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

**Part 4:
UML to XML conversion rules**

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

Partie 4: Règles de conversion d'UML en XML



STANDARDSISO.COM : Click to view the full PDF of ISO 21219-4:2019



COPYRIGHT PROTECTED DOCUMENT

© ISO 2019

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

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	iv
Introduction	v
1 Scope	1
2 Normative references	1
3 Terms and definitions	1
4 Abbreviated terms	2
5 Rules for UML to XML format description conversion	2
5.1 Definition of XML format description.....	2
5.2 The tpegML schema definition.....	2
5.3 The tpegML xml file definition.....	3
5.4 XML data type ranges.....	3
5.5 Abstract data types.....	4
5.6 TPEG standard tables.....	5
5.7 Application tables (stereotype <<enumeration>>).....	5
5.8 Switching tables.....	6
5.9 Stereotype <<external>>.....	7
5.9.1 TPEG application <<external>> type.....	7
5.9.2 Non-TISA <<external>> XML schema.....	8
5.9.3 Non-TISA <<external>> binary type.....	9
5.10 Compound data types.....	10
5.10.1 Rule 1 — Classes.....	10
5.10.2 Rule 2 — Root element.....	11
5.10.3 Rule 3 — Attributes.....	11
5.10.4 Rule 4 — Specialisations/Abstract classes.....	13
5.11 Example.....	16
Annex A (normative) TPEG Data Types schema definition	23
Bibliography	40

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 21219-4:2015 which has been technically revised.

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: SP14004.

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

Part 4: UML to XML conversion rules

1 Scope

This document specifies the rules for converting TPEG application UML models to the tpegML format description. It contains the XML format definition of the abstract data types defined in ISO 21219-2. Rules for converting compound data types are also defined.

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.

ISO 8601 (series), *Data and time — Representations for information interchange*

ISO 21219-2, *Intelligent transport systems — Traffic and travel information (TTI) via transport protocol experts group, generation 2 (TPEG2) — Part 2: UML modelling rules*

ISO 21219-5, *Intelligent transport systems — Traffic and travel information (TTI) via transport protocol experts group, generation 2 (TPEG2) — Part 5: Service framework (TPEG2-SWF)*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 21219-2 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

Specification Identification

string that uniquely identifies a certain version of a certain TPEG application or toolkit

EXAMPLE The TPEG Traffic Event Compact application, version 2.0 is identified by the Specification Identification string "TEC_2_0".

3.2

Specification Name

string that verbosely describes a certain TPEG application or toolkit

EXAMPLE The TPEG Traffic Event Compact application, "TrafficEventCompact".

3.3

application namespace prefix

string of the lower case application/toolkit abbreviation as defined in the UML tagged value "ApplicationAbbreviation"

4 Abbreviated terms

The abbreviated terms in ISO 21219-2 and the following apply.

XML eXtensible Markup Language

XSD XML Schema Definition

UML Unified Modelling Language

app a placeholder for an application namespace prefix to create examples in this document. It is replaced by the lowercase application abbreviation of the relevant TPEG application.

5 Rules for UML to XML format description conversion

5.1 Definition of XML format description

The xml format description of TPEG applications is included in application and toolkit specifications as a normative annex. This annex shall be named according to the following scheme:

[Specification Identification], tpegML representation

The annex shall have two subclauses: Introduction and Application data type definition. The content of these subclauses is subject to the specifications in this clause.

The Introduction shall use a similar formulation as in the following:

This clause defines the tpegML format representation of the *[specification or toolkit name]* message components, datastructures and its attributes. For further descriptions of these objects see the related clauses *[reference to clauses]* in this document.

The application data type definition shall follow the rules that are specified in [5.10](#).

5.2 The tpegML schema definition

The xml format of TPEG applications will be described according to „XML Schema Definition“. To use a tpegML schemas, the `<xs:schema/>` tag shall contain the definition for the namespace <http://www.tisa.org/TPEG/TPEGDataTypes> *[MajorVersion]* *[MinorVersion]*, where the version numbers are the release versions of the specification as released by TISA (while the version numbers may be exemplarily in the subsequent tables).

Each tpegML schema shall contain at least the following definitions:

- Default namespace shall be set to the tpegML schema itself;
- Target namespace shall be set to the tpegML schema itself;
- The XSD elements shall be qualified;
- The *elementFormDefault* should be set to "qualified";
- The *attributesFormDefault* should be set to "qualified";
- The prefix and namespace of service framework tpegML schema shall be declared and imported (see ISO 21219-5).

