INTERNATIONAL STANDARD

ISO 21219-2

First edition
2019-07

Intelligent transport systems — Traffic and travel information (TTI) via transport protocol experts group, generation 2 (TPEG2) —

Part 2: UML modelling rules (TPEG2-UMR)

Systèmes intelligents de transport — Informations sur le trafic et le tourisme via le groupe expert du protocole de transport, génération 2 (TPEG2) —

Partie 2: Règles de modelage UML

Reference number
ISO 21219-2:2019(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

This first edition cancels and replaces ISO/TS 21219-2:2014 which has been technically revised. The main changes compared to the previous edition are as follows:

A list of all parts in the ISO 21219 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

History

TPEG technology was originally proposed by the European Broadcasting Union (EBU) Broadcast Management Committee, who established the B/TPEG project group in the autumn of 1997 with a brief to develop, as soon as possible, a new protocol for broadcasting traffic and travel-related information in the multimedia environment. TPEG technology, its applications and service features were designed to enable travel-related messages to be coded, decoded, filtered and understood by humans (visually and/or audibly in the user's language) and by agent systems. Originally, a byte-oriented data stream format, which may be carried on almost any digital bearer with an appropriate adaptation layer, was developed. Hierarchically structured TPEG messages from service providers to end-users were designed to transfer information from the service provider database to an end-user's equipment.

One year later, in December 1998, the B/TPEG group produced its first EBU specifications. Two documents were released. Part 2 (TPEG-SSF, which became ISO/TS 18234-2) described the syntax, semantics and framing structure, which was used for all TPEG applications. Meanwhile, Part 4 (TPEG-RTM, which became ISO/TS 18234-4) described the first application for road traffic messages.

Subsequently, in March 1999, CEN/TC 278, in conjunction with ISO/TC 204, established a group comprising members of the former EBU B/TPEG and this working group continued development work. Further parts were developed to make the initial set of four parts, enabling the implementation of a consistent service. Part 3 (TPEG-SNI, ISO/TS 18234-3) described the service and network information application used by all service implementations to ensure appropriate referencing from one service source to another.

Part 1 (TPEG-INV, ISO/TS 18234-1) completed the series by describing the other parts and their relationship; it also contained the application IDs used within the other parts. Additionally, Part 5, the public transport information application (TPEG-PTI, ISO/TS 18234-5), was developed. The so-called TPEG-LOC location referencing method, which enabled both map-based TPEG-decoders and non-map-based ones to deliver either map-based location referencing or human readable text information, was issued as ISO/TS 18234-6 to be used in association with the other applications of parts of the ISO/TS 18234 series to provide location referencing.

The ISO/TS 18234 series has become known as TPEG Generation 1.

TPEG Generation 2

When the Traveller Information Services Association (TISA), derived from former forums, was inaugurated in December 2007, TPEG development was taken over by TISA and continued in the TPEG applications working group.

It was about this time that the (then) new Unified Modelling Language (UML) was seen as having major advantages for the development of new TPEG applications in communities who would not necessarily have binary physical format skills required to extend the original TPEG TS work. It was also realized that the XML format for TPEG described within the ISO/TS 24530 series (now superseded) had a greater significance than previously foreseen, especially in the content-generation segment and that keeping two physical formats in synchronism, in different standards series, would be rather difficult.

As a result, TISA set about the development of a new TPEG structure that would be UML-based. This has subsequently become known as TPEG Generation 2.

TPEG2 is embodied in the ISO/TS 21219 series and it comprises many parts that cover introduction, rules, toolkit and application components. TPEG2 is built around UML modelling and has a core of rules that contain the modelling strategy covered in ISO 21219-2, ISO 21219-3 and ISO 21219-4 and the conversion to two current physical formats: binary and XML; others could be added in the future. TISA uses an automated tool to convert from the agreed UML model XMI file directly into an MS Word document file, to minimize drafting errors, that forms the annex for each physical format.

TPEG2 has a three-container conceptual structure: message management (ISO 21219-6), application (several parts) and location referencing (ISO/TS 21219-7). This structure has flexible capability and can accommodate many differing use cases that have been proposed within the TTI sector and wider for hierarchical message content.

TPEG2 also has many location referencing options as required by the service provider community, any of which may be delivered by vectoring data included in the location referencing container.

The following classification provides a helpful grouping of the different TPEG2 parts according to their intended purpose. Note that the list below may be incomplete, e.g. new TPEG2 parts may be introduced after publication of this document.

- Toolkit parts: TPEG2-INV (ISO/TS 21219-1), TPEG2-UML (ISO 21219-2), TPEG2-UBCR (ISO 21219-3), TPEG2-UXCR (ISO 21219-4), TPEG2-SFW (ISO 21219-5), TPEG2-MMC (ISO 21219-6), TPEG2-LRC (ISO/TS 21219-7).
- Special applications: TPEG2-SNI (ISO/TS 21219-9), TPEG2-CAI (ISO/TS 21219-10), TPEG2-LTE (ISO/TS 21219-24).
- Location referencing: TPEG2-OLR (ISO/TS 21219-22), TPEG2-GLR (ISO/TS 21219-21), TPEG2-TLR (ISO 17572-2), TPEG2-DLR (ISO 17572-3).
- Applications: TPEG2-PKI (ISO/TS 21219-14), TPEG2-TEC (ISO/TS 21219-15), TPEG2-FPI (ISO/TS 21219-16), TPEG2-TFP (ISO 21219-18), TPEG2-WEA (ISO/TS 21219-19), TPEG2-RMR (ISO/TS 21219-23), TPEG2-EMI (ISO/TS 21219-25), TPEG2-VLI (ISO/TS 21219-26).

TPEG2 has been developed to be broadly (but not totally) backward compatible with TPEG1 to assist in transitions from earlier implementations, while not hindering the TPEG2 innovative approach and being able to support many new features, such as dealing with applications having both long-term, unchanging content and highly dynamic content, such as parking information.

This document is based on the TISA specification technical/editorial version reference: SP10031.

Intelligent transport systems — Traffic and travel information (TTI) via transport protocol experts group, generation 2 (TPEG2) —

Part 2: UML modelling rules (TPEG2-UMR)

1 Scope

This document specifies rules for the creation and extending of TPEG application UML models. The rules are intended to ensure that TPEG application UML models can be interpreted unambiguously for conversion to physical format representations. TPEG application UML models that are defined according to these rules can be used for automatic generation of TPEG standards and for automatic generation of TPEG application physical format descriptions.

This document also specifies the preferred structure of TPEG application specifications.

The TPEG abstract data types and the set of TPEG tables of common use are specified in the annexes.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions 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
abstract data type
data type of atomic nature

3.2
attribute compartment
graphical section of a UML class box positioned directly under the *class name compartment* (3.3)

3.3
class name compartment
top most graphical section of a class box defining the name of the class and optionally a stereotype, inherited class and package scope

3.4
data structure
data type being composed of other data types being either of *abstract* (3.1) or complex data type, not having a *component header* (3.6), stereotyped as <<DataStructure>>

3.5
component
revisable, named, complex data type, not stereotyped as <<DataStructure>>

3.6
component header
data structure (3.4) consisting of a component identifier, component length indicator and attribute length indicator

3.7
element
component (3.5) or *data structure* (3.4)

3.8
link
relation between two or more *elements* (3.7)

3.9
TPEG Application
set of classes and rules defining TPEG information services at the highest layer of the ISO OSI model

4 Abbreviated terms

IPR	Intellectual Property Right(s)
ISO	International Organization for Standardization
LRC	Location Referencing Container
MMC	Message Management Container
OSI	Open Systems Interconnection
PTI	PTI Public Transport Information
TISA	Traveller Information Services Association
TPEG	Transport Protocol Experts Group
UML	Unified Modelling Language

5 TPEG UML model definition

5.1 Allowed UML elements

TPEG UML models are based on the UML standard[1], but only use a subset of the elements defined in the standard. This clause provides a description of the elements of UML that are used for modelling TPEG. This clause also defines restrictions on these elements. TPEG UML models shall only use the UML elements described in this clause. The defined restrictions shall be obeyed.

