

DIN CEN/TS 16614-4

ICS 35.240.60

**Öffentlicher Verkehr –
Netzwerk- und Fahrplan-Austausch (NeTex) –
Teil 4: Europäisches Profil für Fahrgastinformationen;
Englische Fassung CEN/TS 16614-4:2020**

Public transport –
Network and Timetable Exchange (NeTex) –
Part 4: Passenger Information European Profile;
English version CEN/TS 16614-4:2020

Transport public –
Échange des données réseau et horaires (NeTex) –
Partie 4: Profil Européen pour l'Information Voyageur;
Version anglaise CEN/TS 16614-4:2020

Gesamtumfang 183 Seiten

DIN-Normenausschuss Automobiltechnik (NAAutomobil)
DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

Nationales Vorwort

Dieses Dokument (CEN/TS 16614-4:2020) wurde vom Technischen Komitee CEN/TC 278 „Intelligente Verkehrssysteme“ ausgearbeitet, dessen Sekretariat von NEN (Niederlande) gehalten wird.

Das zuständige deutsche Normungsgremium ist der Gemeinschaftsarbeitsausschuss NA 052-02-71 GA „Gemeinschaftsarbeitsausschuss NAAutomobil/DKE: Intelligente Verkehrssysteme“ des DIN-Normenausschusses Automobiltechnik.

Technische Spezifikationen sind nicht Bestandteil des Deutschen Normenwerks.

Eine Technische Spezifikation ist das Ergebnis einer Normungsarbeit, das wegen bestimmter Vorbehalte zum Inhalt oder wegen des gegenüber einer Norm abweichenden Aufstellungsverfahrens von DIN noch nicht als Norm herausgegeben wird.

Zur vorliegenden Technischen Spezifikation wurde kein Entwurf veröffentlicht.

Das Präsidium des DIN hat mit seinem Beschluss 1/2004 festgelegt, dass von dem in den Regeln der europäischen Normungsarbeit von CEN/CENELEC verankerten Grundsatz, wonach Europäische Normen in den drei offiziellen Sprachen Deutsch, Englisch und Französisch veröffentlicht werden, in begründeten Ausnahmefällen abgewichen und auf die deutsche Sprachfassung verzichtet werden kann.

Dieses Dokument enthält unter Berücksichtigung des DIN-Präsidialbeschlusses 1/2004 nur die Englische Fassung von CEN/TS 17154-1:2019.

Diese europäische Technische Spezifikation legt für den öffentlichen Verkehr Normen zum Datenaustausch für Netzwerke und Fahrpläne (NeTEx) fest. NeTEx wird als Ergänzung zu SIRI (en: Service Interface for Realtime Information) gesehen.

Die Nutzung von offenen normierten Schnittstellen verbessern im Allgemeinen die Wirtschaftlichkeit und technische Realisierbarkeit von System im öffentlichen Verkehr. Eine große Anzahl von Systemlieferanten können eigenständige austauschbare Systeme herstellen und somit konkurrenzfähige Produkte am Markt anbieten. Die Norm fördert somit den Wettbewerb und vermeidet proprietäre Lösungen einzelner Anbieter.

Standardisierte Lösungen erlauben außerdem automatische Prüfprozeduren aller funktionellen Module und reduzieren somit Prüfaufwand und Kosten. Dieses Profildokument der Reihe CEN/TS 16614 fokussiert auf Informationen, die relevant zum Befüllen von Fahrgastinformationsdiensten sind.

Nationaler Anhang NA (informativ)

Begriffe

Die Benummerung der folgenden Begriffe und Abkürzungen ist identisch mit der Benummerung in der Englischen Fassung.

3 Begriffe

Für die Anwendung dieses Dokuments gelten die Begriffe nach EN 12896 (Transmodel V6) und CEN/TS 16614 (NeTEx) und die folgenden Begriffe.

ISO und IEC stellen terminologische Datenbanken für die Verwendung in der Normung unter den folgenden Adressen bereit:

- IEC Electropedia: verfügbar unter <http://www.electropedia.org/>
- ISO Online Browsing Platform: verfügbar unter <http://www.iso.org/obp>

3.1

Attribut

(en: attribute)

Eigenschaft einer Entität

3.2

konzeptuelles Datenmodell

(en: conceptual data model)

Beschreibung einer realen Domain bezogen auf Entitäten, Beziehungen und Attribute in implementierungsunabhängiger Art und Weise

Anmerkung 1 zum Begriff: Es sollte eine Struktur liefern, auf der der Rest der Entwicklung eines Anwendungssystems basieren kann.

3.3

konform

(en: conformant)

Erfüllung aller Regeln einer Spezifikation, sowohl syntaktisch als auch anders; Konformität kann streng, vergrößert oder erweitert sein

3.4

Verbraucher

(en: consumer)

System, das profilkonforme Daten importiert

3.5

Datendomain

(en: data domain)

Datenstruktur (in diesem Dokument ein Teil des Referenzdatenmodells für den öffentlichen Verkehr), welche auf sich gegenseitig bedingende Daten baut und die Tatsache berücksichtigt, dass es einen funktionellen Bereich oder Funktionsgruppen gibt, die diese Datenmenge als Ganzes verwendet

3.6

Datenmodell

(en: data model)

Beschreibung einer realen Domain bezogen auf Daten und deren Beziehungen

3.7

Datenlieferant

(en: data provider)

Organisation, die dafür verantwortlich ist, Daten zu liefern (dies wird ggf. von der Datenquelle oder dem Ersteller unterschieden)

3.8

dominante Gültigkeitsbedingung

(en: dominant validity condition)

Gültigkeitsbedingung, die sich an einen Versionsrahmen hängt, und sich auf ihren gesamten Inhalt bezieht

3.9

Entität

(en: entity)

(Daten)Objekt, dass seine eigene Existenz hat (im Gegensatz zu einem Attribut)

3.10

externe Referenz

(en: external reference)

Referenz auf ein anderes Objekt in einem anderen Dokument als das Dokument, welches die Referenz beinhaltet

3.11

Fahrgeldmanagement

(en: fare management)

alle Aktivitäten bezüglich Geldeinsammlung von Fahrgästen

3.12

funktioneller Bereich

(en: functional area)

zufällig festgelegte Menge von Aktivitäten, die in diesem Dokument dafür verwendet werden, die Eigenschaften und Grenzen des Datenmodells und Austauschprofile festzulegen

3.13

interne Referenz

(en: internal reference)

Referenz auf ein anderes Objekt im selben Dokument, welches die Referenz beinhaltet

3.14

Interoperabilität

(en: interoperability)

Fähigkeit von (Sub)Systemen, mit anderen (Sub)Systemen nach einer Menge vordefinierter Regeln (Schnittstelle) zu interagieren

3.15

logisches, normalisiertes Modell

(en: logical normalised model)

vergleichendes Datenmodell, das nicht vollständig normalisiert ist, d.h. nicht vollständig den normalisierenden Regeln folgt und möglicherweise redundant ist

3.16**objektorientiertes Datenmodell**

(en: object-oriented data model)

Datenstruktur, die nach den Prinzipien ausgedrückt ist, die eine direkte Implementierung als objektorientierte Datenbank ermöglichen, in der die Information in Form von Objekten ausgedrückt wird, d. h. die Darstellung nach den Prinzipien der Verkapselung, im Speziellen, dass auf alle Daten durch zugehörige Operationen (Methoden) zugegriffen wird oder diese modifiziert werden

3.17**Betriebsüberwachung und Steuerung**

(en: operations monitoring and control)

alle Aktivitäten bezogen auf den Transportprozess, d. h. Echtzeitfunktionen bezogen auf das Fahren und den Transport von Fahrgästen nach vorgegebenen Anweisungen, inklusive der Überwachung des Fahrprozesses und dessen Steuerung im Falle von Abweichungen, sowie alle Aktivitäten, die den Fahrprozess unterstützen (Ampelpriorisierung, Fahrstraßensteuerung, Auswahl des Bahnsteigs, Fortschritt/Verzögerungshinweis usw.)

Anmerkung 1 zum Begriff: Solche Informationen werden oftmals von computerunterstützten Werkzeugen unterstützt, auch bekannt als automatisierte Fahrzeugüberwachung (en: Automated Vehicle Monitoring, AVM).

3.18**Fahrgastinformation**

(en: passenger information)

alle Aktivitäten bezogen darauf, den Nutzer entweder über den geplanten oder den aktuellen Transportdienst zu informieren

3.19**Hersteller**

(en: producer)

System, welches Daten veröffentlicht, die mit einem Profil konform sein sollen

3.20**Profil**

(en: profile)

Untermenge eines Standards zur Auswahl der benötigten Konzepte (Entitäten und dazugehörige Attribute) für einen spezifischen Anwendungsfall oder eine Menge von Anwendungsfällen, ergänzt durch Regeln, die festgelegt wurden, um Möglichkeiten unterschiedlicher Interpretationen offener Teile des Standards zu begrenzen

3.21**Profilkodewert**

(en: profile code value)

Metadaten, die zur Beschreibung eines Profils verwendet werden; ein normierter Kodewert, eindeutig im Namensbereich des Profils, welcher hilft, das Profil festzulegen

3.22**Profilmetadaten**

(en: profile metadata)

Daten, welche die Eigenschaften eines Profils beschreiben, wie z. B. der erlaubte Rahmentyp und beschränkte Kodewerte

3.23**Profilprüfer**

(en: profile validator)

Softwaretool, welches den Inhalt eines profilkonformen XML-Dokuments überprüft, um sicherzustellen, dass dieses konform zur Profilspezifikation ist

3.24

Echtzeitsteuerung

(en: real-time control)

siehe Betriebsüberwachung und Steuerung

3.25

Zeitplanung

(en: scheduling)

siehe taktische Planung

3.26

spezifischer Rahmen

(en: specific frame)

Versionsrahmen, welcher möglicherweise nur bestimmte Typentitäten beinhaltet, die sich mit einem bestimmten funktionellen Bereich beschäftigen, z. B. um einen Fahrplan zu beschreiben

3.27

strenge Konformität

(en: strict conformance)

Konformität mit einem Profil in der Art und Weise, dass ein Dokument nur die festgelegten Elemente beinhaltet (d. h. dass es nicht vergrößert oder erweitert ist)

3.28

taktische Planung

(en: tactical planning)

alle Aktivitäten bezogen auf die taktischen Transportplanungen, aufgeteilt auf Fahrzeugzeitplanungen, Fahrerzeitplanungen, Dienstenteilung

3.29

XML-Dokument

(en: XML document)

Datei, welche Daten beinhaltet, die mit XML-Tags exakt konform mit einem XML-Schema gekennzeichnet sind, welches genaue Syntaxregeln für die Namen festlegt, die Verschachtelung und Kardinalität der Tags sowie die Formate der Datenwerte

3.30

XML-Überprüfer

(en: XML validator)

Softwaretool, welches den Inhalt eines XML-Dokumentes prüft um sicherzustellen, dass dieses mit einem Schema konform ist

English Version

Public transport —
Network and Timetable Exchange (NeTex) —
Part 4: Passenger Information European Profile

Transport public —
Échange des données réseau et horaires (NeTex) —
Partie 4: Profil Européen pour l'Information Voyageur

Öffentlicher Verkehr —
Netzwerk- und Fahrplan-Austausch (NeTex) —
Teil 4: Europäisches Profil für Fahrgastinformationen

This Technical Specification (CEN/TS) was approved by CEN on 2 March 2020 for provisional application.

The period of validity of this CEN/TS is limited initially to three years. After two years the members of CEN will be requested to submit their comments, particularly on the question whether the CEN/TS can be converted into a European Standard.

CEN members are required to announce the existence of this CEN/TS in the same way as for an EN and to make the CEN/TS available promptly at national level in an appropriate form. It is permissible to keep conflicting national standards in force (in parallel to the CEN/TS) until the final decision about the possible conversion of the CEN/TS into an EN is reached.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and United Kingdom.



EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

Contents

Page

European foreword	8
Introduction	9
1 Scope.....	13
2 Normative references.....	13
3 Terms and definitions	13
4 Presentation conventions	16
4.1 Recap of NeTEx presentation conventions	16
4.2 Additional EPIP presentation conventions	17
4.3 Transmodel definitions and EPIP	18
5 Framework.....	19
5.1 Introduction.....	19
5.2 DataManagedObject, Version and Codespace.....	19
5.2.1 Overview	19
5.2.2 Entity (Abstract).....	21
5.2.3 EntityInVersion (Abstract)	21
5.2.4 DataManagedObject (Abstract)	22
5.2.5 AlternativeText (Subcomponent of DataManagedObject).....	23
5.2.6 KeyValue (Subcomponent of DataManagedObject)	23
5.2.7 Version.....	24
5.2.8 Codespace	24
5.2.9 ValidityCondition	26
5.2.10 ValidBetween	27
5.2.11 ResponsibilitySet	27
5.2.12 ResponsibilityRoleAssignment (Subcomponent of ResponsibilitySet)	27
5.2.13 DataSource	28
5.3 Relationships and entity references	29
5.3.1 General rules	29
5.3.2 VersionOfObjectRef.....	30
5.3.3 Integrity Checking of XML Documents	31
5.3.4 Identity uniqueness scope	32
5.4 Frames	33
5.4.1 Overview	33
5.4.2 Version Frame (Abstract)	34
5.4.3 TypeOfFrame	35
5.4.4 Specific frames and General Frames.....	36
5.4.5 Composite Frame	37
5.4.6 FrameDefaults.....	37
5.5 Generic elements	38
5.5.1 Introduction.....	38
5.5.2 TypeOfValue (Abstract)	38
5.5.3 ValueSet.....	39
5.5.4 GroupOfEntities (Abstract)	40
5.5.5 Point (Abstract)	40
5.5.6 Location (Subcomponent of Point).....	41
5.5.7 Link (Abstract).....	41

5.5.8	LinkSequence (Abstract).....	42
5.5.9	PointInLinkSequence (Abstract).....	43
5.5.10	Zone (Abstract).....	43
5.5.11	Transport Mode and Submode	43
5.5.12	Assignment (Abstract)	44
5.5.13	Presentation (Subcomponent – Multipurpose)	45
5.5.14	Locale (Subcomponent – Multipurpose)	46
5.5.15	Booking Arrangements (Subcomponent – Multipurpose).....	46
5.5.16	ContactDetails (Subcomponent – Multipurpose)	47
5.6	Accessibility.....	47
5.6.1	Overview	47
5.6.2	AccessibilityAssessment (Subcomponent – Multipurpose)	48
5.6.3	AccessibilityLimitation (Subcomponent of AccessibilityAssessment).....	49
5.7	Facility sets	50
5.7.1	Overview	50
5.7.2	FacilitySet (Abstract).....	51
5.7.3	ServiceFacilitySet.....	51
5.7.4	Accommodation – (Subcomponent of ServiceFacilitySet)	51
5.7.5	SiteFacilitySet.....	52
6	Stop Profile.....	52
6.1	Overview	52
6.2	Stop Place hierarchies.....	54
6.3	Other types of Place	57
6.3.1	Introduction.....	57
6.3.2	Place (Abstract).....	58
6.3.3	AddressablePlace (Abstract)	58
6.3.4	SiteElement (Abstract)	59
6.3.5	Site (Abstract)	60
6.3.6	Address (Abstract)	61
6.3.7	PostalAddress (Subcomponent of SITE in EPIP)	62
6.3.8	RoadAddress (Subcomponent of SITE in EPIP).....	62
6.3.9	Topographic Place.....	63
6.3.10	TopographicPlaceDescriptor (Subcomponent).....	64
6.3.11	Point of Interest.....	64
6.4	Stop Places.....	65
6.4.1	StopPlace	65
6.4.2	SiteComponent (Abstract)	67
6.4.3	StopPlaceSpace (Abstract)	67
6.4.4	Quay (Subcomponent of Quay)	67
6.4.5	Entrance (Abstract).....	68
6.4.6	StopPlaceEntrance	69
6.5	GroupOfStopPlaces	69
6.6	TariffZone.....	69
7	Network and Timetables.....	70
7.1	General	70
7.2	Basic network topology	70
7.2.1	Overview	70
7.2.2	Line	71
7.2.3	AllowedLineDirection (Subcomponent of Line).....	73
7.2.4	GroupOfLines	73
7.2.5	Network.....	74
7.2.6	DestinationDisplay.....	74

7.2.7	Via.....	75
7.2.8	DestinationDisplayVariant.....	75
7.2.9	Organisation (Abstract).....	76
7.2.10	Operator.....	77
7.2.11	Authority.....	77
7.3	Routes.....	77
7.3.1	Overview.....	77
7.3.2	RoutePoint.....	78
7.3.3	RouteLink.....	78
7.3.4	Route.....	79
7.3.5	PointOnRoute (Subcomponent of Route).....	79
7.4	Service patterns.....	80
7.4.1	Overview.....	80
7.4.2	JourneyPattern (Abstract in EPIP).....	81
7.4.3	ServicePattern.....	82
7.4.4	PointInJourneyPattern (Abstract in EPIP).....	82
7.4.5	StopPointInJourneyPattern (Subcomponent of JourneyPattern).....	83
7.4.6	Direction.....	84
7.4.7	TimingPoint (Abstract in EPIP).....	85
7.4.8	ScheduledStopPoint.....	85
7.4.9	Service Link.....	86
7.5	Assigning Stops.....	87
7.5.1	Overview.....	87
7.5.2	StopAssignment (Abstract).....	87
7.5.3	PassengerStopAssignment.....	88
7.5.4	DynamicStopAssignment.....	88
7.5.5	TrainStopAssignment (Subcomponent of Passenger Stop Assignment).....	89
7.6	Connections, transfers and interchanging.....	90
7.6.1	Overview.....	90
7.6.2	Transfer (Abstract).....	90
7.6.3	Transfer Duration (Subcomponent of Transfer).....	91
7.6.4	Connection.....	91
7.6.5	ConnectionEnd (Subcomponent of Connection).....	92
7.6.6	SiteConnection.....	92
7.6.7	SiteConnectionEnd (Subcomponent of SiteConnection).....	93
7.6.8	DefaultConnection.....	93
7.6.9	DefaultConnectionEnd (Subcomponent of DefaultConnection).....	94
7.7	Vehicle journeys and passing times.....	94
7.7.1	Overview.....	94
7.7.2	Journey (Abstract).....	96
7.7.3	ServiceJourney.....	96
7.7.4	PassingTime.....	97
7.7.5	PassingTime (Abstract).....	98
7.7.6	TimetabledPassingTime (Subcomponent – Service Journey).....	98
7.7.7	GroupOfServices.....	98
7.7.8	GroupOfServicesMember (Subcomponent of GroupOfServices).....	99
7.8	Very frequent journeys.....	100
7.8.1	Introduction.....	100
7.8.2	TemplateVehicleJourney.....	100
7.8.3	JourneyFrequencyGroup (Abstract).....	101
7.8.4	HeadwayJourneyGroup (Subcomponent of TemplateJourney).....	101
7.9	Transferring between services.....	102
7.9.1	Introduction.....	102

7.9.2	Interchange (Abstract)	102
7.9.3	ServiceJourneyInterchange	103
7.10	Trains and train coupling	103
7.10.1	Overview	103
7.10.2	TrainNumber.....	104
7.10.3	JourneyPart (Subcomponent of ServiceJourney).....	105
7.10.4	JourneyPartCouple	105
7.10.5	JourneyPartPosition (Subcomponent of JourneyPartCouple)	106
7.11	Vehicles and Trains.....	107
7.11.1	Overview	107
7.11.2	VehicleType	107
7.11.3	Train	108
7.11.4	TrainElement (Subcomponent of Train).....	109
7.11.5	TrainComponent (Subcomponent of Train)	109
7.12	Notices	110
7.12.1	Introduction.....	110
7.12.2	Notice (Subcomponent - Multipurpose)	110
7.12.3	NoticeAssignment (Subcomponent - Multipurpose)	111
7.13	Calendars.....	112
7.13.1	Overview	112
7.13.2	DayType	112
7.13.3	PropertyOfDay (Subcomponent of DayType).....	113
7.13.4	Timeband (Subcomponent of DayType)	114
7.13.5	ServiceCalendar	115
7.13.6	OperatingDay (Subcomponent of ServiceCalendar)	115
7.13.7	DayTypeAssignment (Subcomponent of ServiceCalendar).....	116
7.13.8	Operating Period (Abstract in EPIP).....	117
7.13.9	UicOperatingPeriod (Subcomponent of ServiceCalendar).....	117
7.14	Flexible services.....	118
7.14.1	Introduction.....	118
7.14.2	FlexibleLine	118
7.14.3	FlexiblePointProperties (Subcomponent - Multipurpose)	118
7.14.4	FlexibleServiceProperties (Subcomponent of SERVICE JOURNEY)	119
8	Common rules	120
8.1	Version management of data.....	120
8.2	Use of validity conditions in EPIP	121
8.3	Populating creation and change dates.....	121
8.4	Codespaces and the structure of identifiers	122
8.4.1	Introduction.....	122
8.4.2	Use of codespaces in EPIP.....	123
8.4.3	Identifier structure for EPIP data objects.....	123
8.4.4	Identifier structure for EPIP version frames	124
8.5	Versions of the EPIP	129
8.6	Dependencies between frames.....	129
8.7	Organisation and nesting of elements within an EPIP document	130
8.8	Responsibility for data.....	130
8.9	Handling large data sets.....	131
8.10	Available types of frame.....	132
8.10.1	Type of Frame list.....	132
8.10.2	Classes for SPECIFIC FRAMES.....	133
8.10.3	Classes for COMPOSITE FRAMES	137
8.10.4	Classes for EPIP Metadata FRAMES	138

8.11	Completeness of data.....	139
8.11.1	Introduction.....	139
8.11.2	Empty values.....	139
8.12	Data exchange	140
8.13	PublicationDelivery	140
8.13.1	Introduction.....	140
8.13.2	Publication Request (Subcomponent of PublicationDelivery)	141
8.13.3	NetworkFrameTopic (Subcomponent of PublicationDelivery)	141
8.13.4	NetworkFilterByValue (Subcomponent of PublicationDelivery).....	142
8.13.5	PublicationDelivery example	142
8.14	Names of files for EPIP documents	144
8.15	Extending the profile	145
9	Profile compliance and data quality checking	146
9.1	Introduction.....	146
9.2	Methods of checking the quality of data	146
9.3	Quality issue resolution.....	147
9.4	Profile validator rules	147
9.5	Metadata for profiles	148
9.6	Summary of data quality rules for EPIP	149
9.7	Referential integrity (automatic).....	155
	Annex A (normative) Data Dictionary	156
	Annex B (normative) Facility code list.....	166
B.1	AccessFacilityList.....	166
B.2	AccessibilityInfoFacilityList.....	166
B.3	AccessibilityToolList.....	166
B.4	AccommodationAccessList.....	167
B.5	AccommodationFacilityList	167
B.6	AssistanceFacilityList.....	167
B.7	BoardingPermissison	167
B.8	BookingProcessFacilityList.....	168
B.9	CarServiceFacilityList.....	168
B.10	CateringFacilityList	168
B.11	CouchetteFacilityList	169
B.12	FamilyFacilityList.....	169
B.13	FareClasses.....	169
B.14	GenderLimitation.....	169
B.15	GroupBookingFacility	170
B.16	LuggageCarriageFacilityList.....	170
B.17	MealFacilityList.....	170
B.18	MedicalFacilityList	170
B.19	MobilityFacilityList	171
B.20	NuisanceFacilityList.....	171
B.21	PassengerCommsFacilityList.....	171
B.22	PassengerInformationEquipmentList.....	172
B.23	PassengerInformationFacilityList	172
B.24	RetailFacilityList	172
B.25	SafetyFacilityList.....	172
B.26	SanitaryFacilityList	173
B.27	ServiceReservationFacilityList	173
B.28	TicketingFacilityList	173
B.29	TicketingServiceFacilityList.....	174

B.30	UicProductCharacteristicList	174
B.31	UicTrainRate	174
B.32	VehicleFacilityList	174
Annex C (informative)	Related EU Directive	175
Bibliography		177

European foreword

This document (CEN/TS 16614-4:2020) has been prepared by Technical Committee CEN TC 278 “Intelligent transport systems”, the secretariat of which is held by NEN.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN shall not be held responsible for identifying any or all such patent rights.

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

NeTEx is made up of three parts defining a single European Standard, 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.

Part 1 is fully standalone, Parts 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.

This document is highly technical, and 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, attribute names are in bold/italic style and use camelcase style with no spaces (***JourneyPattern*** for example).
- To avoid confusion with usual wording, attribute types are in italic style and use camelcase style with no spaces (*TypeOfEntity* for example).

According to the CEN/CENELEC Internal Regulations, the national standards organisations of the following countries are bound to announce this document: Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.

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 document specifies a Profile of Network and Timetable Exchange (NeTEx) for public transport. NeTEx is intended to be used to exchange information between Public Transport organisation systems containing scheduled public transport data. It can also be seen as a complement to the SIRI (Service Interface for Real-time Information) standard (EN 15531 series), as SIRI needs reference data exchanged in the scope of NeTEx before any possible real-time exchange.

Well-defined, open interfaces have a crucial role in improving the economic and technical viability of public transport Information Systems of all kinds. Using standardized 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 functions.

NeTEx improves 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:
 - introducing common architectures for message exchange;
 - introducing a modular set of compatible information services for real-time vehicle information;
 - using common data models and schemas for the messages exchanged for each service;
 - introducing a consistent approach to data management.
- Technical advantages include the following: reusing 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.