The path to the tpegML schema shall have following syntax:

["http://www.tisa.org/TPEG/\[SpecificationIdentification\]"](http://www.tisa.org/TPEG/[SpecificationIdentification])

with

[Specification Identification] = [upper case Application or Toolkit abbreviation]_[Major version number]_[Minor version number]

The *schemaLocation* of all imported schemas shall be changed to local path for the validating process.

Accordingly, the resulting start tag of tpegML schema shall be:

```
<xs:schema xmlns="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:sfw="http://www.tisa.org/TPEG/ServiceFramework_0_0"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <xs:import namespace="http://www.tisa.org/TPEG/SFW_0_0"
    schemaLocation="http://www.tisa.org/TPEG/SFW_0_0.xsd" />
  ...
</xs:schema>
```

5.3 The tpegML xml file definition

For the xml file defined by the tpegML schema definition in 5.2, the following shall apply:

- No default name space shall be used. All elements and attributes shall have namespace prefixes. This enhances readability and prevents attribute naming problems with imported xsds.
- The name space of the applications and toolkits should be defined by using the application namespace prefix.
- The document encoding shall be "UTF-8".

Accordingly, the xml document will look similar to this example ("app" and "APP" used as placeholders):

```
<?xml version="1.0" encoding="UTF-8"?>
<app:applicationRootElement
  xmlns:app="http://www.tisa.org/TPEG/APP_0_0"
  xmlns:mmc="http://www.tisa.org/TPEG/MMC_0_0"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
  xmlns:sfw="http://www.tisa.org/TPEG/SFW_0_0"
  ...
</app:applicationRootElement>
```

The xml files defined by this document can hold one application message only (UML class tagged as "ApplicationRoot"). To encode several TPEG application messages, an xml framing shall be applied, for example as defined in ISO 21219-5.

5.4 XML data type ranges

XML data type	Range
xs:byte	-128 .. 127

XML data type	Range
xs:short	–32768 .. 32767
xs:int	–2147483648 .. 2147483647
xs:integer	–infinite .. infinite
xs:unsignedByte	0 .. 255
xs:unsignedShort	0 .. 65535
xs:unsignedInt	0 .. 4294967295
xs:float	$m \cdot 2^e$, where $-2^{24} < m \leq 2^{24}$ and $-149 < e \leq 104$

5.5 Abstract data types

The UML attributes of abstract data types shall be converted to the XSD local elements using the “TPEG Data Types” schema [see [Annex A](#)]. The “TPEG Data Type” schema contains the XSD definition for all TPEG abstract data types. To use “TPEG Data Types” schema, the <xs:schema/> tag shall contain the definition for the namespace [http://www.tisa.org/TPEG/TPEGDataTypes_\[MajorVersion\]_\[MinorVersion\]](http://www.tisa.org/TPEG/TPEGDataTypes_[MajorVersion]_[MinorVersion]), where the version numbers are the release versions of the specification as released by TISA. Additionally, the schema shall be imported by adding the <xs:import/> element (see XSD example). The definition of the “schemaLocation” attribute within the <xs:import /> element can differ from the defined namespace URI but it shall contain the valid URI where the “TPEG Data Type” schema is to be found.

To achieve a common appearance of all tpegML schemas it is recommended to use “tdt” prefix for the “TPEG Data Types” namespace.

When using an UML attribute of abstract data type, the tpegML schema shall contain the following definition:

```
<xs:element name="UMLAttributeName" type="tdt:AbstractDataType"/>
```

where *UMLAttributeName* is the name of the attribute as defined in UML, *tdt* is the prefix of the “TPEG Data Types” namespace and *AbstractDataType* is one of the abstract data types defined in ISO 21219-2.

The syntax of abstract data types in tpegML format is described in [Annex A](#).

XSD example:

```
<xs:schema xmlns="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <xs:import namespace="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
    schemaLocation="http://www.tisa.org/TPEG/TPEGDataTypes_0_0.xsd"/>
  <xs:complexType name="class1">
    <xs:sequence>
      <xs:element name="attr1" type="tdt:FixedPercentage"/>
      <xs:element name="attr2" type="tdt:Boolean"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

XML example:

```
<?xml version="1.0"?>
<app:class1 xmlns:app="http://www.tisa.org/TPEG/[Specification Identification]">
  <app:attr1>3</app:attr1>
  <app:attr2>true</app:attr2>
</app:class1>
```

5.6 TPEG standard tables

The UML attributes of the TPEG standard table type shall be converted similar to abstract data types. The “TPEG Data Types” schema contains definition for all TPEG standard tables.

The syntax of abstract data types is described in [Annex A](#).

XSD example:

