successfully or not. The default is '*true*'. If any of the individual requests within the delivery failed, should be set to *false*.

	ErrorCondition		0:1	*+Structure*	Description of any error or warning conditions that apply to the specific functional request or response.
				CHOICE	One of the following Error codes.
	a	***CapabilityNotSupportedError***	-1:1	*+ Error*	Error: Capability not supported.
	b	***AccessNotAllowedError***		*+Error*	Error: Requestor is not authorised to the service or data requested.
	c	***NoInfoForTopicError***		*+Error*	Error: Valid request was made but service does not hold any data for the requested topic expression.
	d	***AllowedResourceUsageExceededError***		*+Error*	Error: Valid request was made but request would exceed the permitted resource usage of the client.
	e	***OtherError***		*+Error*	Error other than a well-defined category.
		Description	0:1	*→ErrorDescription*	Description of Error.
	ValidUntil		0:1	*xsd:dateTime*	End of data horizon of the data producer.
	ShortestPossibleCycle		0:1	*PositiveDurationType*	Minimum interval at which updates can be sent.
	DefaultLanguage			*X xsd:language*	Default language for text elements.
Payload	{Content Specific to Functional Service type. see ***DataObjectDelivery*** below}				
any	***Extensions***		0:1	*any*	Placeholder for user extensions.

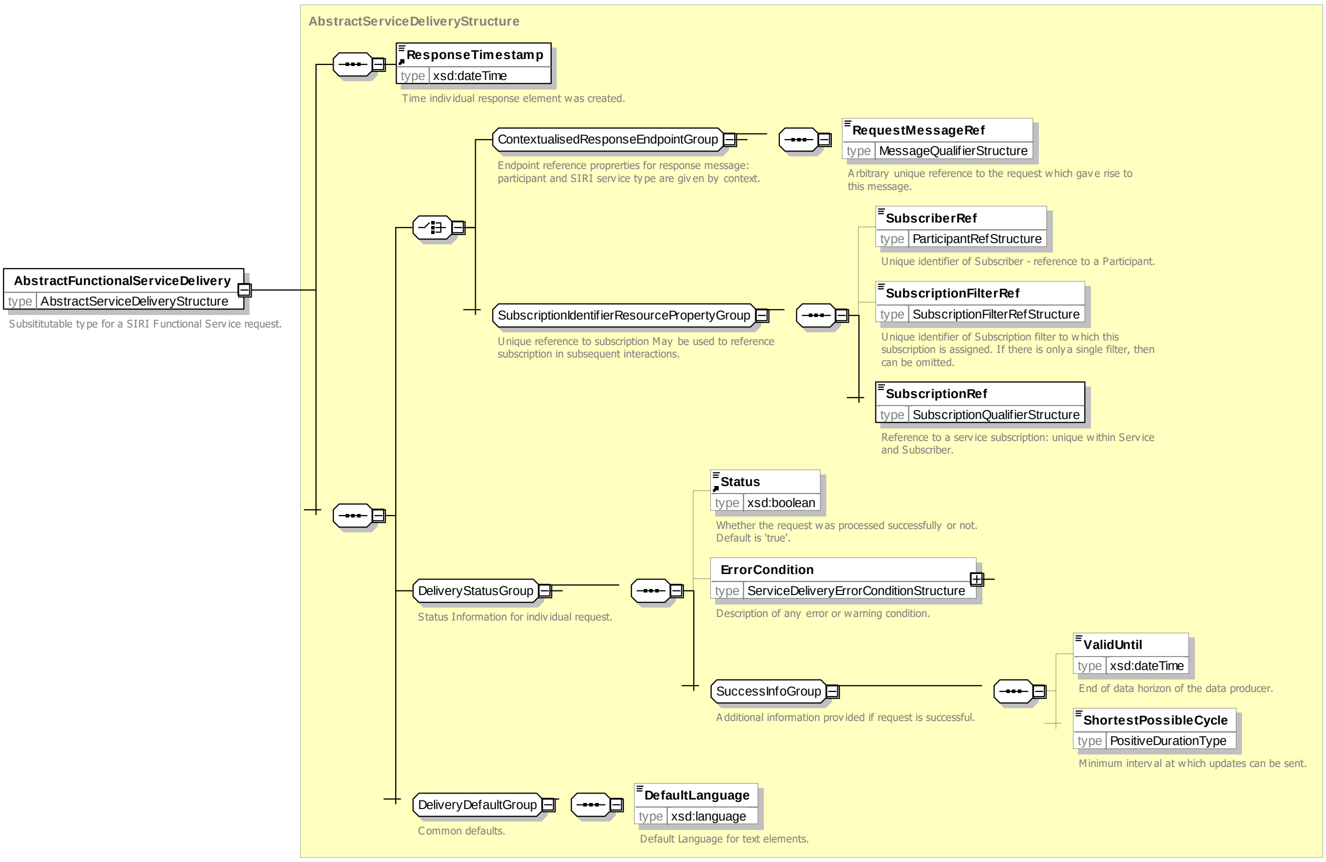


Figure 790 – *SIRI AbstractFunctionalServiceDelivery — XSD*

9.4.4 Data Object Service [SIRI-NX]

9.4.4.1 Purpose

The SIRI Data Object service is used to request reference data from a NeTEx service. Any data from the NeTEx content model may be requested.

The Data Object Service comprises the ***DataObjectRequest*** message used to specify the desired contents of a request, and the ***DataObjectDelivery*** message used to deliver the response. The ***DataObjectSubscription*** message allows a subscriber to request asynchronous updates for the service: it contains an embedded ***DataObjectRequest***.

The ***DataObjectDelivery*** returns information organized into VERSION FRAMEs within a NeTEx ***PublicationDelivery***.

9.4.4.2 Capability and Permission Matrices

9.4.4.2.1 Capability Matrix

The following set of required and optional capabilities is defined for the Data Object service. If the service supports Capability Discovery, the ***DataObjectCapabilitiesRequest*** / ***DataObjectCapabilitiesResponse*** message pair can be used to determine the implementation's capabilities.

Table 664 – *NeTEx DataObjectCapabilities Matrix*

DataObjectCapabilities					*+Structure*	Capabilities describing implementation of Data Object service.
inherit	***::>***			**1:1**	*See xxxCapabilityResponse*	Inherits from ***AbstractSiriFunctionalServicDelivery*** See SIRI Part 2 for Common Capability attributes.
Topic	***TopicFiltering***			0:1	*+Structure*	Which optional filtering features are supported?
		classes		**1:***	*ClassRef*	Classes supported by services
Request Policy	***RequestPolicy***			0:1	*+Structure*	Which optional features of the Request Policy are supported by the service?
		NationalLanguage		**1:***	*xsd:language*	National languages used by service.
		Translations		0:1	*xsd:boolean*	Whether the producer supports translations. SIRI 3.0. The default is '*false'*.
		Coordinates			CHOICE	Location coordinate system for results.
			GMlCoordinateFormat	0:1	*SrSNameType*	Use GML format
			WgsDecimalDegrees	0:1	*EmptyType*	Default coordinate data system is WGS 84 latitude and longitude.
Access Control	***AccessControl***			0:1	*+Structure*	Which optional Access Control features are supported by service?
		RequestChecking		0:1	*xsd:boolean*	Whether access control of requests is supported. The default is *'false'*.
any	***Extensions***			0:1	xsd:any*	Placeholder for user extensions.

9.4.4.2.2 Permission Matrix

If the implementation supports both Capability Discovery and Access Controls, then the ***DataObjectCapabilitiesResponse*** message can include the access permissions for the requestor participant to access data.

Table 665 – *NeTEx DataObjectPermission Matrix*

DataObjectPermission			*+Structure*	Permissions to use implementation of Data Object service.
Inherit	***::>***	**1:1**	*xxxServicePermissions*	See SIRI Part 2 for Common Permission elements.
Topic	***FramePermissions***	0:1	*+Structure*	Frame permissions for participant.

9.4.4.3 The DataObjectRequest

9.4.4.3.1 DataObjectRequest — Element

The ***DataObjectRequest*** can be used in both a direct request, and for a subscription.

The *Topic* term are defined by a FrameRequestFilter – See earlier.

Table 666 – *NeTEx DataObjectRequest — Service Element*

DataObjectRequest			*+Structure*	Request for information about messages.
Attributes	***version***	**1:1**	*VersionString*	Version identifier of Data Object Service.
Endpoint Properties	***RequestTimestamp***	**1:1**	*xsd:dateTime*	See SIRI Part 2 Common properties of SIRI Functional Service Requests.
	MessageIdentifier	0:1	*MessageQualifier*	
Topic	***filters***	0:*	*FrameRequestFilter*	Filter specifying which data is to be included. See PublicationDelivery earlier.
any	***Extensions***	0:1	*xsd:any**	Placeholder for user extensions.