5.1.1 Class

A class provides a description of the structure of the data stored in an instance of a class. The data are stored in the class attributes.

Classes shall be named according to the following regular expression: “^[a-zA-Z]\w*$” which is equivalent to “class names shall start with an English alphabet character, followed by zero or more English alphabet characters, digits or underscores”. As an exception, classes stereotyped as enumerations shall be named according to the following regular expression: “^[a-z]{3}\d{3}:[azA-Z]\w*$” which is equivalent to “enumeration class names shall start with three lower-case English alphabet characters, followed by three digits, a colon, a English alphabet character, then zero or more English alphabet characters, digits or underscores”. Upper camel case should be used for class names, with the first letter of every word capitalized. Additionally, class names shall be unique within their TPEG application or TPEG toolkit scope.

5.1.2 Abstract class

Abstract classes may be used to define shared properties of specialized child classes.

5.1.3 Attribute

An attribute provides a data type description of data that is stored in a class. Attributes can be either of primitive data type or compound data type.

An attribute shall be named according to the following regular expression: “^[a-zA-Z]\w*$” which is equivalent to “attribute names shall start with an English alphabet character, followed by zero or more English alphabet characters, digits or underscores“. As an exception, attributes within classes stereotyped as enumerations shall be named as defined in 5.2.3.5. Lower camel case should be used for attribute names, with the first letter in lowercase and the first letters of subsequent words in uppercase. Additionally, attributes shall be named uniquely within their class.

Within a class, an attribute has a multiplicity. If not explicitly indicated, the multiplicity is one. Other multiplicities may be indicated between square brackets: [minOccurs .. maxOccurs].

minOccurs and maxOccurs shall be positive integers, where maxOccurs shall be greater or equal to minOccurs. Alternatively, maxOccurs may be theoretically unbounded, in which case an asterisk “*” is modelled. However, in physical formats there may be upper bounds due to length restrictions and size indicator data type limitations.

Attribute multiplicity shall be interpreted as listed in Table 1. If no multiplicity is indicated, a multiplicity of *one* (mandatory attribute) is implied.

Table 1 — TPEG multiplicity

Multiplicity	TPEG meaning
minOccurs=1 and maxOccurs=1	mandatory attribute
minOccurs=1 and maxOccurs>1	mandatory list of attributes
minOccurs=0 and maxOccurs=1	optional attribute
minOccurs=0 and maxOccurs>1	optional list of attributes
minOccurs>1	mandatory list of attributes with a minimum occurrence

Attributes in classes are always modelled as public. Each attribute must have a data type. Attributes occur in the order as listed in the class definition in TPEG physical formats, unless this is overruled by the stereotype <<UnorderedComponentGroup>> .

NOTE Optional list of attributes with a minimum occurrence can be modelled as optional attribute including a data structure including a mandatory list of attributes with minOccurs>1.

5.1.4 Dependency

Figure 1 is the graphical representation used for ordered components (attributes stereotyped as <<OrderedComponentGroup>>) and DataStructures to show the hierarchical structure of the UML model.

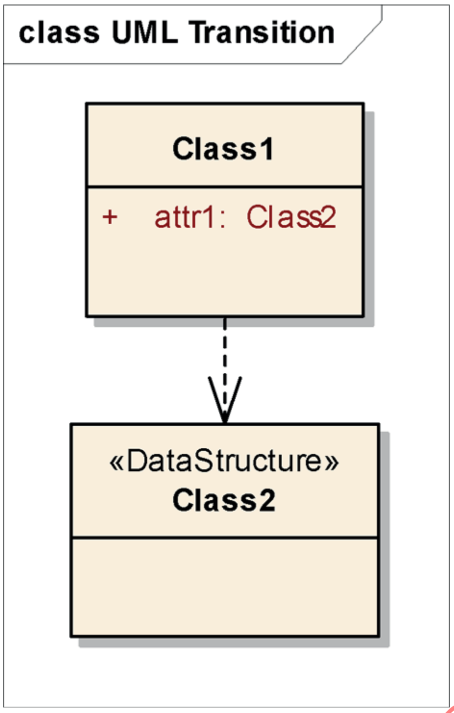

Figure 1 — UML dependency relation

NOTE Aggregation and composition associations are deprecated in TPEG models; the preferred method is inclusion as an attribute.

5.1.5 Specialization

A specialization (see Figure 2) relates a parent class to a child class. The child class inherits properties from the parent class. Classes shall not inherit from multiple parent classes. Classes shall only inherit from classes with the same stereotype.

Derived classes copy all attributes from the parent class. Parent classes shall contain classes not stereotyped as <<DataStructure>> or <<BinaryDataContainer>>. Parent classes shall be modelled as abstract class. In future versions of a standard, parent classes shall not be extended. Classes shall not be both parent and child class.

NOTE Extending parent classes in future versions of a standard breaks backwards compatibility.

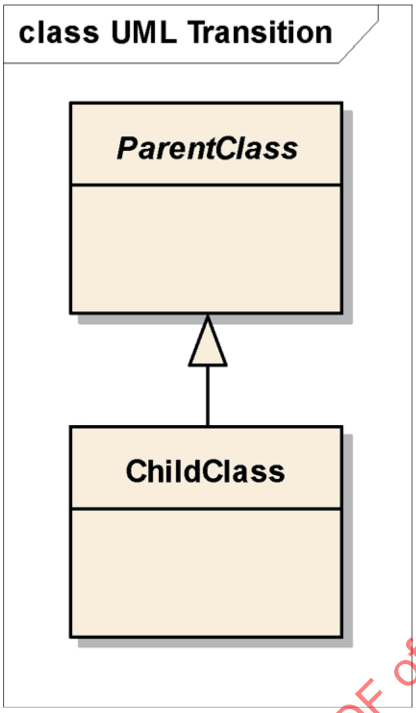

Figure 2 — UML specialization relation

5.1.6 Stereotype

A stereotype is used to provide an additional classification of UML properties. A physical format specification may use stereotype information to select a rule set for converting UML to the physical format.

The stereotypes as listed in Table 2 may be used for UML modelling of TPEG applications. Other stereotypes shall not be used.

Table 2 — TPEG stereotypes

UML element	Stereotype	TPEG meaning
Package	TPEG Application	Self standing protocol specification for a given application.
Package	TPEG Toolkit	Specification of general interest being referenced by different other specifications.
Package	TPEG DataTypes	Specification defining data structures and tables belonging to one single package.
Class	DataStructure	TPEG data structure.
Class	Enumeration	List of defined, constant expressions not containing attributes or sub data elements.
Class	External	TPEG Component defined in an external document.
Class	BinaryDataContainer	Component carrying binary data in its body that is not further specified within TPEG.
Attribute	OrderedComponentGroup	Attribute is of component type, and belongs to the group of components occurring in the order as defined by the attribute order.
Attribute	UnorderedComponentGroup	Attribute is of component type, and belongs to the group of components that may occur in random order, after all other attributes.