```
<xs:schema xmlns="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <xs:complexType name="class1">
    <xs:sequence>
      <xs:element name="attr1" type="tdt:typ001_LanguageCode"/>
      <xs:element name="attr2" type="tdt:typ002_SpecialDay"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

XML example:

```
<?xml version="1.0"?>
<app:class1 xmlns:app="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0">
  <app:attr1 tdt:table="typ001_LanguageCode" tdt:code="33" />
  <app:attr2 tdt:table="typ002_SpecialDay" tdt:code="2" />
</app:class1>
```

5.7 Application tables (stereotype <<enumeration>>)

The UML class with stereotype <<enumeration>> shall be converted to the XSD with the same syntax as TPEG standard tables. The colon in the class name shall be replaced by an underscore “_”. The value of the attribute “table” shall always contain this class name as fixed value.

To use a UML attribute of the application table type, the XSD shall contain the following complex type.

XSD example:

```
<xs:complexType name="xxxxyy_applicationTableName">
  <xs:attribute name="table" type="xs:string" fixed="xxxxyy_applicationTableName" use="required"/>
  <xs:attribute name="code" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:unsignedByte">
        <!-- here the restrictions can be defined -->
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>
```

where *applicationTableName* is the name of the application table, *xxxxyy* is the application table prefix [1], and *code* is an integer value up to 255

Accordingly the UML attribute of this type shall be:

```
<xs:element name="UMLAttributeName" type="xxxxyy_applicationTableName">
```

5.8 Switching tables

Each child table of a switching table group shall be converted to the XSD according to the rule in 5.7. The abstract parent class of a switching table group will be defined as standard application table with the following conditions:

- The “table” attribute shall be defined as enumeration of all child table names;
- The “code” attribute shall not have any restrictions and shall always be set to “xs:unsignedByte” type.

The XSD example of the abstract blank class for a switching table group:

```
<xs:complexType name="SubTableType">
  <xs:attribute name="table" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="xyz101_SubTableType1"/>
        <xs:enumeration value="xyz102_SubTableType2"/>
        ...
        <xs:enumeration value="xyz10N_SubTableTypeN"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="code" type="xs:unsignedByte" use="required"/>
</xs:complexType>
```

The XSD example for usage of a switching table type linked to a main table:

```
<xs:complexType name="someClass">
...
  <xs:element name="mainTable" type="xyz100_TableType"/>
  <xs:element name="switchedTable" type="SubTableType"/>
...
</xs:complexType>
```

XML example for usage of a switching table depending on the set main table value:

```
<someClass>
  <mainTable table="xyz100_MainTableType" code="1" />
  <switchedTable table=" xyz101_SubTableType1" code="100" />
</someClass>
...
<someClass>
  <mainTable table="xyz100_MainTableType" code="2" />
  <switchedTable table="xyz101_SubTableType2" code="20" />
</someClass>
```

There is no formal specification of the lookup value in the main table and the related switching table to choose. The default relation is supposed to be 1 to 1. The special relation shall be defined in application or toolkit specification.

If the restriction of the switching table "code" attribute is checked, the defined table for selected switching table name shall be used.

5.9 Stereotype <<external>>

The class with the stereotype <<external>> defines a type which is defined in an external specification. To use the <<external>> type, the external specification shall provide either XML schema definition or binary representation of this type. This rule defines usage of <<external>> type for the following external specifications:

- TISA TPEG application specification containing the <<external>> type;
- Non-TISA specification containing XML schema definition for the <<external>> type;
- Non-TISA specification containing only binary definition for the <<external>> type.

5.9.1 TPEG application <<external>> type

To use a class with the stereotype <<external>> defined in a TISA TPEG application specification, the prefix and the namespace of the <<external>> TISA TPEG application shall be defined and imported. The prefix will be used then to qualify the <<external>> type. The prefix shall be set to the value defined by the "tpegMLprefix" class tagged value. The namespace URI shall be set to the value defined by the "tpegMLuri" class tagged value.

Additionally, the XML schema defined by the namespace shall be imported by adding the <xs:import/> element (see XSD example). The definition of the "schemaLocation" attribute within the <xs:import/> element can differ from the defined namespace URI but it shall contain the valid URI where the <<external>> TPEG application XML schema is to be found.

The local element from <<external>> type shall be defined with type name set to class name from which the class stereotyped as <<external>> is derived. The name of class stereotyped as <<external>> is a placeholder only for the name of the class defined in a TPEG application specification.

Abstract classes may neither be imported nor used via this method. Instead each child class type shall be imported.