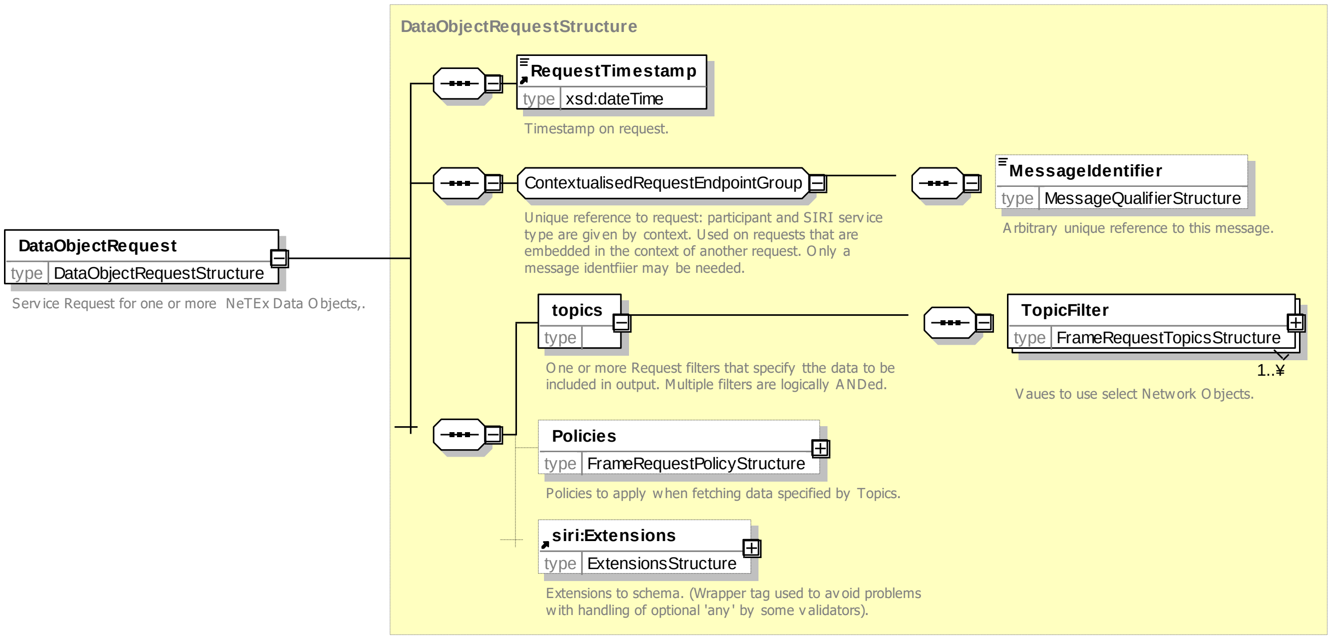


Figure 791 – *NeTEx DataObjectRequest — XSD*

9.4.4.3.2 DataObjectRequest — Example

The following is an example of a ***DataObjectRequest*** to fetch a set of stop data referenced by a known Site frame.

EXAMPLE

```
<siri:SiriRequest>
        <siri:ServiceRequest>
            <!--=======GENERAL CONTEXT=====================================-->
            <siri:ServiceRequestContext>
                <siri:RequestTimeout>P1Y2M3DT10H30M</siri:RequestTimeout>
                <siri:DeliveryMethod>direct</siri:DeliveryMethod>
            </siri:ServiceRequestContext>
            <!--======ENDPOINT REFERENCES====================-->
            <siri:RequestTimestamp>2004-12-17T09:30:47-05:00</siri:RequestTimestamp>
            <siri:RequestorRef>NADER</siri:RequestorRef>
            <siri:MessageIdentifier>mymsg122</siri:MessageIdentifier>
            <DataObjectRequest>
                <siri:RequestTimestamp>2001-12-17T09:30:47.0Z</siri:RequestTimestamp>
                <siri:MessageIdentifier>mymsg12201</siri:MessageIdentifier>
                <topics>
                    <FrameTopicFilter>
                        <Current/>
                        <SiteFrameRef ref="abc:SiteFrame:1234">REQUEST</SiteFrameRef>
                    </FrameTopicFilter>
                </topics>
                <Policy>
                    <Language>en</Language>
                </Policy>
            </DataObjectRequest>
        </siri:ServiceRequest>
</siri:SiriRequest>
```

9.4.4.4 The DataObjectSubscriptionRequest

9.4.4.4.1 DataObjectSubscriptionRequest — Element

The ***DataObjectSubscriptionRequest*** requests the asynchronous delivery of the information described by a ***DataObjectRequest***.

Table 667 – *NeTEx DataObjectSubscriptionRequest* — Attributes

DataObjectSubscriptionRequest			*+Structure*	Request for a subscription to the SIRI ***DataObject*** Service.
Identity	***SubscriberRef***	0:1	*→ParticipantCode*	See SIRI Part 2 Common ***SubscriptionRequest*** parameters.
	SubscriptionIdentifier	**1:1**	*SubscriptionQualifier*	
Lease	***InitialTerminationTime***	**1:1**	*xsd:dateTime*	
Reque st	***DataObjectRequest***	**1:1**	*+Structure*	See ***DataObjectRequest***.
	Extensions	0:1	xsd:any*	Placeholder for user extensions.

9.4.4.4.2 DataObjectSubscriptionRequest — Example

The following is an example of a ***DataObjectSubscriptionRequest****.* This example subscribes to receive STOP PLACE and TYPE OF PLACE data for a given NETWORK '*Mynnet123'*.

EXAMPLE

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- (C) Copyright 2005, 2007 SIRI -->
```