5.1.7 Tagged values

Tagged values may be used to provide additional information on a UML element, used for the creation of the specification document.

Only the tagged values listed in Table 3 shall be used.

Table 3 — Allowed tagged values

Tag	UML Element	TPEG meaning	Example
ApplicationAbbreviation	Package	Abbreviation of the application name as string.	TEC
ApplicationName	Package	Name of the application as string.	Traffic Event Compact
ApplicationNo	Package	Part number to be included in the target specification title.	A value of „2“ will result in: Intelligent transport systems — Traffic and Travel Information (TTI) via Transport Protocol Expert Group, Generation 2 (TPEG2) — Part 2: UML Modeling Rules (TPEG2-UMR_1.1/001)
ApplicationRoot	Package	Name of the root class of an application as string.	TECMessage
ApplicationVersion	Package	Version of the specification that is generated from this model. Major. Minor VersionNumber.	3.1
ApplicationRelease	Package	Integer that is increased each time the model is changed	Integer/345
TableEntryExample	Enumeration attribute	Comment for a table entry	
Documentation		Description of generic properties of a class	
Description		Description of single attributes within a class	
GCId	Class	Positive integer value in the range of [0..255] that uniquely identifies a class in the binary physical format.	1
ModelingRulesVersion	Package	Technical version number (MajorMinorVersion) of the UML Modelling Rules specification to which this model complies.	1.2
tpegMLprefix	Class	For classes stereotypes as <<External>> an namespace prefix may be given here.	Tec
tpegMLuri	Class	For classes stereotypes as <<External>> an namespace URI may be given here.	http://www.tisa.org/tpeg/...

In UML packages that are stereotyped as <<TPEG Application>> , the ApplicationAbbreviation, ApplicationName, ApplicationRoot ApplicationNo and ModelingRulesVersion tagged values are mandatory.

5.1.8 Notes

Notes may be used to provide additional information that is used for generating the specification document.

5.2 Modelling rules and recommendations

5.2.1 General

TPEG UML models are used to generate TPEG specifications. A fundamental assumption is that applications will develop and new features will be added. Correct designs permit applications to be upgraded and extended over time, providing new features to new decoders, and yet permit existing decoders to continue to operate. This clause describes design principles that shall be obeyed when building and upgrading TPEG applications.

5.2.2 Order of elements

In a physical format, attributes shall occur in the same order as listed in the UML class definition (see Figure 3).

When components may occur in any order (independent of the order in which they are listed in the UML class definition), they should be modelled as attributes with the stereotype <<UnorderedComponentGroup>> and of the type of the corresponding class. The unordered components shall be linked by the embedding class using a dependency relation.

When components shall occur in a specific order, they shall be modelled as attributes with the stereotype <<OrderedComponentGroup>> and of the type of the corresponding class. The ordered components shall be linked by the embedding class using a dependency relation.

Mandatory attributes should occur *before* optional attributes. Mandatory Booleans should occur *after* the other mandatory attributes. Optional attributes should occur *after* mandatory attributes. Components shall occur *after* all other attributes. Ordered components shall occur *before* unordered components.

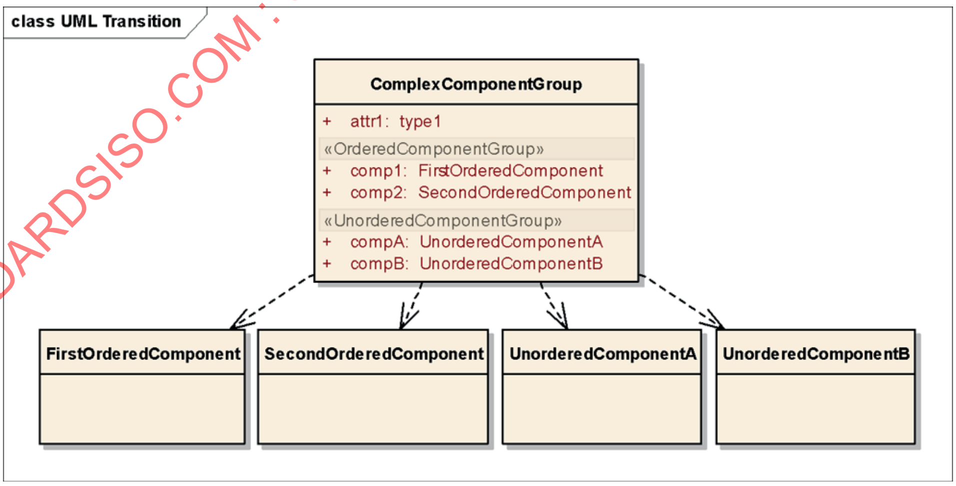

Figure 3 — Ordering of class elements

NOTE Special rules for extending TPEG UML models for newer revisions of a standard are provided in 5.3. Extending TPEG UML models in a backwards compatible way can break the recommendations for ordering mandatory and optional elements described in 5.2.2.

5.2.3 Stereotypes

5.2.3.1 TPEG Application

The <<TPEG Application>> stereotype is used to identify a UML package as TPEG Application.

NOTE UML packages that are stereotyped as <<TPEG Application>> can as well be stereotyped as <<TPEG Toolkit>> if the use of classes is encouraged in other applications.

5.2.3.2 TPEG Toolkit

TPEG Toolkits are used to share common functionality between different TPEG Applications. For example the Location Referencing Container and Message Management Container are toolkits that are used by all TPEG applications. A TPEG Application therefore can refer to a data type definition not specified in the same model.

TPEG Toolkits are designed such that its root components are defined as templates which can be used as external reference within other packages. A TPEG Application using a toolkit template therefore needs to specify a unique interface class for this instantiation of the imported toolkit interface's component.

All subsequent components in a toolkit are defined as out of the scope of the TPEG application, i.e. the toolkit on its own defines subcomponents beginning with local identifiers.

The <<TPEG Toolkit>> stereotype is applied at UML package level.

5.2.3.3 TPEG DataTypes

General TPEG datatypes and TPEG Application specific datatypes as well as classes stereotyped as <<enumeration>> are defined in separate UML packages. This only applies for elementary data types and classes that are stereotyped as <<DataStructure>>.

The <<TPEG DataTypes>> stereotype is applied at UML package level.

5.2.3.4 DataStructure

The TPEG binary format distinguishes between *components* and *datastructures*. In this physical representation, a component is a compound data type, containing a header, providing type and length information. The type information of a component shall be unique within an application. A datastructure is a compound datatype not containing this type and length information.

Datastructures shall explicitly be stereotyped as <<DataStructure>> . Classes that are not explicitly stereotyped as <<DataStructure>> shall be interpreted as component. The differentiation between components and datastructures is not relevant for tpegML as both variants are represented as xs: element.

Components shall be included in datastructures by using attributes typed as this component.

The <<DataStructure>> stereotype is applied at UML class level.

The specific usage of datastructures or components depends on the requirements of the particular application. Components should be used wherever future extensions are envisioned, and where 'future proofing' is a strong requirement. Datastructures are more bandwidth efficient as they contain no header information but are not extendible in a backwards-compatible manner.

5.2.3.5 Enumeration

TPEG Tables are modelled as enumerations and shall have no more than 256 entries. Each enumeration shall have a unique name which consists of two parts. The first part is the abbreviation of the application where the enumeration is specified in, appended with a three-digit decimal number, starting at value

001. The second part is a description of the content of the table in camel-case. The two parts are separated by a colon.

EXAMPLE typ001:LanguageCode.