A profile is an ancillary document to the standard which specifies additional rules for implementation in a given context. The profile contains information such as:

- Details of the objects used in an exchange.
- Details on the options proposed by the standard.
- Details on optional elements.
- Precision on the identifier codes to be used.
- Advice on grouping elements.

The reason for having a detailed profile specification is that it facilitates implementation. A developer intending to implement a certain service or type of service in a given environment need only implement the smaller set of options and parameters specified in the profile, and is given a number of additional rules that restrict or simplify the required processing of data.

This European Passenger Information Profile (EPIP) for NeTEx is for exchanging passenger information; it describes a minimal information set to feed passenger information services in a

European wide and multimodal context. It covers basic network and timetable data as explained further below.

Profiles – a Rationale

Standards are by their nature and definition, consensus documents. In the case of the CEN and ISO standards, and more particularly in the field of traffic applications, these are established at an international level, with a significant number of different stakeholders involved in their development and discussion. This means that CEN and ISO standards take into account requirements that are far beyond many local implementations of such standards. Implementing a standard for a complex data model (such as that for transport) represents a significant investment, so care is taken to achieve a comprehensive and well abstracted solution and standards are written to have with the longest possible life and stability.

These factors tend to result in standards documents being large and detailed, consequently requiring significant effort to read and understand – and this is certainly the case for Transmodel and NeTEx. It may also be difficult to see how the standard can best applied in practice, since there are many subtle choices to make out of all the possibilities and there is insufficient space to include extensive examples in the formal documentation.

Another cause of complexity is that standards such as NeTEx and SIRI (since they were developed to harmonise different national standards), include features and options whose purpose is to ensure compatibility with the different systems developed in specific countries. For example, SIRI includes services dedicated to the advanced management of connection guarantees, as implemented in the German VDV standard, but used only in a few countries such as Switzerland and Germany; NeTEx also includes features to help compatibility with practices peculiar to particular countries, such as are found in the French NEPTUNE, British TransXChange, Swedish NOPTIS, etc. standards.

In addition, local or national specificities may require the specific use of a specific data sets and specific identifier systems for particular information. For example, the UK has a national system for the identification of stops (NaPTAN), which is naturally required to be used within NeTEx and SIRI exchanges, but which is not relevant for other European countries except in a few cross-border services.

Finally, some elements proposed by the standards are optional and relevant only for a particular business context: so that it needs to be decided if these items are relevant for a particular implementation or not.

A profile is intended to address some of the above issues by offering a predefined set of choices for use in specific context, and setting out additional explicit rules that help to simplify implementation.

Additionally, a profile can be accompanied by the definition of specific testing procedures to assess the conformance of the implemented solution with the profile.

A profile remains fully compliant with the standard; it merely implements a subset, using a well-defined code system.

From a practical point of view, profiles can be seen as a set of implementation guidelines; instead of having to face the challenge of analysing the whole standard, discovering the relevant parts for a given application, and then adjusting optional values and parameters for the intended application, a profile

can be specified to address the needs of a particular application and then used for any subsequent similar initiatives.

The European Passenger Information Profile (EPIP)

The EPIP focuses on information relevant for feeding passenger information services, it therefore excludes operational information that is not relevant for this purpose. Fare information, apart from basic Tariff zones, though relevant for passengers, is also considered as out of scope, but will be addressed by a future separate profile.

Typical use cases for the Passenger Information Profile are:

- provision of data to a journey planner.
- provision of data to a mapping tool to show the network on a map (possibly interactive).
- provision of data to timetable printing/visualisation tools.
- provision of data to a stop or line finder.

The profile has been designed to be as concise as possible and to focus on the data elements needed to fulfil the EC's *Priority action A of ITS-Directive*.

- This means that features specific to a particular country will not in general not be included (the profile is designed to be sharable across Europe, and data sets compliant with it are expected to be usable by any NeTEx conformant passenger information system in Europe).
- Each country may also define a further national NeTEx Profile (for internal use nationally), extending the European profile and providing additional information.
- In the interests of simplicity, a number of the more complex features specific to certain modes have also been omitted from the formal profile, for example boarding position on train platforms.
- The EPIP is divided into three parts:
- **Common core elements** that are shared by the two other profile parts given in this document and that may also be reused later on by other future profiles (Fare Profile, Accessibility Profile, etc.)
- **Stop Profile**, describing the STOP PLACES and their hierarchy.
- **Network and Timetables Profile**, describing the network topology and all related timing information (timetables).

Dividing the profile into three parts allows for a modular implementation, with a minimal overlap between separate documents and their corresponding components. The modularisation also facilitates the separate exchange of stop data from that of timetable and other information (but it is, of course, still possible to exchange them together). One of the reasons for this partition is that there are more and more countries developing separate national or regional stop database registries who require the ability to exchange only the stop description.

The EPIP supports certain common features that are important in a European context:

- The ability to exchange translations of text elements in multiple alternative national languages.
- Basic accessibility information about sites and services.

Data consistency and Quality

One of the goals of a profile is to simplify and enhance interoperability. However, mere syntactic compliance with the EPIP profile alone will not guarantee interoperability; the data being exchanged must also be of good quality, that is, complete and self-consistent as data, -- and correspond to the external reality it is meant to be describing.

The profile therefore also includes a set of consistency and quality checking rules. Following these rules will not, of course, necessarily guarantee the absolute quality of any dataset (nor will it validate the data against the real-world), but will prevent many of the basic quality issues that are commonly encountered. A number of rules may be controlled automatically, thus for example, it is possible to check that a SERVICE JOURNEY has PASSING TIMES for each stop, and that these are plausibly spaced; but other quality measures cannot necessarily be checked. For example, the lack of a STOP PLACE name can be detected, as can the use of a duplicate, but not whether it is actually the correct one in use in the real-world.

NeTEx versions

The EPIP is specified for the revised version 1.1 of NeTEx, issued in 2019. It may also be used with the original version 1.0 of NeTEx, issued in 2014, though certain elements are of course lacking.

If recourse to the NeTEx specification is needed, it is strongly recommended that the 1.1 version is used as it contains numerous clarifications and corrections to the original 1.0 document.

1 Scope

This document is a profile of CEN/TS 16614 series. It focuses on information relevant to feed passenger information services and excludes operational and fares information.

NeTEx is dedicated to the exchange of scheduled data (network, timetable and fare information) based on Transmodel V6 (EN 12986) and SIRI (CEN/TS 15531-4/5 and EN 15531-1/2/3) and supports information exchange of relevance to public transport services for passenger information and AVMS systems.

As for most data exchange standards, defining subsets of data and dedicated rules for some specific use case is of great help for implementers and for the overall interoperability. This subset is usually called profile and this profile targets passenger information as only use case.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

EN 12896, *Road transport and traffic telematics - Public transport - Reference data model (Transmodel) Parts 1 to 9*

CEN/TS 16614-1, *Network and Timetable Exchange (NeTEx) - Network description*

CEN/TS 16614-2, *Network and Timetable Exchange (NeTEx) - Timing information*

CEN/TS 16614-3, *Network and Timetable Exchange (NeTEx) - Fare description*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in EN 12896 (Transmodel V6) and CEN/TS 16614 (NeTEx) and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- ISO Online browsing platform: available at <https://www.iso.org/obp>
- IEC Electropedia: available at <http://www.electropedia.org/>

3.1

attribute

property of an entity

3.2

conceptual data model

description of a real-world domain in terms of entities, relationships and attributes, in an implementation independent manner

Note 1 to entry: It should provide a structure on which the rest of the development of an application system can be based.

3.3

conformant

satisfying all the rules of a specification, both syntactic and otherwise; conformance may be strict, augmented, or extended

3.4

consumer

system that imports profile conformant data

3.5

data domain

data structure (in this document, a part of the Reference Data Model for Public Transport) made up of data related to each other, through the fact that there is a functional area or group of functions using this data set as a whole

3.6

data model

description of a real-world domain in terms of data and relationships

3.7

data provider

organisation responsible for providing data (this may be distinct from that of the data source or the producer)

3.8

dominant validity condition

validity condition that attaches to a version frame and applies to all its contents

3.9

entity

object (data) that has its own existence (as opposed to an attribute)

3.10

external reference

reference to another object in a document other than the document holding the reference

3.11

fare management

all activities related to the collection of money from passengers

3.12

functional area

arbitrarily defined set of activities, used, in this document, to define the objectives and limits of the data model and exchange profile

3.13

internal reference

reference to another object in the same document as that holding the reference

3.14

interoperability

ability of (sub)systems to interact with other (sub)systems according to a set of predefined rules (interface)

3.15

logical normalised model

relational data model that is not fully normalised, i.e. does not completely follow the normalisation rules and thus may be redundant

3.16

object-oriented data model

data structure expressed according to principles that allow for a direct implementation as an object-oriented database, where information is represented in form of objects, i.e. respecting the principle of encapsulation meaning in particular that each data is accessed or modified through operations (methods) belonging to it

3.17

operations monitoring and control

all activities related to the transportation process, i.e. real-time functions related to the driving and transportation of passengers according to given instructions, including the monitoring of the driving process and its control in case of deviations, as well as all activities that support the driving process (traffic light priority, track switching, bay selection, advance/delay advice, etc.)

Note 1 to entry: Such functions are often assisted by computer-aided tools, known as Automated Vehicle Monitoring (AVM)

3.18

passenger information

all activities related to informing the users either about the planned or about the actual transportation services

3.19

producer

system that outputs data intended to be conformant to a profile

3.20

profile

subset of a standard selecting the needed concepts (entities and associated attributes) for a specific use case or set of use cases, and complemented by rules defined to restrict possibilities of divergent interpretations of open parts of the standard

3.21

profile code value

metadata used to describe a profile; a standardised code value, unique with the namespace of the profile, that helps define the profile

3.22

profile metadata

data describing the properties of a profile, such as the permitted frame types and restricted code values

3.23

profile validator

software tool that checks the content of a profile conformant XML document to ensure that it is compliant with the profile specification

3.24

real-time control

see Operations monitoring and control

3.25

scheduling

see Tactical Planning

3.26

specific frame

version frame that may only contain certain types entities concerned with a particular functional area, e.g. to describe a timetable

3.27

strict conformance

conformance to a profile such that a document contains only the specified elements (i.e. that is not augmented or extended)

3.28

tactical planning

all activities related to the tactical planning of transportation, split into vehicle scheduling, driver scheduling, rostering

3.29

XML document

file containing data marked up with XML tags conforming exactly to an XML schema that specifies precise syntactic rules for the names, nesting and cardinalities of the tags and for the formats of the data values

3.30

XML validator

software tool that checks the content of an XML document to ensure that it is compliant with a schema

4 Presentation conventions

NOTE The conventions used in this document are an extension of the ones used in the main NeTEx document.

4.1 Recap of NeTEx presentation conventions

NeTEx follows certain conventions for the presentation of technical terms:

- Transmodel conceptual model elements ('ENTITIES') are shown in UPPER CASE, for example, "LINE", "SCHEDULED STOP POINT". For readability, they may be pluralised with a lower case 's', e.g. "some LINES."
- All XML elements in the NeTEx XML schema are shown in ***bold italic***, for example, ***Line***. Compound names are camel cased without a space and are never pluralised, e.g. ***"ScheduledStopPoint"***.
- Properties that are implemented as the XML class attributes of a complex class are in ***bold lower-case italic***, for example, ***"id"***, ***"order"***.
- Properties of an entity that are implemented as an XML child element, are also capitalised and shown in bold for example, ***"ScheduledStopPoint / Name"***, ***"SiteElement / IsCovered"***. This is regardless of whether the child element is a simple XML type (e.g. *xsd:boolean*) or a complex XML type (e.g. *PresentationStructure*).
- Restricted lists of enumeration values, (which are mostly in *lowerCamelCase*) are shown in *italics*; where values appear within free text, they are encased in single quotes to distinguish them from the plain text words, for example, *'busStop'*, *'canalBarge'*, *'rail'*.

- One-to-one relationships that are implemented as versioned references are shown as simple attribute names, for example, ***ScheduledStopPointRef***, ***ZoneRef***. When there is a reflexive relationship, or more than one relationship between the same two types these are qualified by a prefix to distinguish between them for example, ***ParentZoneRef***, ***FromPointRef***, ***ToPointRef***, etc.
- One-to-many relationships that are implemented as collections are lower cased and are always plural, for example, ***stopPoints***, ***vehicleJourneys***; *thus, the relationship name is used as a wrapper tag*. Such collections may variously comprise sets of simple versioned references (e.g. multiple instances of ***ScheduledStopPointRef***) or of embedded elements whose full defining is included inline within the enclosing element (e.g. multiple instances of ***ScheduledStopPoint***).
- Data types are shown in *italic*; built-in xml types are shown in *italic* and *lowerCamelCase*, usually with the *xsd* namespace prefix, for example, *xsd:dateTime*, *xsd:normalizedString*, *xsd:integer*. NeTEx data types are shown in *italic* and *UpperCamelCase*, without a prefix. e.g. *LengthType*, *EmailType*, *DirectionTypeEnumeration*.
- The NeTEx data types that are complex (and hence have a further definition elsewhere) are shown with an underline, e.g. *DataManagedObject*, *Line*, etc. An exception is made for the common utility types such as *MultilingualString*, and all subtypes of *VersionOfObjectRef*, which are shown without an underline.
- All compound names of data types are camel cased without a space, for example, *ScheduledStopPointIdType*.
- Stereotype names are enclosed in guillemets (chevrons), for example, «TM VIEW», «enum», «FK».