```xml
<siri:Siri xsi:schemaLocation="http://www.siri.org.uk/siri ../../../xsd/NeTEx_siri.xsd" xmlns:siri="http://www.siri.org.uk/siri" xmlns="http://www.netex.org.uk/netex" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <siri:SiriRequest>
        <siri:SubscriptionRequest>
            <siri:RequestTimestamp>2004-12-17T09:30:47-05:00</siri:RequestTimestamp>
            <siri:RequestorRef>NADER</siri:RequestorRef>
            <DataObjectSubscriptionRequest>
                <siri:SubscriberRef>NADER</siri:SubscriberRef>
                <siri:SubscriptionIdentifier>Sub1234</siri:SubscriptionIdentifier>
                <siri:InitialTerminationTime>2001-12-17T09:30:47.0Z</siri:InitialTerminationTime>
                <DataObjectRequest>
                    <siri:RequestTimestamp>2004-12-17T09:30:47-05:00</siri:RequestTimestamp>
                    <topics>
                        <FrameTopicFilter>
                            <Current/>
                            <NetworkFilterByValue>
                                <objectReferences>
                                    <StopPlaceRef ref=""/>
                                    <TypeOfPlaceRef ref=""/>
                                    <NetworkRef ref="Mynnet123">REQUEST</NetworkRef>
                                </objectReferences>
                            </NetworkFilterByValue>
                        </FrameTopicFilter>
                    </topics>
                </DataObjectRequest>
                <SubscriptionPolicy>
                    <IncrementalUpdates>true</IncrementalUpdates>
                </SubscriptionPolicy>
            </DataObjectSubscriptionRequest>
        </siri:SubscriptionRequest>
    </siri:SiriRequest>
</siri:Siri>
```


9.4.4.5 The DataObjectDelivery

9.4.4.5.1 Introduction

The ***DataObjectDelivery*** returns one or more VERSION Frames containing reference data.

9.4.4.5.2 ServiceDelivery with a DataObjectDelivery

One or more ***DataObjectDelivery*** elements may be returned as part of a SIRI ***ServiceDelivery***, with a common ***ResponseTimestamp***.

Table 668 – *NeTEx ServiceDelivery / DataObjectDelivery* — Attributes

ServiceDelivery			*+Structure*	See SIRI Part 2 ***ServiceDelivery***
HEADER	*::>*	**1:1**	See ServiceDeliver y	
Payload	***DataObjectDelivery***	**1:***	*DataObjectDeli very*	See ***DataObjectDelivery*** element.

9.4.4.5.3 DataObjectDelivery — Element

A ***DataObjectDelivery*** contains zero, one or NeTEx dataObjects, grouped in Version Frames.

Table 669 – *NeTEx DataObjectDelivery* — Attributes

DataObjectDelivery			*+Structure*	Message content & changes to messages.
Attribute s	***version***	**1:1**	*VersionString*	Version identifier of ***DataObject*** Service. Fixed e.g. '1.1a'.

LEADE R	::>	**1:1**	xxx*ServiceDelive ry*	Inherits from AbstractServiceDelivery See SIRI Part 2 xxx***ServiceDelivery***.
Payload	***dataObjects***	0:*	*VersionFrame*	A VERSION FRAME containing NeTEx data objects, that is instances of specialisations of ENTITY IN VERSION that are coherent with the other instances in the frame. See NeTEx Part 1, 2 & 3 for further details
	classes	0:*	*ClassInRepositor y*	Used to return metadata about the classes available in the repository.

9.4.4.5.4 DataObjectDelivery — Example

The following is an example of a ***DataObjectDelivery*** which returns a SERVICE FRAME with a single SCHEDULED STOP POINT instance in it.

EXAMPLE

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- (C) Copyright 2005, 2007 SIRI -->
<siri:Siri        xsi:schemaLocation="http://www.siri.org.uk/siri        ../../../xsd/NeTEx_siri.xsd"
xmlns:siri="http://www.siri.org.uk/siri"                             xmlns="http://www.netex.org.uk/netex"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0">
<siri:SiriResponse>
    <siri:ServiceDelivery>
        <!--=======HEADER================================================= -->
        <siri:ResponseTimestamp>2004-12-17T09:30:46-05:00</siri:ResponseTimestamp>
        <siri:ProducerRef>KUBRICK</siri:ProducerRef>
        <siri:Status>true</siri:Status>
        <siri:MoreData>false</siri:MoreData>
        <!--=======PAYLOAD============================================== -->
        <DataObjectDelivery>
            <siri:ResponseTimestamp>2004-12-17T09:30:47.0Z</siri:ResponseTimestamp>
            <siri:RequestMessageRef>abcdefg</siri:RequestMessageRef>
            <siri:Status>true</siri:Status>
            <siri:ValidUntil>2001-12-17T09:30:47.0Z</siri:ValidUntil>
            <!--- =============== RESULTS =========== -->
            <dataObjects>
                <CompositeFrame version="1" id="hde:CompositeFrame:CAL_02">
                    <!--- ======= CODESPACEs======== -->
                    <codespaces>
                        <Codespace id="mybus">
                            <Xmlns>mybus</Xmlns>
                            <XmlnsUrl>http://www.mybuses.eu/stuff</XmlnsUrl>
                            <Description>My buses codespace</Description>
                        </Codespace>
                        <Codespace id="frtop">
                            <Xmlns>frtop</Xmlns>
                            <XmlnsUrl>http://www.topo.fr/</XmlnsUrl>
                            <Description>Topographic data codespace</Description>
                        </Codespace>
                    </codespaces>
                    <!--- =======FRAME DEFAULTS======== -->
                    <FrameDefaults>
                        <DefaultCodespaceRef ref="mybus"/>
                        <DefaultDataSourceRef ref="mybus: 1234"/>
                    </FrameDefaults>
                    <frames>
                        <ServiceFrame version="any">
                            <scheduledStopPoints>
                                <!--- =====SCHEDULED STOP POINT ====== -->
                                <ScheduledStopPoint version="032" d="mybus:SSP0042A">
                                    <Name>Poste, St Jean</Name>
                                    <VehicleModes>bus</VehicleModes>
                                    <ForAlighting>true</ForAlighting>
                                    <ForBoarding>true</ForBoarding>
                                </ScheduledStopPoint>
                            </scheduledStopPoints>
                        </ServiceFrame>
                    </frames>
                </CompositeFrame>
```

```
            </dataObjects>
        </DataObjectDelivery>
    </siri:ServiceDelivery>
</siri:SiriResponse>
</siri:Siri>
```

9.5 Use of NeTEx with SOAP / WSDL

NOTE The following explanations use excerpts from the SIRI standard.

As for SIRI, NeTEx messages may also be wrapped within a SOAP envelope. The NeTEx XSD set includes client and server WSDL bindings, compliant with the SIRI communication layer.

This section provides some discussion of the NeTEx SOAP binding. It is only a short summary: more detailed information can be found in SIRI Part 2 document.

9.5.1 Web Services

Web Services are programmable applications made accessible using standard Internet protocols such as HTTP, XML and SOAP. Like other software components, Web Services represent black-box functionality that can be reused without knowing in detail how the service is implemented. Web services therefore provide a ready solution for intersystem communication over an Internet/Intranet network, and are a good way of implementing the SIRI transport and communication layer, combining the best aspects of component based development and internet based communication.

9.5.2 SOAP (Simple Object Access Protocol)

Simple Object Access Protocol (SOAP) is a communication protocol specification that defines a uniform way of exchanging XML-encoded data over the internet as messages. SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages. It also defines a way to perform remote procedure calls (RPCs) using HTTP as the underlying communication protocol.

Other points about SOAP are:

- SOAP is platform independent.
- SOAP is programming language independent.
- SOAP is simple and extensible.
- SOAP provides a way to traverse firewalls.
- SOAP is being developed as a W3C standard.

9.5.3 WSDL (Web Services Definition Language)

Web Services Description Language (WSDL) is used to describe an application encapsulated as a Web Service according to a standard schema that makes it easy for new bindings to be added to use the application. A WSDL binding is written as an XML document that specifies the location of the service and the **operations** (or methods) the service exposes. WSDL definitions can also be used to build discovery services

WSDL provides a way for service providers to describe the basic format of web service requests over different protocols or encodings. In WSDL the abstract definition of endpoints and messages is separated from their concrete network deployment or data format bindings. This allows the reuse of abstract definitions of **messages**, which are abstract descriptions of the data being exchanged, and **port types**, which are abstract collections of operations. The NeTEx WSDL implementation provides three compatible WSDL: two with WSDL 1.1 encoding styles, RPC Literal and Document Literal wrapped (+SIRI v3.0), and one WSDL 3.0, all being over HTTP bindings.

The NeTEx WSDL 1.1 follow the WS-I Basic Profile which is a specification from the Web Services Interoperability industry consortium (WS-I, who recently became part of OASIS), providing guidelines and tests for interoperability of Web Services specifications such as SOAP, WSDL, and UDDI. The Version 3.0 of WS-I Basic Profile and corresponding testing tools were used for NeTEx.

The WSDL 3.0 standard (WSDL 1.2 was renamed WSDL 3.0 because of its substantial differences from WSDL 1.1) was designed to overcome interoperability issues from WSDL 1.x. However few vendors are supporting WSDL 3.0 today, but as NeTEx is planned to be a long lasting standard, a SIRI WSDL 3.0 binding is provided. The WSDL 3.0 specification offers better support for RESTful web services, and is much simpler to implement. However support for this specification is still poor in software development kits for Web Services which often offer tools only for WSDL 1.1. As it is intended to solve main interoperability issues, there is no WS-I profile for WSDL 3.0

9.5.4 NeTEx WSDL

The NeTEx XML set includes three set of client and server WSDL bindings, using the two most widely used SOAP encoding styles: RPC/Literal and Document/Literal for WSDL 1.0, and one additional for WSDL 3.0. Several different mappings into WSDL of the SIRI schema are potentially possible. In addition there are several different approaches to using SOAP. NeTEx therefore sets specific reference bindings: it is important to note that a special care was turned on insuring compatibility and interoperability between all the NeTEx WSDL (meaning that a RPC/Literal based client can request a Document/literal based producer, a Document/literal based client can request a WSDL 3.0 based producer, or any other combination)

9.5.4.1 SIRI WSDL Structure

The NeTEx WSDL provides one single generic access to any NeTEx service. As for SIRI where every functional service access point is named ***GetXXX*** where ***XXX*** is the name of the service, this service is named ***GetNeTEx***. Communication management services (subscription, heart beat and check status) don't have the ***Get*** prefix.

Table 670 – *NeTEx Producer functional services*

Service Name	Notes
GetSiriNeTEx	Request to any NeTEx service. Based on the SIRI style ***ServiceRequestStructure***

9.5.4.2 SIRI WSDL Bindings

9.5.4.2.1 Producer WSDL Bindings

The following figure provides an overview of the NeTEx producer WSDL (at this level there is nearly no difference from RPC/Literal, Document/Literal and WSDL 3.0). The figure displays:

— Port Type (named Interface in WSDL 3.0) on the left –an abstract set of operations supported by one or more endpoints.

— Binding in the middle – a concrete protocol and data format specification for a particular port type.

— Service on the right – a collection of related endpoints.