When applicable, the enumeration entry with index zero should have the value “unknown”. If tables are likely to be extended in future versions of a standard, a default value shall be defined. This default value shall have index 255.

Additionally, enumeration entry names shall be named according to the following regular expression: “^(25[0- 5]|2[0-4][0-9]|[01]?[0-9][0-9]?):.+$”, which is equivalent to “enumeration entry names shall start with a numerical value between 0 and 255, followed by a colon and one or more arbitrary characters”, where the text following the colon should only contain printable characters and whitespaces.

EXAMPLE See Figure 4.

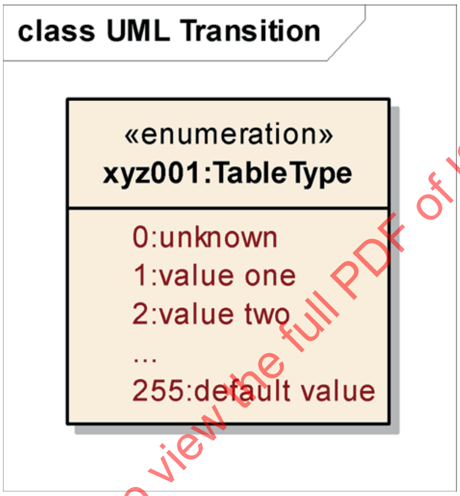

Figure 4 — Enumeration stereotype

The <<Enumeration>> stereotype is applied at UML class level.

5.2.3.6 External

TPEG applications can use classes that are not defined in the same package, but are defined elsewhere. These classes shall be stereotyped as <<External>> .

The <<External>> stereotype is applied at UML class level.

Classes stereotyped as <<External>> shall not contain or inherit any attributes, but define the container formats only, while the content is defined in the referenced class/specification.

There are three different levels of external references:

1) Classes defined in other TPEG applications or TPEG toolkits.

   In this case, the class shall be derived from the class in the other package. Additionally, the tpegMLprefix and tpegMLuri shall be used.

   In the case where a class stereotyped as <<External>> has already been derived from another class, this second external generalisation is valid. Classes imported via this mechanism shall not be abstract.

2) Classes not defined according to this document, but an XSD is available.

   In this case, the class shall not be derived from a class in another package, but the tpegMLprefix and tpegMLuri shall be specified as tagged values for this class.

3) Classes not defined according to this document with only proprietary binary data being available.

   In this case, the class shall not be derived from a class in another package and the tpegMLprefix and tpegMLuri shall not be used.

5.2.3.7 UnorderedComponentGroup

By default, *components* that are included in classes should be modelled as being unordered: The order in which the components appear in a model has no influence on the order in a physical format representation. These components shall be modelled as attributes with the stereotype <<UnorderedComponentGroup>> .

The <<UnorderedComponentGroup>> stereotype is applied at UML attribute level.

5.2.3.8 OrderedComponentGroup

By default, components that are included in classes are unordered (see 5.2.3.7). When *components* should appear in a fixed order, they shall be modelled as attributes with the stereotype <<OrderedComponentGroup>> and of the type of the corresponding class. See 5.2.2 for details on ordering of attributes.

The <<OrderedComponentGroup>> stereotype is applied at UML attribute level.

5.2.3.9 BinaryDataContainer

TPEG applications can use classes that carry binary data which is not defined in TPEG, but is defined elsewhere.

These classes shall be stereotyped as <<BinaryDataContainer>>.

The <<BinaryDataContainer>> stereotype is applied at UML class level.

Classes stereotyped as <<BinaryDataContainer>> shall not contain or inherit any attributes, but define the container formats only, while the content is defined elsewhere. Such classes shall have no child classes and no classes shall be derived from them.

5.2.4 Data types

An overview of TPEG abstract data types is given in Annex A.

5.2.5 Optional Booleans

Due to encoding in the binary physical format, usage of optional Booleans is deprecated. The preferred solution is to use two mandatory Booleans, with the first one indicating the validity of the second Boolean. Alternatively, the enumeration as defined in typ008:OptionalBoolean (see Annex B) may be used as attribute.

5.2.6 Tables and Switching Tables

TPEG Tables represent defined groupings of pre-defined concepts (see 5.2.3.5 for details on the modelling of tables and Annex B for an overview of the general TPEG tables). TPEG enables hierarchical tables (see example below) by using ‘Switching Tables’. Switching Tables are modelled in UML using abstract classes. The switched table is referenced by including an attribute of the abstract table class type (see Figure 5).

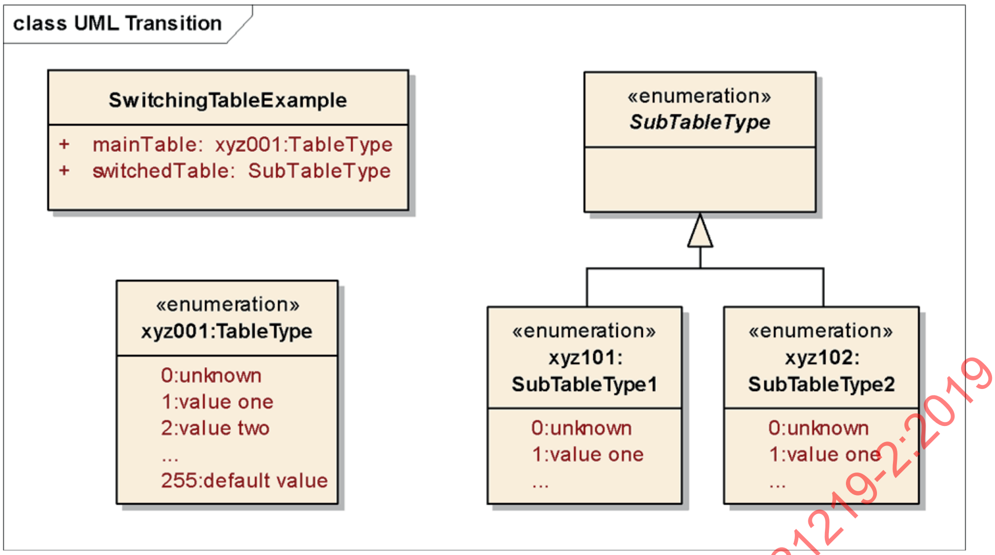

Figure 5 — TPEG Switching Table concept

The Application Specification shall define an unambiguous relation between the mainTable entry and the switchedTable type. It is recommended that the name of the switchedTable matches the entry in the mainTable and that the switchedTable number has a direct relation with the mainTable number.

EXAMPLE 'One table called 'xyz001:furniture' has one entry '1:chair'. One hierarchical level below the 'xyz001:furniture' table might be a table called 'xyz101:chair', having entries '1:computer chair', '2:rocking chair', etcetera. If an attribute of type xyz001:furniture has value 1, the referenced table is of type xyz101:chair.

5.2.7 Inclusion of DataStructures

Classes that are stereotyped as <<DataStructure>> shall be included in a class by an attribute of type of this DataStructure, which shall not be stereotyped <<UnorderedComponentGroup>> or <<OrderedComponentGroup>>.

5.2.8 Inclusion of Components in DataStructures

When a class stereotyped as <<DataStructure>> shall include a component, this class shall be modelled as an attribute stereotyped as <<OrderedComponentGroup>> .