4.2 Additional EPIP presentation conventions

A few additional conventions have been added beyond those used in NeTEx in order to cover specific aspects of EPIP:

(1) Indication of which attributes are included in the profile.

EPIP uses highlighting to mark up the NeTEx model definitions as follows:

Table 1 – Additional presentation convention

«FK»	<i>dataSourceRef</i>	<i>DataSourceIdType</i>	0:1	Data system which originated data instance. ¹
	<i>created</i>	<i>xsd:dateTime</i>	0:1	Date and time of creation of ENTITY. This is a specific addition (rule, constraint, etc.) for the profile. ²
1 No highlight means that it is identical or similar (same meaning) to the original NeTEx text 2 Light Grey highlight emphasizes an additional rule or constraint added to the original NeTEx text				

(2) Profile code values

Wherever possible, the EPIP standardises the code values used to classify and characterise elements to a systematic set of documented values. In NeTEx, several different mechanisms are used to specify the permitted code values;

- Fixed enumerations** defined as part of the NeTEx schema (see presentation convention above e.g. '*riverBus*'). The EPIP mandates a subset of the NeTEx codes - see Annex B for the permitted values.
- Specialisations** of TYPE OF VALUE, used to define open ended code sets which may be added to over time without changing the schema, and for arbitrary user classifications, for example, to

record legacy entity classifications. EPIP itself uses the TYPE OF VALUE mechanism in a few cases to specify additional standardised codes: these are assigned to an “*epip_metadata*” CODESPACE (<https://netex-cen.eu/epip>) indicated by an ‘*epip*’ prefix. The identifiers of EPIP values are shown in lower case italics (if necessary, with underscore marks between words), and appear within italics in text, for example, ‘*epip:monomodal*’.

- c) **TypeOfFrame** instances: The EPIP uses several TYPEs OF FRAME to specify the usage of VERSION FRAMEs in the profile. The identifiers of the **TypeOfFrame** instances are shown in upper case italics with underscore marks between words, for example, ‘*epip:EU_PL_LINE_OFFER*’.

(3) Indication of Abstract classes

NeTEx makes extensive use of class *inheritance*; this significantly simplifies specification by avoiding repetition since shared attributes are declared on a superclass and subclasses are then defined that have the common properties of the superclass without having to repeat them. Most superclasses are ‘abstract’ – that is, there is no actual instance of them; only the terminal subclasses are ‘concrete’.

An inconvenience of inheritance is that if one wishes to understand the properties of any single concrete class, one must look also at all of its superclasses (and it is not always obvious whether this should best be done before or after studying the concrete class). For this reason, the EPIP profile includes the necessary abstract classes to understand the concrete classes, even though the concrete classes are never directly instantiated in a NeTEx document.

- The superclasses are flagged in headings by the suffix ‘**(Abstract)**’ and are generally presented in top down order of inheritance (so they will appear before the concrete class of actual interest).
- In UML diagrams (as for NeTEx and Transmodel), the names of abstract classes are shown in *italics*.
- Certain superclasses are technically not abstract in the NeTEx, but are not used as concrete classes in EPIP. These are flagged **(Abstract in EPIP)** In this document

(4) Subcomponent Classes

A number of EPIP classes have subcomponents that make up their definition. These provide ancillary detail (for example **AlternativeText**, **AlternativeName**, **TrainComponent**) and are flagged in headings by the suffix ‘**(Subcomponent of XXX)**’, where XXX is their usual parent. They can be ignored on a first reading.

(5) Emphasis of key points to note

In this document, underscore is used to emphasise important points (since italic and bold emphasis is mostly used to distinguish XML elements).

4.3 Transmodel definitions and EPIP

The NeTEx specification uses the same formal definitions to describe its XML class elements as those used by Transmodel to describe the corresponding ENTITIES.

The EPIP adheres to these definitions but, since it is intended to be as readable as possible, in certain places the EPIP specification simplifies, annotates, reorders or paraphrases the formal definitions in the

interests of clarity or brevity. In particular in its narrative it adds comments and examples as to the purpose and intended use of specific features. It also makes similar revisions to the NeTEx definitions of attributes. Regardless of this, the original Transmodel definitions (For which see Annex A) should be considered definitive for the interpretation of concepts and the NeTEx specification definitive for the interpretation of additional attributes.

Note that for readability, the EPIP specification also uses a different order of presentation of the elements compared to that used in Transmodel and NeTEx.

5 Framework

5.1 Introduction

In order to simplify implementation, all NeTEx objects use uniform mechanisms for the common features that are needed for effective data exchange, such as versioning, validity, ascribing responsibility and grouping sets of data.

This section describes the common NeTEx Framework that is shared by all NeTEx functional parts in the profile.

5.2 DataManagedObject, Version and Codespace

5.2.1 Overview

DataManagedObject, together with **Version** and **Codespace**, provides uniform, common properties for all NeTEx entities. **DataManagedObject** is the abstract “upper object” of the NeTEx implementation and all first class NeTEx entities are specialisations of it.

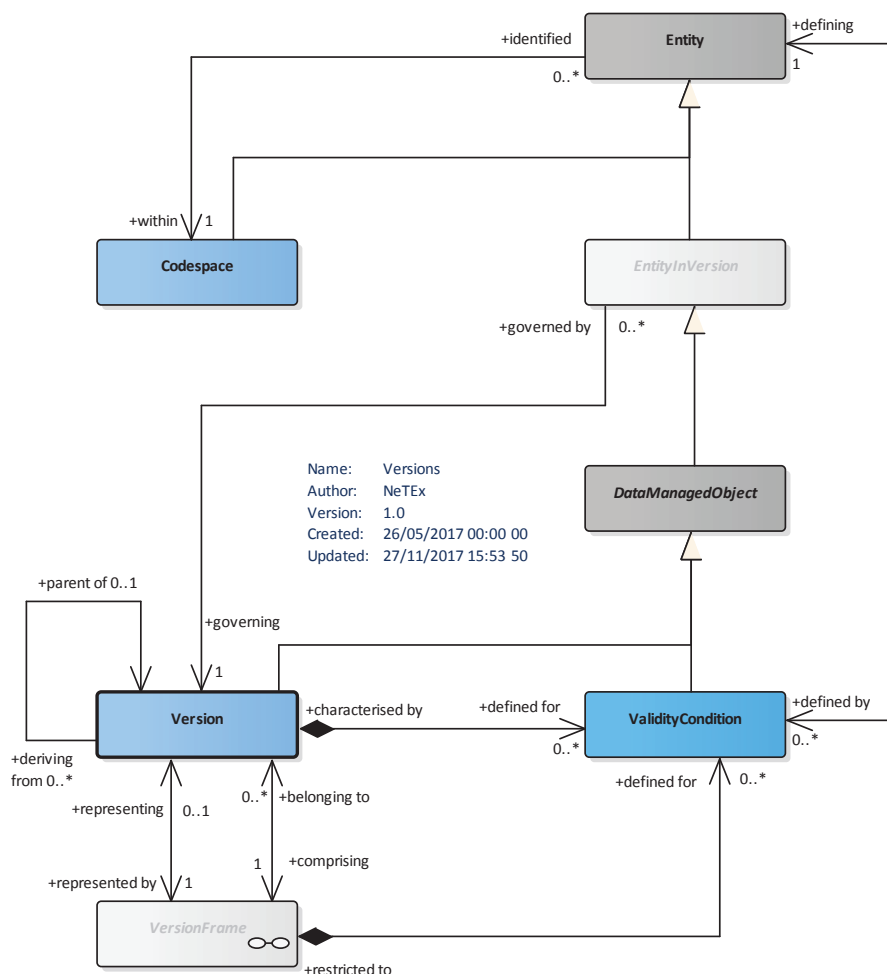


Figure 1 — Versions and Codespaces (physical model) (UML)

In Transmodel, an ENTITY represents an actual object in a repository having an instance of data present in an exchanged data set. Several versions, usually successive, of an ENTITY can be defined; normally a given data set will contain one particular version (but it is also possible to have multiple versions present in the same data set if required).

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 Transmodel's ENTITY IN VERSION. That is to say, data elements in a NeTeX XML document represent a particular version of each 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 in the repository is changed, two successive exports would result in different states, i.e. different instances of the appropriate ENTITY IN VERSION.

In NeTeX, ENTITY IN VERSION is implemented as **EntityInVersion**; this is further specialised as **DataManagedObject** which also reifies certain other separate Transmodel concepts into a single abstract XML class. It provides a convenient way to bring together common versioning, responsibility, and validity condition features from Transmodel, together with the XML extension mechanisms

supported by NeTEx, in a way that is completely uniform for all NeTEx objects and so is simpler to implement.

NeTEx uses **Codespaces** to ensure that the identifiers of instances of elements in an XML document are unique, even if they come from many different sources – see **5.2.8-Codespace** and also **8.4-Codespaces and the structure of identifiers**.

A **ValidityCondition** is used to attach conditions to elements to indicate when they are current or the circumstances in which they should be used (e.g. “In winter”). VALIDITY CONDITIONS can be attached to specific elements and also, through VERSION FRAMEs (see below), to whole sets of objects, so that it is possible to be precise about the conditions governing the coherence and relevance of data.

The **DataManagedObject** allows the assignment of a **Version** and responsibility attributes to an **EntityInVersion** and also one or more instances of a **ValidityCondition**.

5.2.2 Entity (Abstract)

An ENTITY represents an actual object, instances of versions of which are present in an exchanged data set.

Table 2 – Entity – XML Element

Classification	Name	Type	Cardinality	Description
«PK»	id	<i>ObjectIdType</i>	1:1	Identifier of ENTITY. Unique within CODESPACE. Abstract – always use the specific concrete identifier. See 8.4-Codespaces and the structure of identifiers for coding rules.

5.2.3 EntityInVersion (Abstract)

A specific version of an ENTITY, as found in a NeTEx document. The NeTEx **EntityInVersion** implements ENTITY IN VERSION by adding VERSION tracking attributes and VALIDITY CONDITIONS (see later below) for any NeTEx element.

Table 3 – EntityInVersion – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u>Entity</u>	::>	ENTITY ON VERSION inherits from ENTITY.
«FK»	dataSourceRef	<i>DataSourceIdType</i>	0:1	Data system which originated data instance.
«atr»	created	<i>xsd:dateTime</i>	0:1	Date and time of creation of ENTITY.
«atr»	changed	<i>xsd:dateTime</i>	0:1	Date and time of last change to ENTITY.

Classification	Name	Type	Cardinality	Description
«enum»	modification	<i>ModificationEnum</i>	0:1	Nature of modification. Enumerated value: <i>new</i> <i>revise</i> <i>unchanged</i> If unspecified, assume value from parent of containing frame.
«PK»	version	<i>VersionIdType</i>	0:1	VERSION number of this instance of the ENTITY IN VERSION.
«enum»	status	<i>VersionStatusEnum</i>	0:1	Status of ENTITY in VERSION. Default is 'active'. Only 'active' is supported in EPIP
«cntd»	b ValidBetween	<u><i>ValidBetween</i></u>	0:1	OPTIMISATION. Simple version of a VALIDITY CONDITION. Comprises a simple period. NO UNIQUENESS CONSTRAINT.

5.2.4 DataManagedObject (Abstract)

A **DataManagedObject** is a subclass of **EntityInVersion** that implements the common properties of all first-class NeTeX elements, such as:

- A **ResponsibilitySet**, specifying data management and stakeholder roles for the object.
- A **KeyList**: an optional set of an arbitrary list of key/value pairs to provide a simple extensibility mechanism. **KeyList** will not be used in the European profile (however it remains available for local extensions).
- An **Extensions** tag that allows the association of arbitrary user defined XML elements with an element. **Extensions** shall **not** be used within the European profile.
- A reference to a **Branding** for the ENTITY to attribute a marketing content for presenting the element.
- **AlternativeText** a set of alternative variants of any textual attribute of the ENTITY (the main purpose being translations into other national languages).
- **DataManagedObject** is an implementation artefact and does not appear in Transmodel.

Table 4 – DataManagedObject – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u><i>EntityInVersion</i></u>	::>	DATA MANAGED OBJECT inherits from ENTITY IN VERSION.
«FK»	responsibilitySetReference	<i>ResponsibilitySetIdType</i>	1:1	Reference to a RESPONSIBILITY SET defining ownership and management responsibilities for objects.