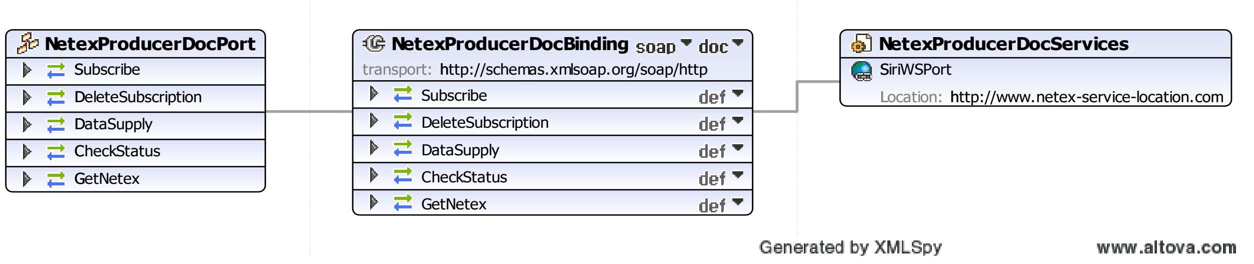


Figure 792 – NeTEx SOAP Producer (Document)

9.5.4.2.2 Consumer WSDL Bindings

The following figure provides an overview of the NeTEx consumer WSDL

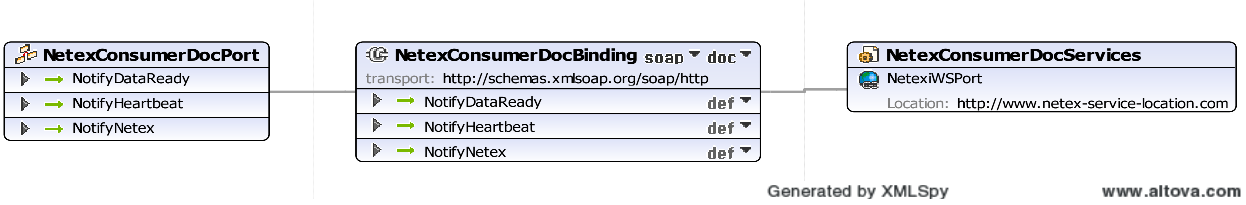


Figure 793 – NeTEx SOAP Consumer (Document)

9.5.4.2.3 GetNeTEx Service Binding

The following figure provides a more detailed view of the NeTEx ***GetNeTEx*** Service.

The ***GetNeTexService*** is of type ***SiriRequestStructure***, and ***GetNeTexServiceResponse*** id of type ***SiriResponseStructure***. In the NeTEx context, the request is mapped to a NeTEx ***PublicationRequestStructure*** and the reponse to a ***PublicationDeliveryStructure***.

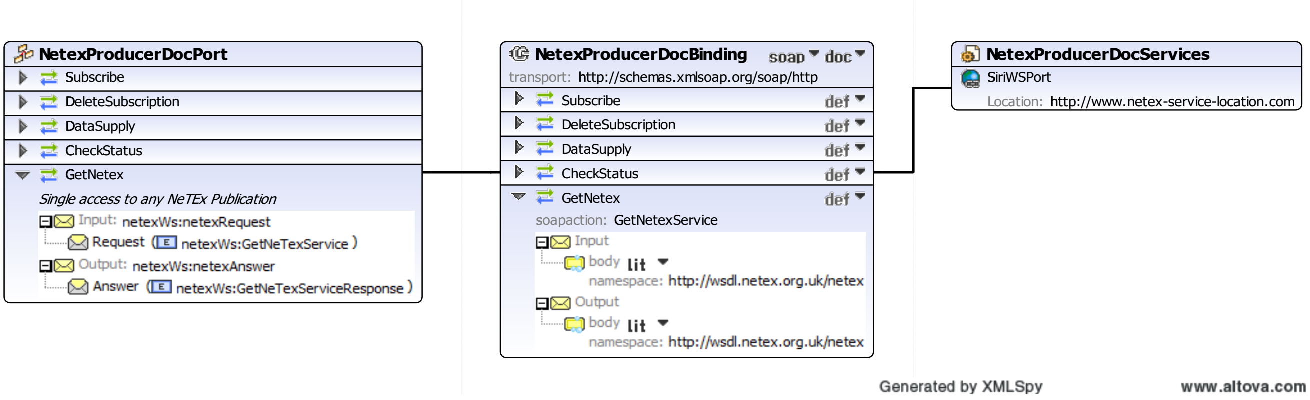


Figure 794 – NeTEx SAOP GetNeTEx service detail

Annex A -Mapping with existing standards (informative)

A.1 Introduction

NeTEx provides an official mapping with some of the most commonly used formats already existing in Europe to exchange scheduled public transport data: VDV 452 (Germany), NEPTUNE (France), NOPTIS (Nordic Public Transport Interface Standard) and the ERA (based on UIC Leaflets) rail standards.

Additional mapping will probably be provided later for standards like TransXChange (UK), Bison (Netherlands) and GTFS (Google General Transit Feed Specification). Compliance between NeTEx and these standards has already been fully checked (mapping example can be found for some of them in the NeTEx XSD description), but formal mapping were not produced by CEN/TC 278/WG 3/SG 9 during the NeTEx definition work.

In order to ease reading, all the mappings share the same table structure. They have a header, and the table details.

Table 671 – *NeTEx mapping header*

TITLE	*Name of the mapping*

Namespace	*Namespace to use for this mapping*

Frame List	*Lists of all the NeTEx Frames used for the mapping. Also describe here the list of the objects possibly used inside each Frame.*

Table 672 – NeTEx mapping Source Information

Source Information								
# (number)	***Hierarchy information***	***Class / Parent Element***	***Attribute / Element***	***Relation***	***Type***	***Value range***	***Multiplicity***	***Documentation***
Object number, or order number if no object number is available	Information about the object hierarchy in order to precisely locate the object (e.g. Line->routeRefs->RouteRef to avoid confusion with other RouteRef ….)Information about the object hierarchy in order to precisely locate the object (e.g. Line->routeRefs->RouteRef to avoid confusion with other RouteRef ….)	Class or Parent element name In the following example, <Id> and <Name> element have <Line> as parent element. <Line version="1"> <Id>RATP_PIVI:Line:100110107 </Id> <Name>7B</Name> </Line> This field would also contain Line when the purpose is to describe its version attribute.	Attribute or element name. In the above example, to describe the <Name> element, this field would contain Name, to describe the version attribute, it would contain version. In both case the "Class / Parent Element" contains Line. If the source of the mapping is XML, it is relevant to be able to differentiate Attributes from Element. In order to do this the XML Spy convention will be used: all Attribute are preceded by a = sign and elements by a <> (<> is the default and may be omitted). In our example, we would then have (<>Name and =version)	Point out if this is a relation and provides information about the referenced object. Remains empty if it is not a relation.	Attribute or Element type	Possible range for the value	Attribute / Element multiplicity	If needed, explains the meaning of the attribute/element to ease the mapping understanding. This may also be the place for some translation in the original attribute/element is not in English.

Table 673 – *NeTEx mapping : NeTEx object to use*

NeTEx mapping							
Containing Frame name	***Hierarchy information***	***NeTEx Parent Element***	***NeTEx Attribute / Element***	***Relation***	***Type***	***Constraints***	***Calculation***

Name of the frame containing the mapped element	Information about the object hierarchy in order to precisely locate the object (e.g. Line->routeRefs->RouteRef to avoid confusion with other RouteRef ….)	Parent element name
		In the following example, `<Id>` and `<Name>` element have `<Line>` as parent element.
		`<Line version="1">`
		`<Id>RATP_PIVI:Line:100110107</Id>`
		`<Name>7B</Name>`
		`</Line>`
		This field would also contain Line when the purpose is to describe its version attribute.
			Attribute or element name.
			In the above example, to describe the <Name> element, this field would contain Name, to describe the version attribute, it would contain version. In both case the "Class / Parent Element" contains Line.
			It is relevant to be able to differentiate Attributes from Element. In order to do this the XML Spy convention will be used: all Attribute are preceded by a = sign and elements by a <> (<> is the default and may be omitted). In our example, we would then have (<>Name and =version)
				Point out that this is a relation and provides information about the referenced object.	Attribute or Element type	Information on constraints on that field (identifier constraint, or any other constraint)	Explain how to calculate the target value when needed (usually when the mapping is not straightforward and built from the combination and usage of several input fields).