5.2.9 Linking abstract classes

Classes can link to abstract classes. In this case, an instance of exactly one of the child classes is instantiated. If one of the child classes is stereotyped as <<External>> the abstract class itself shall not contain any attributes.

EXAMPLE The LinkingClass has one attribute of abstract type, which can be instantiated with ChildClass1 and ChildClass2 (see Figure 6). The class only links to the abstract class. Each linked instance is of one of the two child class types.

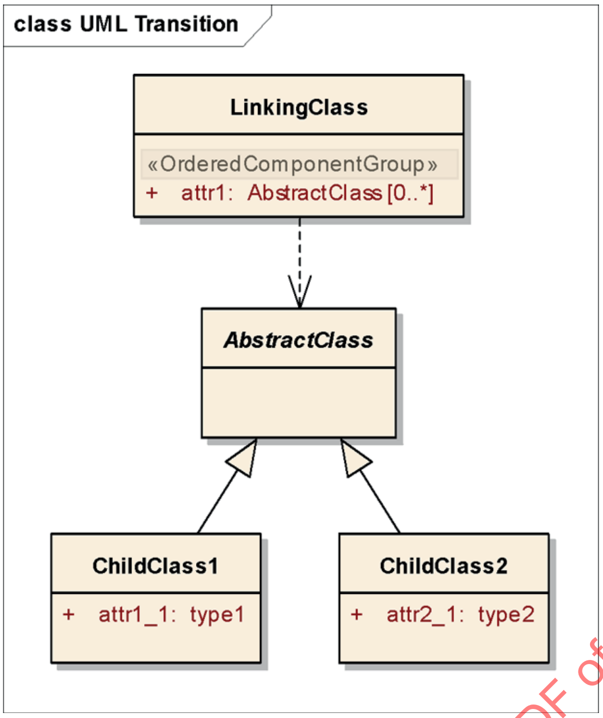

Figure 6 — Linking of abstract class

NOTE Linking of classes is often referred to as switching classes.

5.2.10 Graphical representation

Linked classes should be represented below the linking class. This provides a graphical hierarchical representation, improving readability.

5.2.11 Documentation

Each class shall contain a functional description of the class and its attributes.

NOTE More information on documenting TPEG UML models is given in 5.4.

5.3 Extending TPEG UML models

TPEG provides means for extending applications without breaking backward compatibility. Classes can be extended by appending components, mandatory and optional attributes.

For each revision of a standard, the following rules shall be obeyed:

Classes that are stereotyped as <<DataStructure>> or <<BinaryDataContainer>> shall not be extended.

Mandatory attributes shall be appended *after* all previously defined attributes and *before* all components that are modelled as attributes with the stereotype <<OrderedComponentGroup>> . Optional attributes shall be appended *after* all previously defined attributes and newly added mandatory attributes and *before* all components that are modelled as attributes with the stereotype <<OrderedComponentGroup>> .

The order and the multiplicity of the attributes shall not be changed.

Newly added ordered components shall occur only *after* the last ordered component of the extended class. No additional rules for adding components not stereotyped as <<OrderedComponentGroup>> apply.

Newly added unordered components shall only occur *after* the last unordered component of the extended class. No additional rules for adding components not stereotyped as <<UnorderedComponentGroup>> apply.

Attributes shall not be added to parent classes (classes that other classes inherit from), this would cause breaking backward compatibility.

Other application extensions break backward compatibility and are therefore strongly depreciated.

EXAMPLE Figure 7 shows a complex component in a first and second revision of a standard. The second revision contains a newly added mandatory attribute, optional attribute and ordered component.

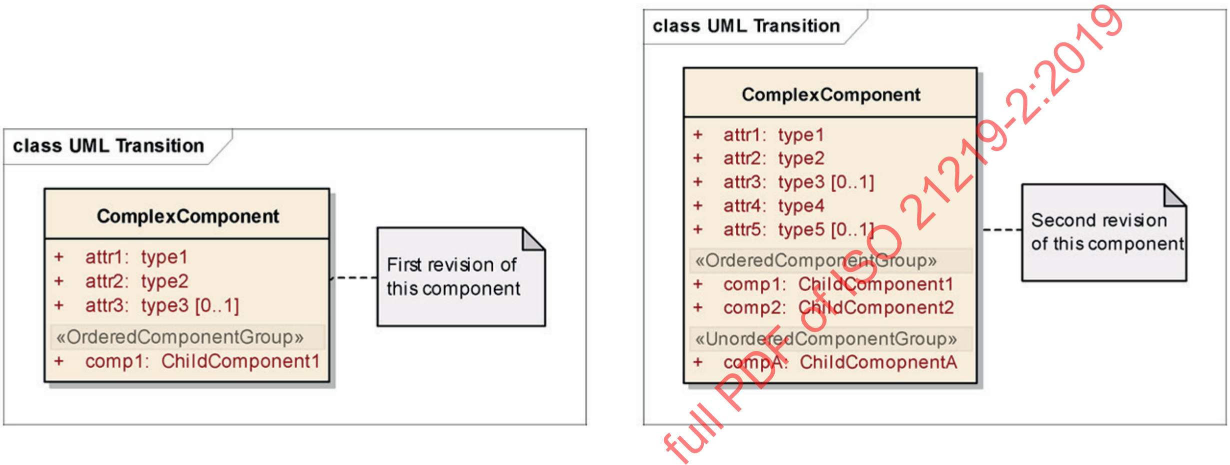

Figure 7 — Extending a TPEG component in a backwards compatible way

5.4 Adding documentation to TPEG UML models

Several parts from TPEG specifications are automatically generated from TPEG UML models. This particularly concerns class and attribute descriptions. All information required for generating these TPEG specification sections is contained in the UML models.

The documentation is considered to be floating plain text structured in paragraphs. Optionally, markup may be used if required.

NOTE The use of markup depends on editors and tools. The markup can be stripped or ignored or misinterpreted dependent on the tools choice.

5.4.1 Class documentation

Each class shall have a definition text or functional description attached via the UML editor's class property documentation feature. This description may augment with additional text outside the class properties, e.g. in a general clause explaining the overall TPEG application or toolkit structure before the UML-generated clauses.

5.4.2 Attribute description

Each class attribute shall have a definition text or functional description attached via the UML editor's attribute documentation feature.

6 Drafting specifications using UML models

6.1 General

This clause specifies rules for creating TPEG specification documents using TPEG compliant UML models.

6.2 Specification of contents

6.2.1 Foreword

See the Foreword of this document. The text of the Foreword may be adapted by ISO so may change compared to the Foreword of this document.

6.2.2 Introduction

See the Introduction of this document. The text of the Introduction may be adapted by ISO so may change compared to the Introduction of this document.

6.2.3 Scope

Rules for the drafting of the scope of TPEG specifications are given in 6.2.1 of [2].

6.2.4 Bibliography

Rules for the drafting of the bibliography of TPEG specifications are given in 6.4.2 of [2].

6.3 Normative clauses

Each TPEG application specification should include the following normative clauses, preferably in the order as given in this subclause.

6.3.1 Application specific constraints

A TPEG application may pose constraints on elements of TPEG specifications the application specification refers to, such as the TPEG Toolkit. If these constraints are purely application specific, they shall be stated in this clause. If these constraints are of more general nature, definition of a separate TPEG profile should be considered.

6.3.2 Class model

This clause contains the UML model of the application. The UML model shall be drafted according to the rules specified in this document.