Classification	Name	Type	Cardinality	Description
«cntd»	keyList	<u>KeyValue</u>	0:*	A list of KeyValue pairs that may be associated with the object. This allows simple arbitrary user extensions.
«FK»	BrandingRef	<i>BrandingRef</i>	0:1	Reference to a BRANDING. (A TypeOfValue) with no added attributes
«cntd»	alternativeTexts	<u>AlternativeText</u>	0:*	Additional Translations of text elements.

5.2.5 AlternativeText (Subcomponent of DataManagedObject)

It is sometimes necessary to provide several variants of a name or other descriptive text, especially if the information is required in several national languages. The **AlternativeText** is a generic way of providing such variants for any textual attribute of a **DataManagedObject**. It can be seen as a complement to the **AlternativeName** mechanism (described later on), and can be used for any name, description or other text.

NOTE The **AlternativeName** element (rather than an **AlternativeText**) is expected to be used for aliases (e.g. “King’s Cross”; “London King’s Cross” “King’s Cross Railway Station”) of proper names, whereas **AlternativeText** is intended primarily for translations (e.g. “en.London”, “fr.Londres”, “it.Londra”, “cn.倫敦”, “ge.ლონდონი”, etc).

In EPIP, **AlternativeText** should always be embedded inline within the parent element.

Table 5 – AlternativeText – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>VersionedChild</u>	::>	AlternativeText inherits from VERSIONED CHILD.
«PK»	attributeName	<i>xsd:NCName</i>	<u>0:1</u>	Name of text attribute for which this is the alternative text. Shall be an existing attribute name
«PK»	useForLanguage	<i>xsd:language</i>	<u>0:1</u>	Name of language for which this is to be used.
«atr»	order	<i>xsd:integer</i>	0:1	Order of name.
	Text	<i>MultilingualString</i> (Language + Text)	1:1	Variant of the text, in the specified language.

5.2.6 KeyValue (Subcomponent of DataManagedObject)

A **KeyValue** is a convenient way to exchange additional specific data attributes not supported in NeTEx. Lists of values can be associated with any **DataManagedObject**. Note that the interpretation and use of each **KeyValue** requires an additional specification of its meaning by the local systems that use them.

In EPIP, extensions to elements will be based only on the Key/Value mechanism (i.e. the Extensions tag will not be used). Furthermore, **KeyList** shall not be used to carry information that can be encoded elsewhere in NeTEx.

Table 6 – KeyValue – XML Element

Classification	Name	Type	Cardinality	Description
«PK»	<i>typeOfKey</i>	<i>xsd:normalizedString</i>	0:1	Type for KEY VALUE.
«PK»	<i>Key</i>	<i>xsd:normalizedString</i>	1:1	Key for KEY VALUE.
	<i>Value</i>	<i>xsd:normalizedString</i>	1:1	Value for KEY VALUE.

5.2.7 Version

NeTeX can support the exchange of data from systems with advanced versioning systems that track every change to every element - as recorded by a **version** attribute on **DataManagedObject**. However, many legacy systems do not support such versioning capabilities and so the provision of *fine-grained* tracking data in a NeTeX document is optional in EPIP. However frame level versioning is mandatory in EPIP.

In Transmodel, VERSIONs are themselves first class objects and can be given descriptions that explain the purpose and validity of the VERSION. These can be exchanged in NeTeX as **Version** instances.

In EPIP the **Version** element itself is not populated; a simple version number in the **version** attribute of an EPIP element is used to indicate the version.

5.2.8 Codespace

A CODESPACE is similar to the XML concept of a *Namespace* that gives a context within which the names of elements and attributes in an XML schema shall be unique, (using the generic W3C domain mechanism), thus permitting the combining of elements from different schemas in the same aggregated model. In a similar way, NeTeX uses **Codespaces** to ensure that the identifiers of instances of elements in an XML document are unique, even if they come from many different systems and are placed in the same document – see examples below. This allows data elements from different countries, different regions, and different organisations to be combined without clashes.

Having an explicit **Codespace** object also makes it possible for applications (such as a profile validator) to reflect on and compute over the elements within **Codespaces** in a generic manner (i.e. they can determine which coding system is being used for a given object and use this knowledge to inform the way they process data). The **Codespace** is thus used for three purposes:

1. To give a context within which to identify objects uniquely.
2. To indicate to consumer systems which identifier system to use to interpret codes (if they care).
3. To manage the distributed allocation of new identifiers to new instances of objects.

The NeTeX schema includes XML uniqueness and referential integrity constraints that (a) check that each identifier is unique and (b) that any versioned reference to an entity is satisfied (i.e. that the referenced element is also present elsewhere in the document). The checks can be applied automatically using any XML validator to any NeTeX document and are fundamental for establishing a basic quality level for data exchange. See section 8.15.

In the context of EPIP, CODESPACES prevent identifier clashes among data provisions coming from different countries (and regions, authorities and operators within countries). Different code structures

may be used within different CODESPACES – see **8.4-Codespaces and the structure of identifiers** for further discussion.

In order to manage a uniform coding system for use by many stakeholders, it may be desirable to partition a CODESPACE into different ADMINISTRATIVE ZONES, based on spatial, modal or other criteria, each with a responsible ORGANISATION. Within the CODESPACE, the use of a specific prefix or prefix range value may then be reserved for the allocation of identifiers to new entities falling under the criteria of that zone. This allows the issuing of new codes to be delegated to the managers of the zone on a distributed basis. The central registrar function is needed only for setting up the zones and issue the prefixes in the first place. ADMINISTRATIVE ZONES are not exchanged directly in the strict EPIP, but can be referenced with an external reference from a **ResponsibilitySet** (see 5.2.11)

A **Codespace** is specified as a path expression by using an internet domain, obtained through IANA (the Internet Assigned Numbers Authority), which provides a process for registering a domain as globally unique. For example, 'tfl.gov.uk', 'bahn.de', 'ratp.fr', 'foo.com', or 'sbb.ch'. The path can be declared in the `<XmlnsUrl>` attribute of a **Codespace** element.

Each **Codespace** has an associated *prefix* which may be used as a surrogate for the codespace throughout a document and is declared on a **VersionFrame** using a **Codespace** element. These prefixes provide an efficient way of encoding unique identifiers in an XML document in that the short prefix can establish a unique context for each identifier on every element *id* in a document (rather than say, a long qualifying string or complex code structure), even if the identifiers come from heterogenous coding systems. The prefix does not have to be the same as the *id* of the **Codespace** (For example, the examples in this document mostly use a prefix of 'epd' for the 'epip_data' **Codespace** and a prefix of 'epip' for the 'epip_metadata' **Codespace**. Technically the prefix is local to the document and different prefixes can be used for the same **Codespace** in different documents, but it is helpful to data consumers to use the same prefix values consistently.

Within a codespace, a data provider may use any system of coding identifiers they like; the only requirement is to provide persistent unique identifiers for each separate entity instance within the codespace.

XML EXAMPLE CODESPACE declarations:

```
<CompositeFrame version="1.0" id="mybus:Line1">
  <codespaces>
    <Codespace id="napt">
      <Xmlns>napt</Xmlns>
      <XmlnsUrl>http://naptan.org.uk/naptan</XmlnsUrl>
      <Description>UK NaPTAN</Description>
    </Codespace>
    <Codespace id="era_stops">
      <Xmlns>era</Xmlns>
      <XmlnsUrl>http://era.org.eu/stops</XmlnsUrl>
      <Description>European Rail Authority</Description>
    </Codespace>
  </codespaces>
</CompositeFrame>
```

Examples of elements that use of the above CODESPACES:

```
<StopPlace id="napt:4701234567">
  <StopPlaceRef ref="napt:4701234567">
    <StopPlace id="era:0018">, etc.
```

An application processing a document is expected to understand any necessary rules peculiar to validating or interpreting the identifiers in a specific **Codespace**. For example, NaPTAN stop identifiers in the UK NaPTAN codespace have the structure '999 0 XXXXXXXXX' where '999' is an area prefix, '0' is fixed and 'XXXXXXXXXX' is a number unique within the area, whose length and nature depends on the mode of transport. An XML validator will not enforce the format rules, but an implementation that is aware of the CODESPACE and its rule system (as indicated by the domain url) is able to do so programmatically if it is relevant to do so (for example a system that is merely aggregating data to pass it onto others might be agnostic about the structure of identifiers). For the general purposes of emitting or parsing a NeTEx document, any such local rules are generally irrelevant; all that matters is that each entity instance can be uniquely identified.

Table 7 – Codespace – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>Entity</u>	::>	CODESPACE inherits from ENTITY.
«PK»	id	<i>CodespaceIdType</i>	1:1	Identifier of CODESPACE. Unique within documents.
«AK»	Xmlns	<i>xsd:NMTOKEN</i>	1:1	CODESPACE prefix, unique within a given XML document e.g. 'napt'
	XmlnsUrl	<i>xsd:anyURI</i>	0:1	CODESPACE path. Globally unique. For example, http://naptan.org.uk/naptan or http://vdv.de/vdv/haltstelle/
	Description	<i>xsd:string</i>	0:1	Description of CODESPACE.

5.2.9 ValidityCondition

A **ValidityCondition** can be used to specify the conditional validity of a **DataManagedObject**.

EPIP restricts use of VALIDITY CONDITIONS to just a simple name (the full NeTEx specification allows complex, value dependent conditions to be constructed).

Table 8 – ValidityCondition – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	Inherits from DATA MANAGED OBJECT. Note that use of most of the inherited values is suppressed.
«PK»	id	<i>ValidityConditionIdType</i>	1:1	Identifier of VALIDITY CONDITION.
	Description	<i>MultilingualString</i>	0:1	Description of VALIDITY CONDITION. See comment on the use of <i>partial</i> for <i>AccessibilityLimitation</i>

5.2.10 ValidBetween

ValidBetween is a simplified view of a **ValidityCondition**, having only a start date and end date. A start or end date may be omitted if unknown (thus specifying any time up to the **ToDate**, or any time after the **FromDate**), but at least one of them shall be provided.

A **FromDate** shall be specified on a **ValidBetween** instance specifying the dominant condition on an EPIP VersionFrame.

Table 9 – ValidBetween – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	Inherits from DATA MANAGED OBJECT. NB none of the inherited attributes are usable.
	FromDate	xsd:dateTime	0:1	Start date of AVAILABILITY CONDITION. Date is INCLUSIVE.
	ToDate	xsd:dateTime	0:1	End of AVAILABILITY CONDITION. Date is INCLUSIVE.

5.2.11 ResponsibilitySet

A **ResponsibilitySet** can be specified for a **DataManagedObject** to specify which organisations are responsible for different data management roles, such as creating, maintaining, distributing or licencing the data.

In EPIP, a **ResponsibilitySet** is normally used at a VERSION FRAME level to apply to all the elements in the frame, rather than on individual objects (though the latter is possible).

Table 10 – ResponsibilitySet – XML 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.
«AK»	PrivateCode	PrivateCode	0:1	A private code that uniquely identifies the element. May be used for inter-operating with other (legacy) systems.
«cntd»	roles	<u>ResponsibilityRoleAssignment</u>	1:*	RESPONSIBILITY ROLE ASSIGNMENTs making up the RESPONSIBILITY SET.

5.2.12 ResponsibilityRoleAssignment (Subcomponent of ResponsibilitySet)

A **ResponsibilitySet** is made up of one or more **ResponsibilityRoleAssignment** instances. Each RESPONSIBILITY ROLE ASSIGNMENT assigns 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 11 – ResponsibilityRoleAssignment – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>VersionedChild</u>		RESPONSIBILITY ROLE ASSIGNMENT inherits from VERSIONED CHILD.
«PK»	id	<i>ResponsibilityRole-AssignmentIdType</i>	1:1	Identifier of RESPONSIBILITY ROLE ASSIGNMENT.
«FK»	ResponsibilitySetRef	<i>ResponsibilitySetRef</i>	0:1	Reference to a RESPONSIBILITY to which this RESPONSIBILITY ROLE belongs.
	Description	<i>MultilingualString</i>	0:1	Description of RESPONSIBILITY ROLE.
«enum»	DataRoleType	<i>DataRoleTypeEnum</i>	0:1	Data Role type of RESPONSIBILITY ROLE. <i>creates</i> <i>collects</i> <i>aggregates</i> Only a subset of data related roles are selected for the EPIP, these distinguish between an ORGANISATION originating the data (<i>creates</i>); one collecting them from multiple sources but without changing them (<i>collects</i>) and one which integrates data sets (merging, correcting, enhancing, etc.) (<i>aggregates</i>)
«enum»	StakeholderRoleType	<i>StakeholderRoleTypeEnum</i>	0:1	Stakeholder role type of RESPONSIBILITY ROLE. <i>Planning</i> <i>Operation</i> Only a subset of data roles are selected for the EPIP, these distinguish between an ORGANISATION that does 'Planning' (typically an Authority); one that operate 'Operation' some aspect of the network (Operators, site managers, etc.); and one having the legal ownership (city, authority, etc.). This information is of relevance for passengers, especially if they have questions or complaints to address.
«FK»	Responsible-OrganisationRef	<i>OrganisationRef</i>	0:1	Reference to an ORGANISATION to which this RESPONSIBILITY ROLE is assigned.
«FK»	ResponsibleAreaRef	<i>AdministrativeZoneRef</i>	0:1	Reference to an ADMINISTRATIVE ZONE to which this RESPONSIBILITY ROLE is assigned.