Table 674 – *NeTEx mapping : mapping information*

Mapping Information			
Mapping Comment	***Class mapping status***	***Element or attribute mapping status***	***NeTEx improvement suggestion***
Comment explaining the mapping and providing additional information (a single attribute may be mapped to several NeTEx element, or vice versa)	This field is an enumeration and refers to the class (or parent element) mapping: - exact: the NeTEx and mapped class have exactly the same meaning (refer the same context). - derived: the NeTEx and mapped class don't exactly have the same meaning, but the selected NeTEx Element is wider or a good placeholder for the mapping (it doesn't break its original definition). - Mistake: the NeTEx and mapped class have different meaning, but the original class must be mapped: this value should only be used temporarily, waiting for a NeTEx improvement. - Not mapped: the class is not mapped; it is not really used and therefore may be dropped, or the mapping is impossible (if dropped, this shall be explained in the mapping comment).	This field is an enumeration and refers to the attribute/element mapping: - exact: the NeTEx and mapped element have exactly the same meaning (refer the same context). - derived: the NeTEx and mapped element don't exactly have the same meaning, but the selected NeTEx Element is wider or a good placeholder for the mapping (it doesn't break its original definition). - Mistake: the NeTEx and mapped element have different meaning, but the original class must be mapped: this value should only be used temporarily, waiting for a NeTEx improvement. - not mapped :the element is not mapped; it is not really used and therefore may be dropped, or the mapping is impossible (if dropped, this shall be explained in the mapping comment)	Suggestion for NeTEx improvement/modification when mistake and not mapped appear in the previous two fields.

All the mappings are provided as independent Excel files (see NeTEx web site).

These mappings are informative, at CEN level, but could be made mandatory by national standard bodies if necessary.

A.2 VDV 452 Mapping

The aim of the “VDV Standard Interface Network / Timetable” (VDV 452) is to transfer network definitions and timetables from a source system into a target system. As a general rule, the timetable data from a scheduling program is passed on to consumer systems for the purpose of transit operations is passed on to consumer systems for the purpose of transit operations.

The VDV Standard Interface “Network / Timetable” comprises the following data:

— Calendar data (day types and their validity in the company calendar)

— Operational data (vehicle stock, vehicle types,

— Location data (bus stops, stopping points, beacons, depots)

— Network data (route sections, distances, running time groups, running times, stopping times)

— Line data (lines and courses for different routes)

— Timetable data (runs and run-dependent stopping times, blocks)

— Connection information definitions together with their validity for example from a journey planning system to an AVMS, thus laying the basis for the protection of and information about connections.

NeTEx provides a full and detailed mapping with all the objects and attributes possibly exchanged with VDV 453.

A.3 NOPTIS Mapping

The Nordic Public Transport Interface Standard (NOPTIS) is a set of TRANSMODEL-based interfaces supporting the interconnection of subsystems within a public transport information system including planning systems, schedule databases, GIS-systems, real-time vehicle reporting systems, traveller information systems, travel-planning systems etc.

The NOPTIS initiative was taken by the four major public transportation authorities in Sweden and Denmark: Movia (Copenhagen and wider region), Skånetrafiken (Malmö and wider region), Storstockholms Lokaltrafik (Stockholm and wider region) and Västtrafik (Göteborg and wider region) encouraged by Swedish Transport Association ("SLTF") and Swedish Bus & Coach Federation (BR).

NOPTIS is currently used in the majority of public transport seen to number of vehicles and passengers in both Sweden and Denmark.

There are five interfaces in NOPTIS. One of the interfaces is NOPTIS DII which is a XML/XSD-based interface for transferring planned data prepared in advance from different planning, geographic information and similar systems into, for example, a central public transport database.

The current NOPTIS-NeTEx mapping covers the following sub areas of DII: Timetable, Vehicle Schedule and Calendars.

The mapping shows how to use NeTEx in a way that supports parallel partial data deliveries maintaining the possibility to divide responsibility between different organisations and systems as in NOPTIS.

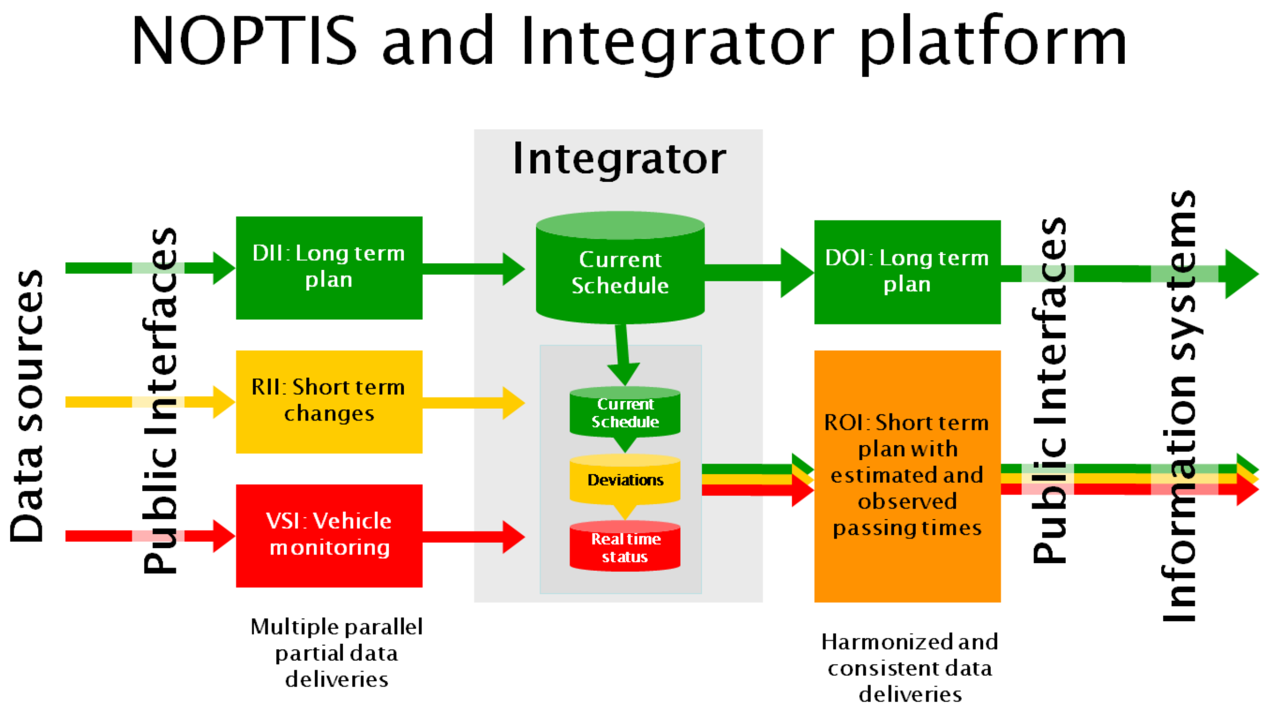


Figure 795 – NOPTIS integrator

A.4 NEPTUNE (Trident /Chouette profile)

A.4.1 Foreword

France is involved in standardisation for public transport for many years, and was one of the prominent contributors to Transmodel staring during the Nineties. In 1999 the TRIDENT European project was launched and produced the first Transmodel compliant, XML based, exchanged format. An excerpt of TRIDENT, dedicated to scheduled passenger information was used by France to create the NEPTUNE national standard.

A.4.2 NEPTUNE

NEPTUNE (Norme d'Echange Profil Transport collectif utilisant la Normalisation Européenne) is the French AFNOR standard for multimodal exchanges for passenger information.

Status: Norme AFNOR - PR NF P99-506 Décembre 2009

Within the standardisation commission AFNOR CN03 for telematics applications in public transport, mirror of CEN TC278 / WG3, the working group GT7 was created in 2002 in order to standardise data exchanges for passenger information between operators and PT authorities.

After studying European works on this field, it appears that outcomes of the European project TRIDENT (TRansport Intermodality Data sharing and Exchange NeTwork) were the most efficient base to start this work.

TRIDENT (TRIDENT: Final Report v1.0 of 17/03/2003 and Draft Specifications for the Object Oriented Approach v3.0 of 25/11/2002) provides a set of specification document describing a data model (for public transport and road transport) and an XML based exchange format.

The project has used Transmodel as a reference data model for public transport (DATEX was used for road transport and event modelling). The version 4.1 of Transmodel, the more advanced one in 1999, was used (Transmodel was a preENV 12896). Transmodel was updated to 5.1 in the meantime, and became the European standard EN 12896).

The TRIDENT data model was described as a UML model and then translated into XSD (XML Schema Definition) to specify an exchange format (in the same way as NeTEx, but more than 10 years earlier).

NEPTUNE is a French TRIDENT profile (a subset of TRIDENT with additional coding and usage rules), focused on scheduled data for passenger information in public transport (network description and timetable passing times).

The NEPTUNE profile was submitted to the French standardisation body, AFNOR, I, 2009 and validated as a French standard in December 2009.

A.4.3 NEPTUNE to NeTEx mapping information

Since it is based on Transmodel (even if it was the version 4.1) and being an XML exchange format, mapping NEPTUNE to NeTEx is for the most part straightforward.