6.3.3 Components

Each class in the class model that is not stereotyped as <<DataStructure>> shall be described in this clause. The description shall at least consist of a functional description of the component and a tabular description of the component attributes. This tabular attribute description shall have four columns: *name*, *type*, *multiplicity* and *description*. The description of attributes of primitive data types shall specify its function and, if required, its allowed value range. The description of attributes of compound data type (components or datastructures) shall refer to the clause in which the respective attribute is defined.

6.3.4 Datatypes

Each class in the class model that is stereotyped as <<DataStructure>> shall be described in this clause. The description shall at least consist of a functional description of the datastructure and a tabular description of the datastructure attributes. This tabular description shall have four columns: *name*, *type*, *multiplicity* and *description*. The description of attributes of primitive data types shall specify its function and, if required, its allowed value range. The description of attributes of compound data type (components or datastructures) shall refer to the clause in which the respective attribute is defined.

6.3.5 Tables

When an application defines application-specific tables, they shall be enlisted in this clause. Each table shall be listed in a separate subclause, the subclause having the name of the respective table. Switched tables (tables that have an abstract parent class, see 5.2.6) shall be grouped in a subclause that has the name of the abstract parent class and shall be listed in subclauses of the abstract parent class.

6.4 Specification of normative annexes

The normative annexes of TPEG specifications shall at least contain the physical format descriptions of the respective standard. Rules for converting TPEG UML models to the TPEG binary format description can be found in [7]. Rules for converting TPEG UML models to the tpegML format description can be found in [8].

Annex A
(normative)

TPEG abstract data types

A.1 Data type definition

The abstract data types enlisted in Table A.1 are available for modelling TPEG applications. Compound data types may be specified in toolkits or within an application.

Table A.1 — TPEG abstract data types

Data type	Definition
BitArray	List of Boolean values.
Boolean	Attribute having values [true, false].
DateTime	Attribute providing date and time information in UTC.
DaySelector	Attribute that allows selection of one or multiple of [Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday].
DistanceMetres	Distance in whole metres [0 .. 4 294 967 295].
DistanceCentiMetres	Distance in whole centimetres [0 .. 4 294 967 295].
Duration	Temporal duration in whole seconds [0 .. 4 294 967 295].
FixedPercentage	Percentage value in whole percents [0 .. 100].
Deprecated: FixedPointNumber	The datatype is deprecated and may be removed in one of the future versions of this document. Please do not make use of this datatype anymore.
Float	Number with decimal precision [$\pm 2^{-126}$ .. $\pm(1-2^{-24}) \times 2^{128}$].
IntSiTi	(Integer Signed Tiny) Signed integer value of fixed size [-128 .. 127].
IntSiLi	(Integer Signed Little) Signed integer value of fixed size [-32 768 .. 32 767].
IntSi24	(Integer Signed 24-bit) Signed integer value of fixed size [-8388608 .. 8388607].
IntSiLo	(Integer Signed Long) Signed integer value of fixed size [-2 147 483 648 .. 2 147 483 647].
IntSiLoMB	(Integer Signed Long MultiByte) Signed integer value of variable size [-2 147 483 648 .. 2 147 483 647].
IntUnTi	(Integer Unsigned Tiny) Unsigned integer value of fixed size [0 .. 255].
IntUnLi	(Integer Unsigned Little) Unsigned integer value of fixed size [0 .. 65 535].
IntUn24	(Integer Unsigned 24-bit) Unsigned integer value of fixed size [0 .. 16777215].
IntUnLo	(Integer Unsigned Long) Unsigned integer value of fixed size [0 .. 4 294 967 295].
IntUnLoMB	(Integer Unsigned Long MultiByte) Unsigned integer value of variable size [0 .. 4 294 967 295].
Character value limitation for String types: ShortString, LongString, LocalisedShortString, LocalisedLongString	According to TPEG2 SNI 3.2 In TPEG2 the only allowed character encoding is UTF-8. To allow seamless conversion between binary and XML formats only the following character values are valid for TPEG2 string types, (as they are allowed within XML documents): [#x1-#xD7FF] | [#xE000-#xFFFD] | [#x10000-#x10FFFF] /* any Unicode character, excluding the surrogate blocks, FFFE, and FFFF and 0. */

Table A.1 *(continued)*

Data type	Definition
ShortString	Sequence up to 255 characters (in single-byte character encoding).
LongString	Sequence of up to 65 535 characters (in single-byte character encoding).
LocalisedShortString	Sequence of up to 255 characters (in single-byte character encoding) accompanied with a language code (typ001:LanguageCode).
LocalisedLongString	Sequence of up to 65 535 characters (in single-byte character encoding) accompanied with a language code (typ001:LanguageCode).
Probability	Probability value with a precision of two decimals [0.00 .. 1.00].
ServiceIdentifier	Three part numeric value [SID_A.SID_B.SID_C] with SID_A, SID_B and SID_C each having a range of [0 .. 255].
TimeInterval	Temporal duration with indication of years [0 .. 130], months [0 .. 12], days [0 .. 31], hours [0 .. 24], minutes [0 .. 60] and seconds [0 .. 60]. At least one attribute must be defined. Undefined attributes imply the default value zero.
TimePoint	Point in time with indication of years [1970 .. 2100], months [0 .. 12], days [0 .. 31], hours [0 .. 24], minutes [0 .. 60] and seconds [0 .. 60]. At least one attribute must be defined. Undefined attributes imply the default value zero.
TimeToolkit	DataStructure with optional elements: <TimePoint>(startTime), <TimePoint>(stopTime), only used together with startTime <TimeInterval>(duration), <typ002:SpecialDay>(specialDay), <DaySelector>(daySelector).
Velocity	Velocity in integer units of metres per seconds [0 .. 255].
Weight	Weights in integer units of kilogrammes [0 .. 4 294 967 295].
MajorMinorVersion	Two part numeric value [MajorVersion.MinorVersion] with MajorVersion and MinorVersion each having a range of [0 .. 15]. An increase in the MajorVersion shall signal an incompatible version change. An increase in solely the MinorVersion shall signal a backwards compatible version change.
ByteFieldAttribute	Sequence of up to 65535 bytes.

The character encoding of strings within a service is specified in the TPEG SNI Application. In the binary physical format, multibyte character encodings shall use fewer characters if one or more characters in the specific string require more than one byte. The total number of bytes shall not exceed the number of characters specified for single-byte encoding.

A.2 Data type guidelines

A.2.1 MultiBytes

The IntUnLoMB and IntSiLoMB data types provide integer values with the same range as IntUnLo and IntSiLo respectively. In the TPEG binary format however, IntUnLoMB and IntSiLoMB constitute of a datastructure of variable size. If an application requires an integer variable having a large range but generally having smaller values, the IntUnLoMB and IntSiLoMB data types are preferred over their fixed-size versions.

Table A.2 — MultiByte ranges

Size	Range IntUnLoMB	Range IntSiLoMB
1 byte	[0 .. 127]	[-64 .. 63]

Table A.2 *(continued)*

Size	Range IntUnLoMB	Range IntSiLoMB
2 bytes	[0 .. 16 384]	[-8 192 .. 8 191]
3 bytes	[0 .. 2 097 151]	[-1 048 576 .. 1 048 575]
4 bytes	[0 .. 268 435 455]	[-134 217 728 .. 134 217 727]
5 bytes	[0 .. 4 294 967 295]	[-2 147 483 648 .. 2 147 483 647]

A.2.2 TimeInterval

The TimeInterval data structure can be used when an interval in time must be specified with more flexibility than the simple Duration type allows. Each TimeInterval attribute has a number of optional attributes. It is maximally 256 years long. Each attribute can be used as stand-alone attribute or in combination with other attributes. When an attribute is not given, the value zero is implied. Every TimeInterval must specify at least one attribute.