5.2.13 DataSource

An ENTITY is linked to a DATA SOURCE, which is used to identify the system that originated the version of the ENTITY in the exchanged data set. Each ENTITY may have its own DATA SOURCE: for example,

STOP PLACES coming from the consolidated UK NAPTAN National registry of stops might all be declared as sourced from 'nptg:NaPTAN', so that the XML tag for a STOP PLACE might read:

```
<StopPlace version="001" dataSourceRef="nptg:NaPTAN" id="napt:910GWIMBLDN">.
```

Usually all the data in a data set comes from the same source so the **DataSource** is specified at the VERSION FRAME level using the **DefaultDataSource** attribute on the **VersionFrame / FrameDefaults**.

DataSource is a specialisation of **TypeOfValue**. But has no additional attributes of its own. See 5.5.2 below.

5.3 Relationships and entity references

5.3.1 General rules

In order to turn any computer-based representation of a data set (as an object model or database) into an EPIP conformant NeTEx data file that can be exchanged, it shall be serialised by the producer system into an XML document – that is, a stream of characters grouped within XML tags. This requires in particular that the producer system encodes the relationships between entities (on the computer system represented by object pointers or keys in a database) in a way that can be correctly interpreted as the same relationships by any consumer system reading the XML document. This is typically done in two different ways:

1. *Containment* – Related entities are grouped within the body of the referencing element. This is appropriate for implementing **composition** relationships where there is an inherent hierarchy, as say in the **Codespace** example above, or for the contents of a **VersionFrame**.
2. *Reference* – The relationship is implemented by a 'reference' attribute containing the identifier(s) of the related element. This is appropriate for **association** relationships which link two entities. Depending on the semantics and **navigability** of the relationship, a reference may be held on an entity at one or both ends of the relationship.

XML EXAMPLES

For example, to serialise a **StopPlace** → **quays** composition relationship for two platforms of Wimbledon Station, containment within a **quays** tag is used to embed the child **Quay** instances inline;

```
<StopPlace version="8.6" id="napt:910GWIMBLDN">
  <quays>
    <Quay version="8.4" id="napt:910GWIMBLDN1">
      <Name>Platform 1</Name>
    </Quay>
    <Quay version="8.6" id="napt:910GWIMBLDN2">
      <Name>Platform 2 </Name>
    </Quay>
  </quays>
</StopPlace>
```

For example, to serialise a many-to-one **StopPlace** → **TopographicPlace** association for Wimbledon station, a versioned reference to the **TopographicPlace** is held on the **StopPlace**:

```
<StopPlace version="001" id="napt:910GWIMBLDN">
  <Name>Wimbledon Rail Station</Name>
  <TopographicPlaceRef version="any" ref="nptg:E0034695">Wimbledon</TopographicPlaceRef>
</StopPlace>
```

For one-to-many associations, where the referencing element holds multiple references, NeTEx always uses a wrapper tag based on the relationship name to group the references. For example, a **StopPlace / types** association that holds references to two different instances of **TypeOfZone**.

```
<StopPlace version="001" id="napt:910GWIMBLDN">
  <types>
    <TypeOfZoneRef version="any" ref="napt:Type_of_zone@GRLS"/>
    <TypeOfZoneRef version="any" ref="uicStationCat:476"/>
  </types>
</StopPlace>
```

NOTE For flexibility, in the NeTEx schema every dependent child element includes an attribute to hold a reference to its parent. If the parent is evident from the context, as say when nesting child elements within a parent, the reference to the parent does not need to be populated (and in the EPIP shall not be). For example, the **ParentZoneRef** shown here is **not** needed on the **Quay** instances because they are embedded inline within the parent **StopPlace**.

```
<StopPlace version="8.6" id="napt:910GWIMBLDN">
  <quays>
    <Quay version="8.4" id="napt:910GWIMBLDN1">
      <Name>Platform 1</Name>
      <ParentZoneRef version="8.6" ref="napt:910GWIMBLDN">NOT
NEEDED</ParentZoneRef>
    </Quay>
    <Quay version="8.6" id="napt:910GWIMBLDN2">
      <ParentZoneRef version="8.6" ref="napt:910GWIMBLDN">NOT
NEEDED</ParentZoneRef>
      <Name>Platform 2 </Name>
    </Quay>
  </quays>
</StopPlace>
```

5.3.2 VersionOfObjectRef

All NeTEx references are implemented not with a simple identifier value type but with a **VersionOfObjectRef** element which is specialised for each entity type (**LineRef**, **StopPlaceRef**, **ServiceJourneyRef**, etc.). This so as (a) to allow a version to be specified on the reference and (b) to enforce type correctness and facilitate the automatic validation of references using an XML validator.

Table 12 – VersionOfObjectRef – (Abstract)

Classifi- cation	Name	Type	Cardi- nalit y	Description
	NameOfRefClass	<i>NameOfClass</i>	0:1	Reference to object type of referenced ENTITY that is in the specified VERSION FRAME
«FK»	ref	<i>(ObjectIdType)+</i>	1:1	Reference to simple identifier of ENTITY that is in the specified VERSION FRAME. The <i>ObjectIdType</i> is specialised, e.g. <i>LineIdType</i> , <i>StopPlaceIdType</i> , etc.
ONLY ONE OF THE FOLLOWING (choice)				
«FK»	version	<i>VersionIdType</i>	0:1	Reference to VERSION of ENTITY that is in a VERSION FRAME of the current dataset. Mutually exclusive with

				versionRef. Use "any" if the VERSION is unknown (note that this will trigger NeTeX's XML automatic consistency check). The CODESPACE of the version and versionRef is assumed to be the same as that of the ref .
«FK»	versionRef	<i>VersionRef</i>	0:1	Identifier of the VERSION of the referenced entity. For use when reference is external (i.e. in another dataset) and a specific version is required. Mutually exclusive with version . Use "any" if the version is unknown. This will also make clear that the referenced object is external.

5.3.3 Integrity Checking of XML Documents

Missing or incorrect element references are a serious cause of error to a consuming system. NeTeX uses a powerful and very uniform approach to serialising relationships that allows the automated enforcement of referential integrity checks to ensure that references are complete and correct. References may be to a precise version of an object. The checking can be selectively relaxed on references to objects located in a separate document; thus, large data sets, (and also data sets that come from different producers), can be broken down into separate documents.

The distinction between internal (same document) and external object (separate document) references is made by choosing to use either the **version** attribute (*internal*) or **versionRef** attribute (*external*) on a given reference.

- References using the **version** attribute will be checked automatically by an XML validator working from the many constraints that are coded within the schema – i.e. are internal.
- This automatic consistency check is not done on references that use **versionRef**, which are treated as *external* (even if the referenced object is in fact in the same document).

For example, the following encoding of a **StopPlace** → **TopographicPlace** reference will be checked to ensure that elsewhere in the document there is a definition of the specified **TopographicPlace** with the identifier 'nptg:E0034695'

```
<TopographicPlaceRef version="2.2" ref="nptg:E0034695"/>
```

While, in this case the checking has been relaxed.

```
<TopographicPlaceRef versionRef="2.2" ref="nptg:E0034695"/>
```

For readability and to assist debugging, by convention it is helpful to include an annotation on external references with the tag content as to the nature of the reference (since this may not be obvious from the **ref** identifier itself), for example;

```
<TopographicPlaceRef versionRef="2.2" ref="nptg:E0034695">Wimbledon</TopographicPlaceRef>
```

For further details on the use of references in the EPIP see section 8.4

5.3.4 Identity uniqueness scope

All the specialisations of **DataManagedObject** have an **id**. Within a given document, the value of this shall be unique within the class of element. For example, there can be only one **SiteFrame** with an **id** of “foo” in a document. In most cases, different classes of element may have the same **id**; thus, for example it is technically possible (but not advisable) to have both a “foo **SiteFrame** and a “foo” **ServiceFrame** in the same document.

The uniqueness and integrity constraints for NeTEx elements are implemented by XML language features as part of the schema, and so can be automatically enforced by any XML Validator. Note, however, that due to certain technical restrictions on the way the XML key constraints work and are currently implemented, certain classes of related NeTEx elements are currently subject to a wider restriction of uniqueness (that is, uniqueness is shared across a common superclass). For example, PLACEs of any sort shall be unique, so it is not possible to have a **PointOfInterest** with **id**=‘foo’ and a **StopPlace** with **id**=‘foo’ in the same NeTEx conformant document.

NOTE If an identifier coding system includes a specific type in the code structure (as does the EPIP common code structure use with the ‘*epip_data*’ **Codespace**, – see 8.4.3) then all codes are inherently unique within their specific type.

Table 13 – NeTEx classes used in EPIP with a shared uniqueness scope

Supertype	EPIP Concrete classes	Comment
POINT	ScheduledStopPoint, TimingPoint	Useful to have common scope
LINK	ServiceLink, RouteLink, PathLink,	Unnecessarily over restrictive
ZONE & PLACE	Topographic Place, Address Tariff Zone StopPlace, Quay, StopPlaceEntrance, PointOfInterest,	Unnecessarily over restrictive
LINK SEQUENCE	ServicePattern, JourneyPattern	Unnecessarily over restrictive
LINE	Line, FlexibleLine	Useful to have common scope
JOURNEY	ServiceJourney, TemplateServiceJourney	Useful to have common scope
ORGANISATION	Operator, Authority	Useful to have common scope
FACILITY SET	SiteFacilitySet, ServiceFacilitySet	Useful to have common scope

The identifiers of some child elements are further qualified by the use of an ***order*** attribute; they can thus share the same ***id*** value as their parent. Thus, for example ServicePattern ‘foo’ may have children that are children uniquely identified as follows:

```
<ServicePattern id="foo" version="1.1">
  <StopPointInSequence id="foo" version="1.1" order="1">
  </StopPointInSequence>
  <StopPointInSequence id="foo" version="1.1" order="2">
  </StopPointInSequence>
  <StopPointInSequence id="foo" version="1.1" order="3">
  </StopPointInSequence>
Etc.
</ServicePattern>
```

In EPIP within an XML document, elements with an ***order*** attribute shall be serialised in sequence in accordance with the ascending value of their order attribute.

NOTE the uniqueness and existence constraints are declared in the *NeTeX_publication.xsd* schema file and have the effect that all declarations of NeTeX elements shall include an id and a version attribute in the start tag, even if it is just a dummy value (by convention ‘any’).

5.4 Frames

5.4.1 Overview

A **VersionFrame** provides a versionable container that allows a coherent set of related elements to be exchanged as a set, for example the timetable for a line, or the railway stations for a country.

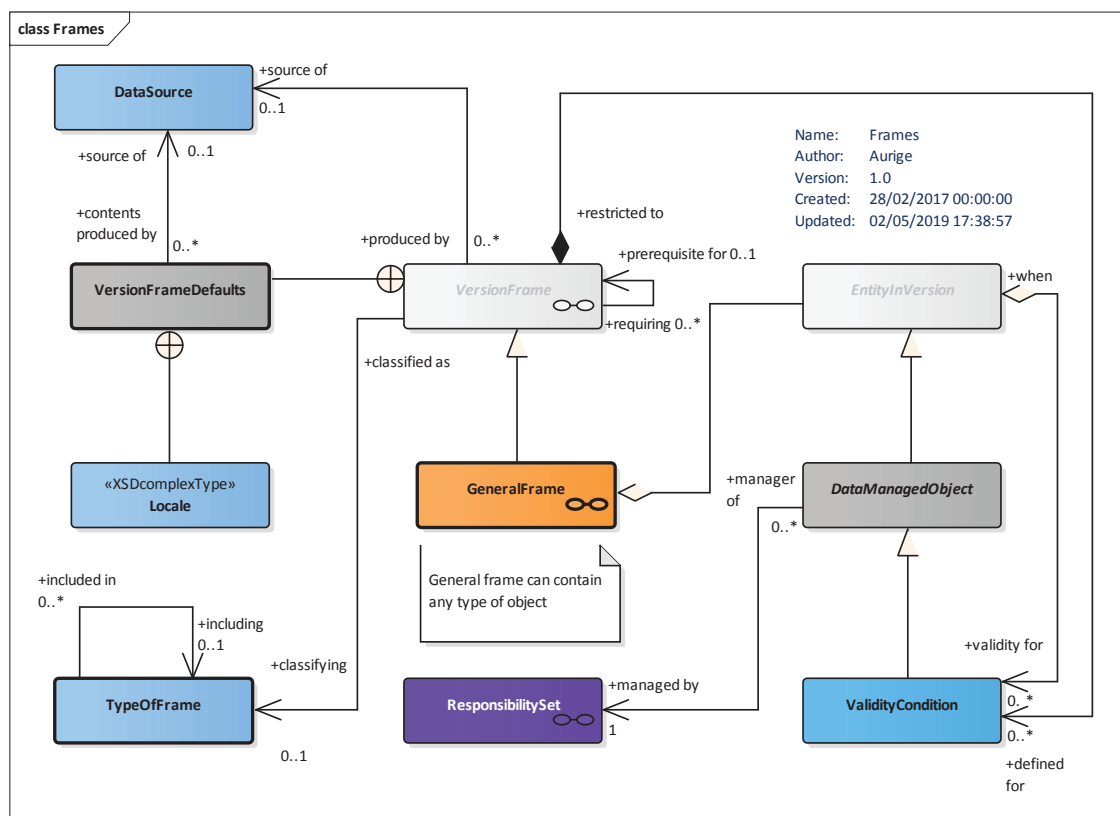


Figure 2 — Version Frames (UML)

Since they embody a coherent set of related elements (that is a complete set of elements. all of compatible version), frames are strongly involved in version management. Each **VERSION FRAME** instance also has one or more **ValidityConditions** attached to it (since it is a **DataManagedObject** in its own right and has a **validityConditions** relationship) that is used to express the validity of the frame and its contents as a whole See **8.1-Version management** for more details.