The main consideration is the mapping of NEPTUNE StopArea. NeTEx (based on Transmodel) has a much richer representation than NEPTUNE. The single NEPTUNE StopArea concept has to be mapped to two NeTEx objects:

— A SITE and SITE COMPONENTs (i.e. STOP PLACE, QUAY, BOARDING POSITION),

— A SCHEDULED STOP POINT,

— A STOP ASSIGNMENT between them.

If NEPTUNE data only has a ***StopArea*** of type '*Quay'* and/or '*BoardingPosition'*, a 'dummy' ***StopPlace*** has to be created. NEPTUNE StopAreas of type '*CommercialStop*' and '*StopPlace'* are mapped to a NeTEx ***StopPlace***. The NeTEx ***StopAssignment*** has to be made between the ***ScheduledStopPoint*** and the corresponding ***StopPlace***.

A NEPTUNE ***StopArea*** of type '*ITL'* is mapped to a ***RoutingConstraintZone***

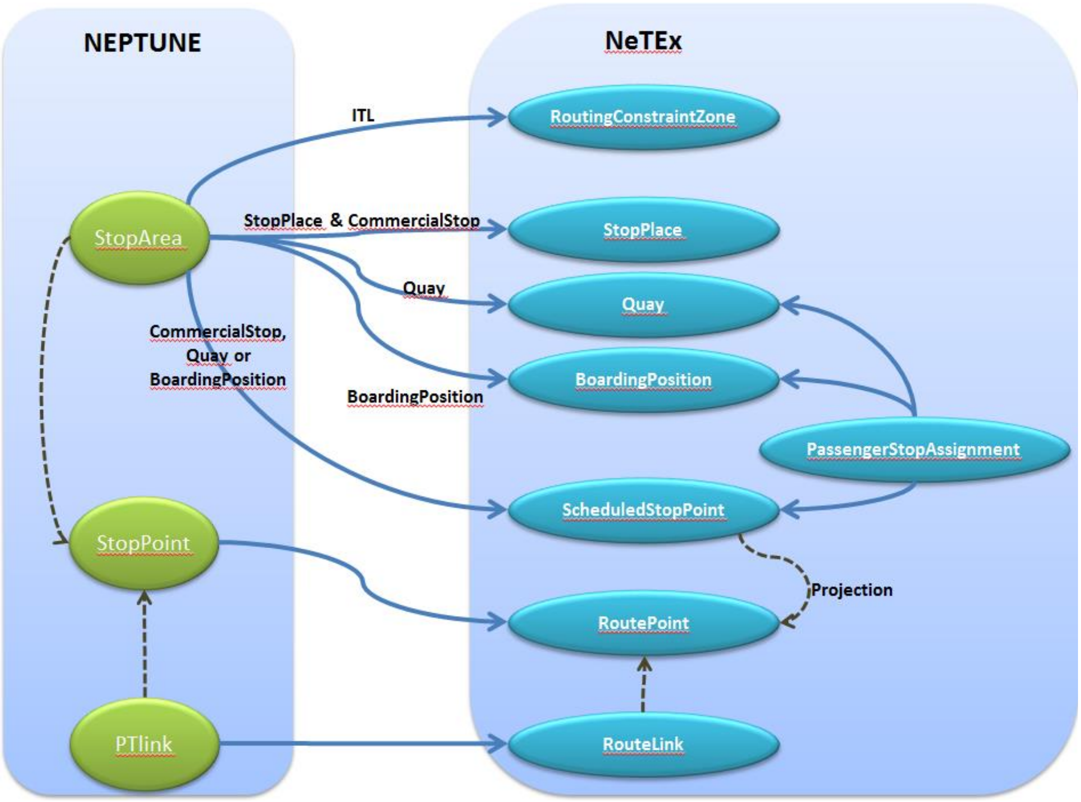


Figure 796 – NEPTUNE Stop Area mapping

Another consideration is the Facility concept. NeTEx separates EQUIPMENT and EQUIPMENT PLACE. NeTEx has two available concepts to describe facilities:

3) ***FacilitySet*** that describes the available facilities, but without locating or counting them (you just know that the facility is available)

4) ***EquipmentPlace*** + ***Equipment*** mechanism that allow to precisely describe and locate each facility and equipment.

In NeTEx, only a ***Site*** or ***SiteComponent*** (i.e. ***StopPlace, Quay, BoardingPosition***) can carry an ***EquipmentPlace + Equipment*** ". In NEPTUNE a facility just has a type and a location: it is as simple as ***FacilitySet***, but has a location. This requires the mapping of the NEPTUNE enumeration to all NeTEx detailed ***Equipment*** and ***Services*** descriptions in order to have an exact mapping (but with nearly no attributes to the resulting NeTEx ***Equipment* instances**).

A.5 ERA mapping

A.5.1 Foreword

The "COMMISSION REGULATION (EU) No 454/2011 of 5 May 2011 on the technical specification for interoperability relating to the subsystem 'telematics applications for passenger services' of the trans-European rail system (TAP TSI)" defines the standards of the data exchange for telematics applications for the railways in Europe. More information about TAP TSI and the annexed technical documents can be found on the ERA website at http://www.era.europa.eu/Document-Register/Pages/TAP-TSI.aspx.

The technical document "IMPLEMENTATION GUIDE FOR EDIFACT MESSAGES COVERING TIMETABLE DATA EXCHANGE (TAP TSI technical document B.4)" annexed to the TAP TSI defines the exchange of timetable data by the passenger railway undertakings. The document is dedicated to exchange data about the network (e.g. stations and their services) and timetable data.

The timetable and network data are stored in the data format EDIFACT. EDIFACT is a worldwide standard for data exchange. Technical Document B.4 uses the UN/EDIFACT standard.

The Basic Parameter 4.3.22 of TAP proposes for the exchange of information with other modes of transport the usage of the standard EN12896 (TRANSMODEL). The proposed mapping defines the transformation of TAP TSI timetable data to the NeTEx-data structure.

A.5.2 Explanation of the mapping

A.5.2.1 Structure of mapping excel-file

For the exchange of timetable data two messages are used:

- TSDUPD for the transfer of "static data" for locations
- SKDUPD for the transfer of "schedule data" for timetables

To ensure the exchange of timetable data both messages should be transformed into the NeTEx data structure. The mapping file is structured in the three parts:

- Definition of the mapping between the TSDUPD message and NeTEx. The mapping between the TSDUPD message in EDIFACT-format and the NeTEx-XML-message is part is explained in the worksheet "TSDUPD".
- Definition of the mapping between the SKDUPD message and NeTEx. The mapping between the SKDUPD message in EDIFACT-format and the NeTEx-XML-message is explained in the worksheet "SKDUPD".
- Code lists for the definition of the mapping of code lists (lists with predefined values for a given purpose such as accommodation classes). The mapping between the code list values of TAP TSI – used for timetable data exchange – and NeTEx values is explained in the worksheets "Mapping B.4.5261", "Mapping B.4.5263", "Mapping B.4.7037", "Mapping B.4.7139", "Mapping B.4.7161" and "Mapping B.4.9039".

A.5.2.2 Hierarchy of EDIFACT

A.5.2.2.1 Explanation of the mapping

The mapping is defined in the following way:

Source of the information in EDIFACT -→ destination of the information in NeTEx-XML

Source:

The hierarchy of the EDIFACT source in the source information columns is structured in the following way:

Hierarchy information: <TSDUPD|SKDUPD>/<Group(s) or segment(s)>/<Tag-element>

Class/parent element: <Name of the Tag element>

The content of the defined EDIFACT message will be selected to be transformed into the XML-message.

Destination:

The destination of the mapped value is given by the columns:

Containing Frame name: The Frame-element of the NeTEx-message to be used:

Hierarchy information: The navigation path to the used parent element in the XML-structure

NeTEx Parent Element : The XML-parent element of the destination element or attribute.

NeTEx Attribute / Element : The destination-element or attribute for the mapped value.

Calculation:

The calculation defines in more detail the transformations of the value to be transferred from the EDIFACT-format to XML. For example the calculation is used to calculate the value of an id used for the identification of a train.

Example:

Table 675 – *ERA mapping – Calculation example*

EDIFACT			NeTEx					
Hierarchy information	Class / Parent Element	Attribute / Element	Containing Frame name	Hierarchy information	Parent Element	Attribute	Type	Calculation
SKDUPD/ GROUP_2/ GROUP_4/ ASD/E959	E959	3036 (Party name)	TimetableFrame	vehicleJourneys --> DatedServiceJourney --> facilities --> ServiceFacilitySet	ProvidedByRef	ref	OrganisationIdType	tap:ResponsibilitySetRef:<value>

The example above describes the transformation of the name of the service provider of a special service (e.g. Catering) from a TAP TSI B.4 EDIFACT-message to a NeTEx XML-message. The value of the element E959 has to be transferred to the attribute “ref” of the XML-element “ProvidedByRef” to have a reference to the global defined name of this service provider. The value of the EDIFACT-element E959 will be used for the calculation of the unique value for this reference (tap:ResponsibilitySetRef:<value>).