A.2.3 TimePoint

The TimePoint data structure can be used when a point in time must be specified with fewer granularities than the simple DateTime allows. Each TimePoint attribute has a number of optional attributes. Each attribute can be used as stand-alone attribute or in combination with other attributes. Every TimePoint must specify at least one attribute.

A.2.4 TimeToolkit

The TimeToolkit allows a set of date and time information to be described. Each TimeToolkit attribute has a number of optional attributes. Each attribute can be used as stand-alone attribute or in combination with other attributes. Every TimeToolkit must specify at least one attribute. For a timestamp the DateTime type should be used.

EXAMPLE An event has a start but no known end-time. In this case, only a start time is used but the end-time is omitted.

Annex B
(normative)

TPEG tables

In TPEG much information is based on tables. These tables represent clearly defined groupings of pre-defined concepts. The idea is to inform the device about the concept and let the device choose the best possible presentation of this concept in the context of the other parts of the TPEG message. This approach means that devices can present concepts, e.g. in any language or even as graphical icons.

This annex specifies the tables that are of general use for TPEG applications.

B.1 typ001:LanguageCode

Language code, based on the 2-alpha codes of ISO 639-1[3].

Table B.1 — typ001:LanguageCode

Code	Reference-English language name	Comment 2-alpha code
0	Unknown	
1	Afar	aa
2	Abkhazian	ab
3	Avestan	ae
4	Afrikaans	af
5	Akan	ak
6	Amharic	am
7	Aragonese	an
8	Arabic	ar
9	Assamese	as
10	Avaric	av
11	Aymara	ay
12	Azerbaijani	az
13	Bashkir	ba
14	Belarusian	be
15	Bulgarian	bg
16	Bihari	bh
17	Bislama	bi
18	Bambara	bm
19	Bengali	bn
20	Tibetan	bo
21	Breton	br
22	Bosnian	bs
23	Catalan	ca
24	Chechen	ce
25	Chamorro	ch
26	Corsican	co

Table B.1 *(continued)*

Code	Reference-English language name	Comment 2-alpha code
27	Cree	cr
28	Czech	cs
29	Church Slavic	cu
30	Chuvash	cv
31	Welsh	cy
32	Danish	da
33	German	de
34	Divehi	dv
35	Dzongkha	dz
36	Ewe	ee
37	Greek	el
38	English	en
39	Esperanto	eo
40	Spanish	es
41	Estonian	et
42	Basque	eu
43	Persian	fa
44	Fulah	ff
45	Finnish	fi
46	Fijian	fj
47	Faroese	fo
48	French	fr
49	Western Frisian	fy
50	Irish	ga
51	Scottish Gaelic	gd
52	Galician	gl
53	Guaraní	gn
54	Gujarati	gu
55	Manx	gv
56	Hausa	ha
57	Hebrew	he
58	Hindi	hi
59	Hiri Motu	ho
60	Croatian	hr
61	Haitian	ht
62	Hungarian	hu
63	Armenian	hy
64	Herero	hz
65	Interlingua (International Auxiliary Language Association)	ia
66	Indonesian	id
67	Interlingue	ie
68	Igbo	ig

Table B.1 *(continued)*

Code	Reference-English language name	Comment 2-alpha code
69	Sichuan Yi	ii
70	Inupiaq	ik
71	Ido	io
72	Icelandic	is
73	Italian	it
74	Inuktitut	iu
75	Japanese	ja
76	Javanese	jv
77	Georgian	ka
78	Kongo	kg
79	Kikuyu	ki
80	Kuanyama	kj
81	Kazakh	kk
82	Kalaallisut	kl
83	Khmer	km
84	Kannada	kn
85	Korean	ko
86	Kanuri	kr
87	Kashmiri	ks
88	Kurdish	ku
89	Komi	kv
90	Cornish	kw
91	Kirghiz	ky
92	Latin	la
93	Luxembourgish	lb
94	Ganda	lg
95	Limburgish	li
96	Lingala	ln
97	Lao	lo
98	Lithuanian	lt
99	Luba-Katanga	lu
100	Latvian	lv
101	Malagasy	mg
102	Marshallese	mh
103	Maori	mi
104	Macedonian	mk /sl
105	Malayalam	ml
106	Mongolian	mn
107	Moldavian	mo
108	Marathi	mr
109	Malay	ms
110	Maltese	mt
111	Burmese	my

Table B.1 *(continued)*

Code	Reference-English language name	Comment 2-alpha code
112	Nauru	na
113	Norwegian Bokmål	nb
114	North Ndebele	nd
115	Nepali	ne
116	Ndonga	ng
117	Dutch	nl
118	Norwegian Nynorsk	nn
119	Norwegian	no
120	South Ndebele	nr
121	Navajo	nv
122	Chichewa	ny
123	Occitan	oc
124	Ojibwa	oj
125	Oromo	om
126	Oriya	or
127	Ossetian	os
128	Panjabi	pa
129	Pa-li	pi
130	Polish	pl
131	Pashto	ps
132	Portuguese	pt
133	Quechua	qu
134	Raeto-Romance	rm
135	Kirundi	rn
136	Romanian	ro
137	Russian	ru
138	Kinyarwanda	rw
139	Sanskrit	sa
140	Sardinian	sc
141	Sindhi	sd
142	Northern Sami	se
143	Sango	sg
144	Serbo-Croatian	sh
145	Sinhalese	si
146	Slovak	sk
147	Slovenian	sl
148	Samoan	sm
149	Shona	sn
150	Somali	so
151	Albanian	sq
152	Serbian	sr
153	Swati	ss
154	Southern Sotho	st

Table B.1 *(continued)*

Code	Reference-English language name	Comment 2-alpha code
155	Sundanese	su
156	Swedish	sv
157	Swahili	sw
158	Tamil	ta
159	Telugu	te
160	Tajik	tg
161	Thai	th
162	Tigrinya	ti
163	Turkmen	tk
164	Tagalog	tl
165	Tswana	tn
166	Tonga	to
167	Turkish	tr
168	Tsonga	ts
169	Tatar	tt
170	Twi	tw
171	Tahitian	ty
172	Uighur	ug
173	Ukrainian	uk
174	Urdu	ur
175	Uzbek	uz
176	Venda	ve
177	Vietnamese	vi
178	Volapük	vo
179	Walloon	wa
180	Wolof	wo
181	Xhosa	xh
182	Yiddish	yi
183	Yoruba	yo
184	Zhuang	za
185	Chinese	zh
186	Zulu	zu

B.2 typ002:SpecialDay

The SpecialDay table lists special types of days, such as a public holiday or similar.

Table B.2 — typ002:SpecialDay

Code	Reference-English ‘word’	Comment	Example
0	Unknown		
1	Weekdays	Monday to Friday	
2	Weekends	Saturday and Sunday	
3	Holiday		

Table B.2 *(continued)*

Code	Reference-English 'word'	Comment	Example
4	public holiday		
5	religious holiday		e.g. Christmas day
6	federal holiday		
7	regional holiday		
8	national holiday		e.g. In UK: Mayday
9	school days		
10	every day		

B.3 typ003:CurrencyType

Currency type, based on the three-alpha codes of ISO 4217[5].