NOTE These frame validity conditions should not be confused with **contentValidityConditions** which are more specific validity conditions that are applied to multiple elements within the frame. EPIP requires that **VALIDITY** conditions that apply to individual elements are embedded within the specific element to which they relate.

VersionFrame itself is abstract: NeTeX provides a set of specialised 'specific' concrete frames for specific purposes (**ServiceFrame**, **SiteFrame**, etc.) each covering a functional subset of the NeTeX data elements; as well as a general purpose **GeneralFrame** that can contain any type of ENTITY IN VERSION.

A **VersionFrame** can be assigned a **TypeOfFrame**; an arbitrary user defined classification (whose use is reserved within EPIP to a predefined set of values).

5.4.2 Version Frame (Abstract)

Regardless of their specific contents, all types of **VersionFrame** have a number of common properties, such as a name, codespaces, etc.

Table 14 – VersionFrame – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	VERSION FRAME inherits from DATA MANAGED OBJECT. NOTE the "Dominant" VALIDITY CONDITION for the FRAME content is attached to the FRAME
«PK»	id	<i>VersionFrameIdType</i>	1:1	Identifier of VERSION FRAME.
	Name	<i>MultilingualString</i>	0:1	Name of VERSION FRAME.
	Description	<i>MultilingualString</i>	0:1	Description of VERSION FRAME.
«FK»	TypeOfFrameRef	<i>TypeOfFrameRef</i>	0:1	Reference to TYPE of VERSION FRAME.
«FK»	BaselineVersion-FrameRef	<i>VersionRef</i>	0:1	Previous baseline version frame that objects in this frame require.
«cntd»	codespaces	<u>Codespace</u>	0:*	CODESPACES used in this frame. Normally there will be at least one. A default may be specified in the Frame defaults.
«cntd»	FrameDefaults	<u>FrameDefaults</u>	0:1	Set of default values to assume for values in the frame if not explicitly stated on individual elements.
«cntd»	prerequisites	<i>VersionFrameRef</i>	0:*	References to other version frames on which this frame depends. +v1.1

Classification	Name	Type	Cardinality	Description
«cntd»	contentValidity-Conditions	<u>ValidityCondition</u>	0:*	Shared VALIDITY CONDITIONS used by elements in the FRAME. NOTE this is <u>not</u> the dominate FRAME validity condition.

5.4.3 TypeOfFrame

A TYPE OF FRAME classifies a VERSION FRAME. It provides a rudimentary form of metadata to describe the permitted contents of a profile (A profile validator may use such data to further check the contents of a document).

The EPIP defines a number of standardised **TypeOfFrame** instances; each one classifies a particular data set to be exchanged under the EPIP and shall be assigned to the appropriate version frame in any EPIP conformant NeTEx document. All frame identifiers are defined within the EPIP **Codespace** with a prefix of 'epip:'.

The list of frame types in EPIP is:

- '**EU_PI_COMMON**': common and generic object types possibly used across multiple other frames (for example: **TypeOfValue**, **DayType**, etc.)
- '**EU_PI_STOP**': all objects necessary for STOP PLACE and SITE descriptions (STOP PLACES, QUAYS, ENTRANCES, POINTS OF INTEREST, etc.).
- '**EU_PI_NETWORK**': all objects necessary for NETWORK descriptions (LINES, ROUTES, JOURNEY PATTERNS, etc.).
- '**EU_PI_TIMETABLE**': all objects related to timing information (VEHICLE JOURNEYS, PASSING TIMES, etc.)
- '**EU_PI_CALENDAR**': all objects related to calendars and types of day.
- The '**EU_PI_STOP**', '**EU_PI_NETWORK**', and '**EU_PI_TIMETABLE**' frame types carry the main "payload" data content for an EPIP exchange. The '**EU_PI_COMMON**', and '**EU_PI_CALENDAR**' frames types carry ancillary data referenced by the payload frames.

A **CompositeFrame** is a specific type of frame that contains a set of VERSION FRAMEs to which the same VALIDITY CONDITIONS have been assigned. It is in effect a grouping mechanism. The EPIP specification defines three composite frames with which to group the payload and ancillary frames:

- '**EU_PI_LINE_OFFER**': gathering all the information related to a single LINE; it shall contain '**EU_PI_COMMON**', '**EU_PI_STOP**', '**EU_PI_NETWORK**', '**EU_PI_CALENDAR**' and '**EU_PI_TIMETABLE**' specific frames.
- '**EU_PI_NETWORK_OFFER**': gathering all the information related to a single NETWORK (so multiple lines); it shall contain '**EU_PI_COMMON**', '**EU_PI_CALENDAR**', '**EU_PI_STOP**', '**EU_PI_NETWORK**' and '**EU_PI_TIMETABLE**' specific frames. There may be multiple instances of a given frame type, for example a separate, '**EU_PI_NETWORK**' and a '**EU_PI_TIMETABLE**' for each line.
- '**EU_PI_STOP_OFFER**': gathering all the information related to a STOPs for area; it shall contain '**EU_PI_COMMON**' and '**EU_PI_STOP**' specific frames.

A detailed description of these types of VERSION FRAME is provided in 8.10.

One further **TypeOfFrame** 'EU_PI_METADATA' can be included in an EPIP document to provide a set of static metadata references that can be used variously by both an XML validator and a parameterised profile validator. If present, internal references can be used for the **TypeOfFrameRef** instances, etc; otherwise these shall be external references that place the version number in a **versionRef** element.

Table 15 – TypeOfFrame – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>TypeOfValue</u>	::>	TYPE OF FRAME inherits from TYPE OF VALUE.
«PK»	id	TypeOfFrameIdType	1:1	Identifier of TYPE OF FRAME. Shall be one of the following: EU_PI_COMMON, EU_PI_STOP, EU_PI_NETWORK, EU_PI_TIMETABLE, EU_PI_LINE_OFFER, EU_PI_NETWORK_OFFER, EU_PI_CALENDAR
«cntd»	classes	ClassInContextRef	0:*	Classes to be present in frames conforming to the TYPE OF FRAME.
«cntd»	includes	TypeOfFrameRef	0:*	TYPEs OF FRAME contained in TYPE OF FRAME. Shall not be cyclic.

5.4.4 Specific frames and General Frames

VersionFrame is specialised to make 'Specific' frames (**SiteFrame**, **ServiceFrame**, etc.) for different functionally related data sets. There is also a **GeneralFrame** which can contain any type of object. They are very similar in concept, except that specific frames only host a limited set of object types, and furthermore group them by separate type (all the LINES together, all the STOP PLACES together, etc.) whilst a **GeneralFrame** places all objects in a single heterogenous list regardless of type.

EPIP will only use the **ResourceFrame**, **SiteFrame**, **ServiceFrame**, **TimetableFrame** and **ServiceCalendarFrame** specific frames, but for brevity the actual frame descriptions are omitted – see the summary descriptions of the permitted contents of 'EU_PI_COMMON', 'EU_PI_STOP', 'EU_PI_NETWORK', 'EU_PI_CALENDAR', and 'EU_PI_TIMETABLE' in 8.10.

In the EPIP, a given type of NeTEx object is only permitted in within one single type of frame.

Table 16 – GeneralFrame – XML 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 (GeneralFrame~Member)

5.4.5 Composite Frame

A **CompositeFrame** is a specific type of frame that contains a set of VERSION FRAMEs to which the same VALIDITY CONDITIONS have been assigned. It is in effect a grouping mechanism that allows different types of specific frame to be combined as a single data set.

The EPIP specification defines two TYPEs OF FRAME for **CompositeFrame**: 'EU_PI_LINE_OFFER' and 'EU_PI_NETWORK_OFFER'; see the summary descriptions of their permitted contents in 8.10.

Table 17 – CompositeFrame – XML Element

Classification	Name	Type	Cardinality	Description
::>	::>	<u>VersionFrame</u>		COMPOSITE FRAME inherits from VERSION FRAME
«PK»	id	<i>CompositeFrameIdType</i>	1:1	Identifier of COMPOSITE FRAME.
«cntd»	frames	<u>VersionFrame</u>	1:*	Frames contained in COMPOSITE FRAME. Shall contain only specific frames of type 'EU_PI_COMMON', 'EU_PI_STOP', 'EU PI CALENDAR', 'EU_PI_NETWORK' and 'EU_PI_TIMETABLE'.

5.4.6 FrameDefaults

The **FrameDefaults** specify default values for certain common properties of elements in a frame, such as the **Codespace**, **DataSource**, time-zone, etc. The defaults are applied to all elements in the frame for which an explicit value has not been specified. The use of defaults can both simplify export and reduce the size of documents.

Table 18 – FrameDefaults – XML Element

Classification	Name	Type	Cardinality	Description
«FK»	DefaultCodespaceRef	<i>CodespaceRef</i>	0:1	Default CODESPACE to assume for an identifier that does not have an explicit CODESPACE specified.
«FK»	DefaultDataSourceRef	<i>DataSourceRef</i>	0:1	DATA SOURCE to use for elements in the frame which do not have a DATA SOURCE specified.
«FK»	DefaultResponsibilitySetRef	<i>ResponsibilitySetRef</i>	0:1	RESPONSIBILITY SET to use for elements in the frame which do not have a RESPONSIBILITY SET specified.
«cntd»	DefaultLocale	<u>Locale</u>	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.
	DefaultLocationSystem	<i>xsd:normalizedString</i>	0:1	Default LOCATION SYSTEM to use for locations in the frame which do not have a LOCATION SYSTEM

Classification	Name	Type	Cardinality	Description
				specified. For EPIP WGS84 is recommended.
«enum»	DefaultSystemOfUnits	<i>SystemOfUnitsEnum</i>	0:1	Default SYSTEM OF UNITS to use for dimensional attributes in the frame (<i>DistanceType</i> , <i>LengthType</i> , <i>WeightType</i> , etc) for which no units have been specified. <i>SiMetresAndKilometres</i> (only value accepted in EPIP)
	DefaultCurrency	<i>CurrencyType</i>	0:1	Default CURRENCY to use for amount attributes in the frame for which a currency has not been specified.

5.5 Generic elements

5.5.1 Introduction

In order to achieve a simpler, uniform implementation, NeTEx model elements are based on a framework (taken from Transmodel) of abstract generic types that are specialised to create the concrete XML elements. The generic elements are abstracted to separate different concerns so that the model is robust against future changes and serve to provide common properties in a consistent manner to the elements that specialise them. There are also some concrete components such as TRANSPORT MODE that are used throughout the different parts of NeTEx. This section of the EPIP describes such elements.

NOTE To make the EPIP specification more readable, certain further ancillary NeTEx elements, such as **BookingArrangements** and **Notice**, which are in fact part of the general NeTEx framework, are presented later under an appropriate element in a specific frame.

5.5.2 TypeOfValue (Abstract)

TYPE OF VALUE describes a simple coded value with a name, associated url and other descriptive properties; it can be specialised to make arbitrary sets of codes used to classify other NeTEx entities, for example TYPE OF POINT, TYPE OF LINK, TYPE OF SERVICE, etc. TYPEs OF VALUE can be used to exchange application metadata that is used to validate or collect data, such as the descriptions and allowed values for codes.

TypeOfValue is abstract: it is specialised to create concrete types for specific purposes, such as **Branding**, **DataSource**, etc.

In the EPIP, in the interests of standardisation, fixed enumerated values are generally preferred, and the values encoded in schema suffice for most purpose.