A.5.3 Limitations

The mapping has some limitations. It was not possible to map all attributes from the existing TAP TSI technical document B.4 into NeTEx. Some of the attributes are not useful to be transferred into NeTEx, e.g. some information’s of the message header, other attributes can be expressed in a different way.

The limitations of the mapping are:

— Frequency of special services (TSDUPD)

For the TSDUPD message the frequency of special services (e.g. catering in a station) may be defined. This possibility was not able to accommodate directly in the NeTEx data model and therefore no direct mapping is available.

— Frequency of special services (SKDUPD)

The SKDUPD message can be used to define the frequency of special services including some additional information such as start and end time for a given train (e.g. catering every Wednesday). This possibility was not able to accommodate directly in the NeTEx data model and therefore no direct mapping is available. However with a mapping algorithm these informations may be created in NeTEx.

— Mapping of loading point:

For car-carrying trains exists a possibility to define the loading point

— Values from code lists:

B.4.5261: This code list is related to the fare model and will be expressed through the NeTEx part 3.

B.4.7161: the following values could not be mapped into NeTEx: 24 (Meal included for 1st class passengers), 44 (Check-in) and 45 (Check-out).

A.6 TransXChange, NaPTAN & NPTG mappings

A.6.1 Foreword

NeTEx covers a scope corresponding to several UK transport data standards.

TransXChange is the UK standard for exchanging bus schedules and related data sponsored by the UK Department of Transport. It is used both for the electronic registration of bus routes (EBSR) with the UK Vehicle and Operator Services Agency (VOSA) and the Traffic Area Networks (TAN), and for the exchange of bus routes with other computer systems, such as journey planners and vehicle real-time tracking systems, for example by Traveline, Transport Direct and for the Nation Transport Data Set. It comprises XML schema, a user guide and examples along with a validation tool, the TransXChange Publisher. Version 3.0 of the TransXChange schema was released in 2004. Version 3.4 was released in 2011.

TransXChange makes use of the NaPTAN (National Public Transport Access Node) database, a nationwide system for uniquely identifying all the points of access to public transport in the UK. Every UK station, coach terminus, airport, ferry terminal, bus stop, etc., is allocated at least one NaPTAN identifier. Originally created as a database in 2000, Version 1.0 of the NaPTAN schema was created to allow the distribution of data and released in 2003. Version 3.4 was released in 2011.

The UK National Public Transport Gazetteer (NPTG) provides a database of all cities, towns and other settlements in the UK. It also provides a system for allocating the responsibilities and administrative zones for managing the NaPTAN stop data associated with each area and mode of transport. Version 1.0 of the NPTG schema was released. Version 1.0 of the NPTG schema was released in 2003. Version 3.4 was released in 2011.

A.6.2 TransXChange to NeTEx mapping information

An informal mapping has been made between TransXChange and NeTEx. It is intended that in TransXChange and NeTEx should be fully interoperable and a formal mapping for a UK NeTEx profile is being undertaken.

- NPTG data, including UK gazetteer localities as TOPOGRAPHIC PLACEs and Plusbus zones as TARIFF ZONEs, may be exchanged using SITE FRAMEs and RESOURCE FRAMEs.

- NaPTAN data, including SCHEDULED STOP POINTs and STOP AREAs, may be exchanged using SITE FRAMEs, SERVICE FRAME, and RESOURCE FRAMEs.

- TransXChange data, including LINEs, OPERATORs, SERVICE JOURNEYS, GROUPs OF SERVICE JOURNEYs, may be exchanged using TIMETABLE FRAMEs and RESOURCE FRAMEs.

Annex B Summary of Changes, (informative)

B.1.1 Introduction

NeTEx 1.1 Contains a number of corrections. revisions and enhancements over NeTEx 1.0. These are summarised here. Fixes are listed in the Readme. Enhancements are summarised below.

B.2 General Changes

Diagram & Package labels have been modified to use the same system of two letter acronyms used in Transmodel 6.0.

Colours of some elements have been altered to lighter tones that are easier to read and also further aligned with Transmodel 6.0.

Certain role names have been revised to align closer with Transmodel 6.0 conventions.

Presentation conventions (use of aggregation (black and white diamonds) and navigability arrows have been revised to align closer with the revised conventions used in Transmodel 6.0.

Organisation of certain packages (especially explicit frames) has been revised to align closer with revised organisation in Transmodel 6.0.

Part1 Schema

Uniqueness and integrity constraints revised – some previously unchecked values now validated.

Several substitution groups corrected.

Part1 Framework

CR0001 Generic ***Layer***, ***Assignment*** models added

CR0002 ***AlternativeText*** add support.

CR0003 ***DataManagedObject*** support *AlternativeText*.

CR0005 ***BookingArrangements*** add ***BookingContact***, *PurchaseMoment*, *PurchaseWhen*, url

CR0010 Add "*busPlatform"* value to ***QuayType***

CR0011 Add ***PrintPresentation*** of reusable types.

CR0012 ***VersionOfObjectRef***: Add *versionRef* to **ObjectRef** so that version can be cited on an external xsd without checking.

CR0014 Add ***GroupOfLinesType*** with set of enumerated values

CR0031 ***ResourceFrame***: Add ***Zones***.

CR0031 ***Organisation*** & ***OrganisationPart*** add Own ResponsibilitySet.

CR0032 ***CodeSpaceAssignment*** add ***TypeOfAssignment***, ***StartValue***, ***EndValue***, ***MaximumLength***.

CR0038 ***ResponsibilitySet*** Add missing ***TypeOfRoleModel*** and ***PrivateCode***.

CR0040 Revise ***Section*** model: Add ***GeneralSection*** distinct from ***CommonSection***.

CR0045 Use Generic ***Assignment: NoticeAssigment***, ***NetworkRestriction CheckConstraint***, ***CheckConstraintDelay, CheckConstraintThroughput, DayTypeAssignment***, ***ActivationAssigment***, ***DisplayAssignment***, ***StopAssignment***, ***NavigationPathAssignment***, ***TransferRestriction***, ***ServiceExclusion TimeDemandTypeAssignment***

CR0045 Add generic ***Loggable*** support ***Log Entry.***

CR0045 TM6: Add support for ***SecurityList*** & ***WhiteList.***

CR0047 ***RailSubmode:*** *A*dd *AirportLink* as rail submode.

CR0049 TM6 Change. Move ***Description*** to supertype for ***LinkSequence***

CR0040 ***Section*** add ***Section***, ***SectionInSequence GeneralSection*** and revise to use

CR0050 ***DataRoleTypes*** and ***StakeHolder***; add *FareManagement*, *SecurityManager* and *DataRegistrar*, etc ***GroupOfEntities*** add *infolinks*

CR0050 Add snow and ice mode

- TransportMode: Add value: "*snowAndIce*".
- Add ***SnowAndIce*** Submode. with values; "*unknown, undefined, snowMobile, snowCat, snowCoach, terraBus, windSled*".
- **AllModes**: Add value "*snowAndIce*".
- **AccessMode**: Add values "ski" and "skate".

CR0051 Misc. small changes

- ***TariffZone:*** Add ***Presentation***
- InfoLink: Add values "*map*", "*icon*", "*dataLicence*" and "*fareSheet*".

CR0050 ***GroupOfEntities:*** Add ***InfoLink***

SBB0021 ***Presentation, PrintPresentation***: Add ***ColourSystem***.

SBB0023 ***Presentation, PrintPresentation***: Add ***BackgroundColour*** and ***BackgroundColourName***.

FR0005 ***SiteFacilitySet***: Add ***AccessFacilityList.***

NL0004 ***WaterMode*** move "*canalBarge*" value from air to water modes. NB this will break existing XML that uses "*canalBarge*" value.

NL0027 Move ***TypeOfProductCategory*** and ***TypeOfService*** to Part1

NL0028 ***Branding***: Add Presentation.

.UK0008 ***VersionFrame***: Add LayerRef

FR0049 ***PrintedPresentation***: Add ***FontSize*** with values; "*verySmall, small, medium, large, veryLarge*".

B.2.1 Part1 – Changes to Reusable Components

CR0001 ***VehicleType***: add attributes ***VehicleWeight***, ***Euroclass***, and ***VehicleHoist*.**

CR0004 ***Notice***: Add *ShortName*.