Table B.3 — typ003:CurrencyType

Code	Reference-English 'word'	Comment
0	Unknown	
1	AED	
2	AFN	
3	ALL	
4	AMD	
5	ANG	
6	AOA	
7	ARS	
8	AUD	
9	AWG	
10	AZM	
11	BAM	
12	BBD	
13	BDT	
14	BGN	
15	BHD	
16	BIF	
17	BMD	
18	BND	
19	BOB	
20	BRL	
21	BSD	
22	BTN	
23	BWP	
24	BYR	
25	BZD	
26	CAD	
27	CDF	
28	CHF	

Table B.3 *(continued)*

Code	Reference-English 'word'	Comment
29	CLP	
30	CNY	
31	COP	
32	CRC	
33	CSD	
34	CUP	
35	CVE	
36	CYP	
37	CZK	
38	DJF	
39	DKK	
40	DOP	
41	DZD	
42	EEK	
43	EGP	
44	ERN	
45	ETB	
46	EUR	
47	FJD	
48	FKP	
49	GBP	
50	GEL	
51	GGP	
52	GHS	
53	GIP	
54	GMD	
55	GNF	
56	GTQ	
57	GYD	
58	HKD	
59	HNL	
60	HRK	
61	HTG	
62	HUF	
63	IDR	
64	ILS	
65	IMP	
66	INR	
67	IQD	
68	IRR	
69	ISK	
70	JEP	
71	JMD	

Table B.3 *(continued)*

Code	Reference-English 'word'	Comment
72	JOD	
73	JPY	
74	KES	
75	KGS	
76	KHR	
77	KMF	
78	KPW	
79	KRW	
80	KWD	
81	KYD	
82	KZT	
83	LAK	
84	LBP	
85	LKR	
86	LRD	
87	LSL	
88	LTL	
89	LVL	
90	LYD	
91	MAD	
92	MDL	
93	MGA	
94	MKD	
95	MMK	
96	MNT	
97	MOP	
98	MRO	
99	MTL	
100	MUR	
101	MVR	
102	MWK	
103	MXN	
104	MYR	
105	MZN	
106	NAD	
107	NGN	
108	NIO	
109	NOK	
110	NPR	
111	NZD	
112	OMR	
113	PAB	
114	PEN	

Table B.3 *(continued)*

Code	Reference-English 'word'	Comment
115	PGK	
116	PHP	
117	PKR	
118	PLN	
119	PYG	
120	QAR	
121	RON	
122	RUB	
123	RWF	
124	SAR	
125	SBD	
126	SCR	
127	SDG	
128	SEK	
129	SGD	
130	SHP	
131	SIT	
132	SKK	
133	SLL	
134	SOS	
135	SPL	
136	SRD	
137	STD	
138	SVC	
139	SYP	
140	SZL	
141	THB	
142	TJS	
143	TMT	
144	TND	
145	TOP	
146	TRY	
147	TTD	
148	TVD	
149	TWD	
150	TZS	
151	UAH	
152	UGX	
153	USD	
154	UYU	
155	UZS	
156	VEF	
157	VND	

Table B.3 *(continued)*

Code	Reference-English ‘word’	Comment
158	VUV	
159	WST	
160	XAF	
161	XAG	
162	XAU	
163	XCD	
164	XDR	
165	XOF	
166	XPD	
167	XPF	
168	XPT	
169	YER	
170	ZAR	
171	ZMW	
172	ZWL	
173	RSD	
174	SSP	
255	Undefined	

B.4 typ004:NumericalMagnitude

The numerical magnitude, or “numag”, table contains unsigned integer values in the range 0 to 3000000 with decreasing precision.

The relation between code *n* and value *r* is given by:

$$r = (5 + \mathrm{sign}(n-5) \times (\mathrm{abs}(n-5) \bmod 45)) \times 10^{(n-5)\ \mathrm{div}\ 45}$$

Table B.4 — typ004:NumericalMagnitude

Code n:	Value r:
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12

Table B.4 *(continued)*

Code **n:**	**Value** **r:**
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28
29	29
30	30
31	31
32	32
33	33
34	34
35	35
36	36
37	37
38	38
39	39
40	40
41	41
42	42
43	43
44	44
45	45
46	46
47	47
48	48
49	49
50	50
51	60
52	70
53	80
54	90
55	100

Table B.4 *(continued)*

Code **n:**	**Value** **r:**
56	110
57	120
58	130
59	140
60	150
61	160
62	170
63	180
64	190
65	200
66	210
67	220
68	230
69	240
70	250
71	260
72	270
73	280
74	290
75	300
76	310
77	320
78	330
79	340
80	350
81	360
82	370
83	380
84	390
85	400
86	410
87	420
88	430
89	440
90	450
91	460
92	470
93	480
94	490
95	500
96	600
97	700
98	800

Table B.4 *(continued)*

Code n:	Value r:
99	900
100	1000
101	1100
102	1200
103	1300
104	1400
105	1500
106	1600
107	1700
108	1800
109	1900
110	2000
111	2100
112	2200
113	2300
114	2400
115	2500
116	2600
117	2700
118	2800
119	2900
120	3000
121	3100
122	3200
123	3300
124	3400
125	3500
126	3600
127	3700
128	3800
129	3900
130	4000
131	4100
132	4200
133	4300
134	4400
135	4500
136	4600
137	4700
138	4800
139	4900
140	5000
141	6000

Table B.4 *(continued)*

Code **n:**	**Value** **r:**
142	7000
143	8000
144	9000
145	10000
146	11000
147	12000
148	13000
149	14000
150	15000
151	16000
152	17000
153	18000
154	19000
155	20000
156	21000
157	22000
158	23000
159	24000
160	25000
161	26000
162	27000
163	28000
164	29000
165	30000
166	31000
167	32000
168	33000
169	34000
170	35000
171	36000
172	37000
173	38000
174	39000
175	40000
176	41000
177	42000
178	43000
179	44000
180	45000
181	46000
182	47000
183	48000
184	49000

Table B.4 *(continued)*

Code n:	Value r:
185	50000
186	60000
187	70000
188	80000
189	90000
190	100000
191	110000
192	120000
193	130000
194	140000
195	150000
196	160000
197	170000
198	180000
199	190000
200	200000
200	200000
201	210000
202	220000
203	230000
204	240000
205	250000
206	260000
207	270000
208	280000
209	290000
210	300000
211	310000
212	320000
213	330000
214	340000
215	350000
216	360000
217	370000
218	380000
219	390000
220	400000
221	410000
222	420000
223	430000
224	440000
225	450000
226	460000

Table B.4 *(continued)*

Code n:	Value r:
227	470000
228	480000
229	490000
230	500000
231	600000
232	700000
233	800000
234	900000
235	1000000
236	1100000
237	1200000
238	1300000
239	1400000
240	1500000
241	1600000
242	1700000
243	1800000
244	1900000
245	2000000
246	2100000
247	2200000
248	2300000
249	2400000
250	2500000
251	2600000
252	2700000
253	2800000
254	2900000
255	3000000

B.5 typ005:CountryCode

This table lists countries as defined by ISO 3166-1[6]. The comment column shows the corresponding 2-alpha code elements.

“undecodable country” is to be used by a client device unable to read the typ005 code used by a service provider — no code value for this word is ever transmitted.

Table B.5 — typ005:CountryCode

Code	Reference-English country name	Comment 2-alpha code
0	Unknown	
1	Afghanistan	AF
2	Åland Islands	AX