Some limited use is made of **TypeOfValue**, however, in particular **TypeOfPlace** instances are used to define values with which to classify a **StopPlace** as “general” or “monomodal”. The EPIP values can be considered as “profile metadata” inherent to the EPIP and they can be referenced with external

references without being explicitly encoded in an EPIP conformant document. However, an explicit encoding is allowed (and enables the checking of references by validators), provided that it conforms to the agreed definition; they should be placed in a 'EU_PI_METADATA' frame.).

An augmented or extended profile might however want to add its own *TypeOfValue* instances. A particular use is to hold legacy classifiers of stops, etc, so that data sets can be exchanged losslessly in a round trip workflow

Table 19 – TypeOfValue – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	TYPE OF VALUE inherits from DATA MANAGED OBJECT.
«PK»	<i>id</i>	TypeOfValueIdType	1:1	Identifier of TYPE OF VALUE.
	<i>Name</i>	MultilingualString	0:1	Name of TYPE OF VALUE.
	<i>Description</i>	MultilingualString	0:1	Description of TYPE OF VALUE.

5.5.3 ValueSet

Many different categories of TYPE OF VALUE can be encoded in a NeTex Document. A **ValueSet** can be used to group values that belong to a given category so that a consumer system can recognize them as a related set of alternative options. (Without the use of **ValueSet** instances, all the different instance of *TypeOfValue* would appear as an undifferentiated list and it would not necessarily be clear to which element they relate).

For example, EPIP specifies two types of STOP PLACE 'monomodal' and 'general': these could be formally encoded and exchanged along with the data that references them (so that any use of them is validated) as follows:

```
<ValueSet version="1.0" id="epip:Types_of_StopPlace" classOfValues="TypeOfPlace">
  <Name>Types of EPIP Stop Place</Name>
  <values>
    <TypeOfPlace version="1.0" id="epip:general">
      <Name>General stop place</Name>
    </TypeOfPlace>
    <TypeOfPlace version="1.0" id="epip:monomodal">
      <Name>Single mode stop place</Name>
    </TypeOfPlace>
  </values>
</ValueSet>
```

Table 20 – ValueSet – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	VALUE SET inherits from DATA MANAGED OBJECT.
«PK»	<i>id</i>	ValueSetIdType	1:1	Identifier of VALUE SET.
«atr»	<i>classOfValues</i>	xsd:NCName	0:1	Name of XML Class for VALUE SET contents.

Classification	Name	Type	Cardinality	Description
	Name	<i>MultilingualString</i>	0:1	Name of VALUE SET.
«cntd»	values	<i>(TypeOfValue)</i>	0:*	Values in VALUE SET

5.5.4 GroupOfEntities (Abstract)

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.

Several different specialisations of GROUP OF ENTITIES are used in the EPIP: **GroupOfStopPlaces**, **GroupOfServices**, etc (and **Place** is in effect a **GroupOfPoints**).

Table 21 – GroupOfEntities – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u><i>DataManagedObject</i></u>	::>	GROUP OF ENTITIES inherits from DATA MANAGED OBJECT.
«PK»	id	<i>GroupOfEntitiesIdType</i>	1:1	Identifier of GROUP OF ENTITIES.
	Name	<i>MultilingualString</i>	0:1	Name of GROUP OF ENTITIES.
	ShortName	<i>MultilingualString</i>	0:1	Short Name of GROUP OF ENTITIES.
	Description	<i>MultilingualString</i>	0:1	Description of GROUP OF ENTITIES.
«FK»	PurposeOfGroupingRef	<i>PurposeOfGroupingRef</i>	0:1	PURPOSE OF GROUPING of GROUP OF ENTITIES.
«AK»	PrivateCode	<i>PrivateCode</i>	0:1	Private code associated of GROUP OF ENTITIES.
«cntd»	(members)	<i>(VersionOfObjectRef)+ / (GroupMember)+</i>	1:1	Reference to ENTITY included in the group. Each concrete specialisation shall add its own set restricted to the correct type of member. References may be simple (e.g. a specialisation of VersionOfObjectRef , or, if they have additional properties be a specialisation of GroupMember such as GroupOfServicesMember .

5.5.5 Point (Abstract)

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.

Table 22 – Point – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<i>DataManagedObject</i>	::>	POINT inherits from DATA MANAGED OBJECT.
«PK»	id	<i>PointIdType</i>	1:1	Identifier of POINT.
	Name	<i>MultilingualString</i>	0:1	Name of POINT.
«cntd»	Location	<i>Location</i>	0:1 1:1	Location of POINT. Mandatory in the EPIP.
«»	PointNumber	<i>xsd:normalizedString</i>	0:1	Arbitrary alternative identifier for the POINT.
«cntd»	projections	<i>Projection</i>	0:*	Projections for POINT.

5.5.6 Location (Subcomponent of Point)

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

Table 23 – Location – XML Element

Classification	Name	Type	Cardinality	Description
«FK»	srsName	<i>LocatingSystemNameType</i>	0:1	GML id of Type of LOCATING SYSTEM used. To be filled using the GML format (e.g. urn:ogc:def:crs:EPSG::4326 for WGS84, see http://www.epsg.org and http://www.opengeospatial.org/ogcUrnPolicy) NOTE WGS84 Is highly recommended for the EPIP.
	Longitude	<i>LongitudeType</i>	1:1	Longitude of Location.
	Latitude	<i>LatitudeType</i>	1:1	Latitude of Location.
	Altitude	<i>AltitudeType</i>	0:1	Altitude of Location.
«cntd»	Coordinates	<i>gml:pos</i>	0:1	GML coordinates providing location in a specified Location system.
	Precision	<i>xsd:decimal</i>	0:1	Precision of coordinates.

5.5.7 Link (Abstract)

A LINK is an oriented spatial object of dimension 1 with view to the overall description of a network, describing a connection between two POINTs.

Different types of links are used to describe the different layers of a Transport Network.

Table 24 – Link – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	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»	LineString	<i>gml:LineString</i>	0:1	GML linestring (detailed geometry) of the LINK.
«FK»	(FromPointRef)	PointRef	1:1	Reference to point at which LINK starts.
«FK»	(ToPointRef)	PointRef	1:1	Reference to point at which LINK ends.

5.5.8 LinkSequence (Abstract)

A LINK SEQUENCE describes a path through a network, such as a bus route. It can consist of an ordered set of POINTS IN SEQUENCE (referencing POINTs of a particular type) or LINKs in SEQUENCE (referencing LINKs of a particular type).

LinkSequence is abstract: it is specialised to create specific types of path such as **Route**, **JourneyPattern**, **ServicePattern**, etc.

Table 25 – LinkSequence – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<u>DataManagedObject</u>	::>	LINK SEQUENCE inherits from DATA MANAGED OBJECT.
«PK»	id	LinkSequenceIdType	1:1	Identifier of LINK SEQUENCE.
	Name	MultilingualString	0:1	Name of LINK SEQUENCE.
	Distance	DistanceType	0:1	Distance of LINK SEQUENCE. Units are as specified for Frame. Default is SI Metres.
«cntd»	(pointsInSequence)	<u>(XxxxPointInSequence)+</u>	1:*	An ordered sequence of POINT IN SEQUENCE elements that each reference a POINT of the specific type for the LINK SEQUENCE. For example, a ServicePattern holds a sequence of StopPointInJourneyPattern instances that each reference a ScheduledStopPoint ,

5.5.9 PointInLinkSequence (Abstract)

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

Table 26 – PointInLinkSequence – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<i>VersionedChild</i>	::>	POINT in LINK SEQUENCE inherits from VERSIONED CHILD.
«atr»	order	<i>xsd:positiveInteger</i>	1:1	Order of POINT in LINK SEQUENCE within sequence.
«FK»	LinkSequenceRef	<i>LinkSequenceRef</i>	1:1	Reference to LINK SEQUENCE containing POINT in LINK SEQUENCE.
	Description	<i>MultilingualString</i>	0:1	Description of POINT in LINK SEQUENCE. +v1.1

5.5.10 Zone (Abstract)

A ZONE defines a two-dimensional PLACE within the service area of a public transport OPERATOR (ADMINISTRATIVE ZONE, TARIFF ZONE, ACCESS ZONE, etc.).

Table 27 – Zone – XML Element (Abstract)

Classification	Name	Type	Cardinality	Description
::>	::>	<i>GroupOfPoints</i>	::>	ZONE inherits from GROUP OF POINTs.
«PK»	id	<i>ZoneldType</i>	1:1	Identifier of ZONE.
«cntd»	members	<i>PointRef</i>	0:*	List of points within ZONE.
«cntd»	Centroid	<i>Point</i>	0:1	Centre POINT of ZONE.
«cntd»	<i>gml:Polygon</i>	<i>gml:Polygon</i>		Polygon associated with zone.

5.5.11 Transport Mode and Submode

A MODE defines the means of transport used or available. NeTeX specialises MODE into TRANSPORT MODE, used to classify public transport systems for instance: bus, tramway, light rail, metro, long-distance rail, ferry.

A VEHICLE TYPE shall belong to a single TRANSPORT MODE. For instance, the 'bus' TRANSPORT MODE will include any sort of bus - standard, articulated, minibus, double-decker, etc.

A LINE is in general operated by only one TRANSPORT MODE: metro line, ferry line, etc.

EPIP limits the use of MODEs to the fixed values provided as an enumeration (*VehicleMode*).

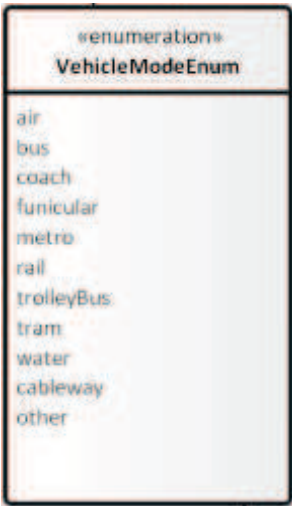


Figure 3 — Transport modes

The SUBMODE allows the TRANSPORT MODE to be further qualified with a subclassification. The subclassifications typically distinguish different contexts of operation for the mode, for example, for the ‘bus’ mode, some submodes are ‘schoolBus’, ‘nightBus’, ‘expressBus’, etc.).

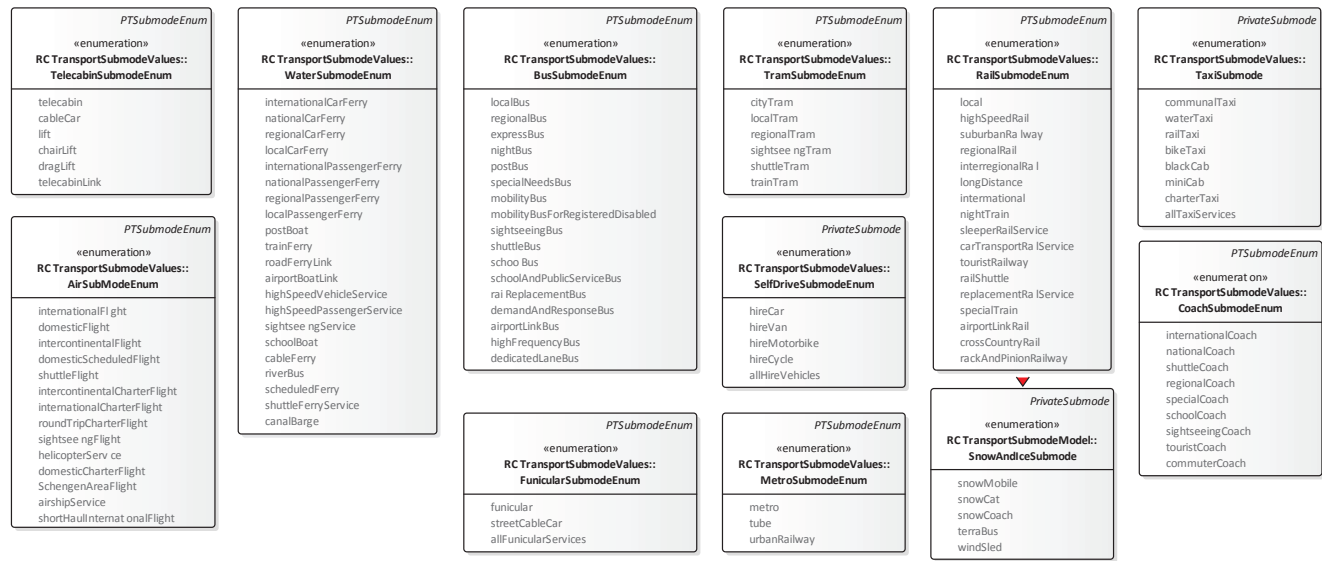


Figure 4 — Transport submodes

5.5.12 Assignment (Abstract)

An ASSIGNMENT specifies a set of properties to be associated with another ENTITY on a temporary or permanent basis. An example is a STOP ASSIGNMENT, used to relate a stop in the time table with a physical stop. It may also allocate a platform (QUAY) for a journey for some or all journeys.