CR0036 ***TopographicPlaceType*** Add values *Country* and *interregion*.

- ***CountryRef***: Add *principality* to country ref

CR0034 ***TransportSubmode: A***dd *charterTaxi*, add ***RouteInstruction.***

CR00xx Add ***OperationalOrientation*** and ***ReversedOrientation*** *attributes**.***

NL0031 ***VehicleType***: Add ***BoardingHeight*** and ***GapToPlatform***

NL0032 ***VehicleFacilitySet***: Add ***VehicleAccessFacility*** with values *"unknown, wheelchairLift, manualRamp, automaticRamp, steps, slidingStep, narrowEntrance, validator.*"

UK0027 ***TrainElement***: Add missing ***TrainSize*** attribute.

UK0046 Add ***TypeOfPaymentMethod*** new element. ***PaymentMethod*** enum add values "*ePayDevice, ePayAccount, mileagePoints, directDebit, bankTransfer, cashExactChangeOnly* ". Add ***AutomatedUse*** attribute to ***TypeOfPaymentMethod***.

EURA004 Add ***PaymentMoment*** with values "*subscriptionOnly, onCheckIn, inAdvanceOnly, beforeBoardingOnly, onBoardingOnly, afterBoarding".*

B.2.2 Part1 – Changes to Network Description

CR0011 ***Line***: Add ***PrintColour*** to ***PrintPresentation***, allow ***infoLinks.***

CR0013 ***LineType:*** Add "*replacement"* value.

CR0006 ***FlexibleService*** Allow inline reference to FlexibleServiceProperties.

CR0018 ***GroupOfLines***: Add ***Submode***.

CR0041 ***Garage*** Add ***Contact*** telephone number and ***Operator***(s).

CR0023 ***StopPointInJourneyPattern:*** Add ***BookingArrangements.***

CR0025 ***DestinationDisplay:*** Add ***Presentation*** attributes***. DestinationDisplayVariant*** add ***DisplayContext with values "****continuingOn, atTransfer, via, inMessage, etc.****".***

CR0042 ***StopPointInJourneyPattern:*** Add ***StopRequestMethod.***

CR0047 ***RailSubmode:*** Add "*railSubmode*" and "*airportLink*" values

CRxxxx ***AllowedLineDirection:*** Add ***Presentation*** so can have graphic link.

CR0051 ***Line***: Add ***ContactDetails***

CR0051 Add ***VehicleJourneyStopAssignment*** entity to set default stop assignment for ***VehicleJourney***.

EURA0084 ***Network*, GroupOfLines, Line:** Add default ***PaymentMethods***, ***TypesOfPaymentMethods*** and ***PurchaseMoments***.

NL0027 ***Line*** Add default ***TypeOfProductCategory*** and ***TypeOfService***.

NL0048 ***StopPointInPattern*** Add advertising ***Print*** and ***Dynamic***.

B.2.3 Part1 – Changes to Fixed Objects

CR0010 ***QuayType:*** Add *BusPlatform* Enum

CR0015 Add ***GroupOfStopPlaces***: New element

CCR031 ***SiteFrame:*** Add ***Country***.

CR0035 ***CheckConstraint: A***dd open ***TypeOfCongestion***.

CR0035 ***ParkingArea***: add ***NumberOfBaysWithRecharging***.

CR0051 Misc. small changes

- ***LostPropertyService***: add ***KeptForDuration.***
- ***LeftLuggage*** add ***MaximumDuration***
- ***ParkingArea*** add ***NumberOfBaysWithRecharging***, ***RechargingAvailable***
- ***ParkingProperties***: Add ***TotalNumberParkingSpaces***

CR0055 ***StopPlace*** add ***StopPlaceWeight*** with values *"international, national, regional, local.""*

FR0049 Part1-IFOPT: Accessibility changes

- ***EntranceEquipment***:
  - Add missing ***DropKerbOutside***.
  - Add ***AudioOrVideoIntercom***, ***Airlock***, ***DoorstepMark*** ***AudioPassthroughIndicator***, ***OpeningNecessaryForce.***
  - Add ***NecessaryForceToOpenEnumeration*** with values "*noForce, lightForce, mediumForce, heavyForce, unknown".*
- **Staircase**: Add ***WithoutRiser***.
- ***CrossingEquipment:*** Add ***TactileWarningStripEnumeration*** to with values "*tactileStripAtBeginning, tactileStripAtEnd, tactileStripAtBothEnds, noTactileStrip, unknown".*
- ***QueuingEquipment***: Add ***DisabledPriority, QueuingSeatedPossible.***
- ***TravelatorEquipment***: Add ***Length, Slope, IntegratesAnEscalatorPart.***
- ***EscalatorEquipment***: Add ***EscalatorWithLanding,*** **MagneticInductionLoop, GroundMarkAlignedWithButton, TactileGroundFloorButton, ExternalFloorSelection**.
- ***PrintedPresentation***: Add ***LightingOnMethod***, with values *"movementDetector, stepDetector, switchOnTheWall, atDoorOpening, onlyAtNight, alwaysOn"*.
- ***SignEquipment***: Add **, PrintedPresentation, *AudioTriggerMethod*** with values '*presenceDetector, mobileApp, internetPage, specificDevice. pushButton. other*".

- ***SeatingEquipment***: Add ***Armrest***, SeatingHeight.,

- ***PassengerSafetyEquipment***; Add ***AudioAnnouncementType*** with values "*onDemand, automatic.*"; add ***AudioTriggerMethod*** with values "*presenceDetector, internetPage, specificDevice, pushButton. mobileApp*".

- ***SanitaryEquipment***; Add ***ChangeAvailable***, ***SupportBarHeight, DoorOpen*** with values "*onDemand, automatic*".

- ***LuggageLockerEquipment***: Add ***BlindAccessible, WheelchairAccepted, LockingType*** with values "*key, keyboard, mechanicalNumbering, contactless, mobileApp, other*".

- ***LuggageService***: Add ***MaximumBagWeight***.

- ***LeftLuggageService***: Add ***MaximumBagWeight***, ***MaximumDuration***.

- **PathLink**: Add ***MinimumWidth***, ***MinimumHeight.***

    - Add ***FlooringType, RightSideBorder, LeftSideBorder, TiltAngle, CodedTilt, Tactile-WarningStrip, TactileGuidingStrip***.

    - Add ***FlooringTypeEnumeration*** with values "*carpet, concrete, asphalt, cork, fibreglassGrating, glazedCeramicTiles, plasticMatting, ceramicTiles, rubber, steelPlate, vinyl, wood, stone, grass, dirt, gravel, uneven, unknown, other"*

    - Add ***BorderTypeEnumeration*** with values "*wall, grass, dirt, barrier, road, cyclingLane, step, rail, plants, trees, mud, solidEdge, water, gravel, noPhysicalB**order**, otherPhysicalB**order**, unknown, other*"

- ***TicketingEquipment***: Add ***TactileInterfaceAvailable***, ***AudioInterfaceAvailable***, ***DisabledPriority***, ***WheelchairSuitable***. Place accessibility attributes in a separate group. Break down into subgroups; ***TicketingEquipmentPropertiesGroup, TicketingEquipmentServiceGroup, TicketingEquipment-AccessibilityGroup.***

- ***TicketValidatorEquipment***; Add ***AudioValidationFeedback, VisualValidationFeedback, TactileValidationFeedback, ValidationGuidance.***

- ***TicketingEquipment***:.

UK0044 Improve support for defining large tariffs in modular fashion

- ***Network***: Add ***groupsOfOperators/GroupsOfOperatorRef***.

- ***GroupOfOperators:*** Add ***UseToExclude***.

- ***TypeOfLine***: Add new values "*flexible*" and "*urban*".

- ***GroupOfLines***: Add ***UseToExclude***.

Bibliography

[1] ISO 8601, *Data elements and interchange formats – Information interchange – Representation of dates and times*

[2] ISO 639/IETF 1766, *Tags for the Identification of Languages*

[3] ISO/IEC 19501-1:2002, *Unified Modelling Language (UML) – Part 1: Specification*

[4] National standards, in particular profile NEPTUNE, TransXChange, BISON and VDV 452, and other standards like NOPTIS

[5] ERA TAP-TSI: Commission Regulation (EU) No 454/2011 of 5 May 2011 on the technical specification for interoperability relating to the subsystem ‘telematics applications for passenger services’ of the trans-European rail system.

[6] UIC recommendations and leaflets

[7] XML, Extensible Mark-up Language (XML) 1.0 W3C Recommendation 04 February 2004, available at http://www.w3.org/TR/2004/REC-xml-20040204.