XSD example:

```
<xs:schema xmlns="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
  <!-- Here, the external namespace is included: -->
  xmlns:ext1="http://www.tisa.org/TPEG/[External Application Identification]"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <!-- Here, import the external namespace: -->
  <xs:import namespace="http://www.tisa.org/TPEG/[External Application Identification]"
    schemaLocation="http://www.tisa.org/TPEG/[External Application Identification].xsd" />
  <xs:complexType name="class1">
    <xs:sequence>
      <xs:element name="attr1" type="tdt:FixedPercentage"/>
      <xs:element name="attr2" type="tdt:Boolean"/>
      <!-- Here, the externally defined attribute type is used: -->
      <xs:element name="attr3" type="ext1:someExternalClass"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

XML example:

```
<?xml version="1.0"?>
<app:class1 xmlns:app="http://www.tisa.org/TPEG/[Application Identification]"
  <!-- Here, the external namespace is included: -->
  xmlns:ext1="http://www.tisa.org/TPEG/[External Application Identification]">
  <app:attr1>3</app:attr1>
  <app:attr2>true</app:attr2>
  <app:attr3>
    <ext1:attr1>true</ext1:attr1>
    ...
  </app:attr3>
</app:class1>
```

5.9.2 Non-TISA <<external>> XML schema

To use a class with the stereotype <<external>> defined in a non-TISA specification, but providing an XML schema for the affected class, the same rule will be applied as for TPEG application <<external>> type except that the local element type name shall be set to the name of the class stereotyped as

<<external>>. The class name, stereotyped as <<external>>, is then the name of the class in the external XML schema.

XSD example:

```
<xs:schema xmlns="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
  <!-- Here, the external namespace is included: -->
  xmlns:ext1="http://www.someuri.org"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <!-- Here, import the external namespace: -->
  <xs:import namespace="http://www.someuri.org"
    schemaLocation="http://www.someuri.org/schema.xsd" />
  <xs:complexType name="class1">
    <xs:sequence>
      <xs:element name="attr1" type="tdt:FixedPercentage"/>
      <xs:element name="attr2" type="tdt:Boolean"/>
      <!-- Here, the externally defined attribute type is used: -->
      <xs:element name="attr3" type="ext1:someExternalClass"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

XML example:

```
<?xml version="1.0"?>
<app:class1 xmlns:app="http://www.tisa.org/TPEG/[Application Identification]"
  <!-- Here, the external namespace is included: -->
  xmlns:ext1="http://www.someuri.org">
  <app:attr1>3</app:attr1>
  <app:attr2>true</app:attr2>
  <app:attr3>
    <ext1:attr1>true</ext1:attr1>
    ...
  </app:attr3>
</app:class1>
```

5.9.3 Non-TISA <<external>> binary type

To use a class with the stereotype <<external>> defined in a non-TISA specification, which provides only binary representation for the affected class, the local element type shall be set to "xs:base64Binary".

XSD example:

```
<xs:schema xmlns="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.tisa.org/TPEG/[Specification Identification]"
  xmlns:tdt="http://www.tisa.org/TPEG/TPEGDataTypes_0_0"
  xmlns:ext1="http://www.tisa.org/TPEG/[External Application Identification]"
  elementFormDefault="qualified"
  attributeFormDefault="qualified">
  <xs:complexType name="class1">
    <xs:sequence>
      <xs:element name="attr1" type="tdt:FixedPercentage"/>
      <xs:element name="attr2" type="tdt:Boolean"/>
      <!-- Here, the externally defined attribute type is used: -->
      <xs:element name="attr3" type="xs:base64Binary"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

XML example:

```
<?xml version="1.0"?>
<app:class1 xmlns:app="http://www.tisa.org/TPEG/[Application Identification]"
  <!-- Here, the external namespace is included: -->
  xmlns:ext1="http://www.tisa.org/TPEG/[External Application Identification]">
  <app:attr1>3</app:attr1>
  <app:attr2>true</app:attr2>
  <app:attr3>VGhhZCBpcyB0aGUgYmluYXJ5IHRlc2VudGF0aW9uIG9mIGEgdm9uLVRJU0EgZXh0ZXJu
  YWwg3BIY2lmaWNhdGlvbg==</app:attr3>
</app:class1>
```

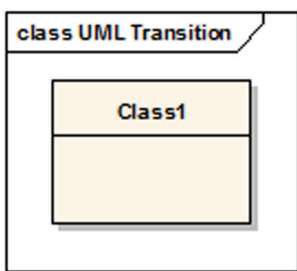
5.10 Compound data types

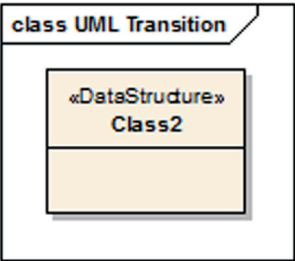
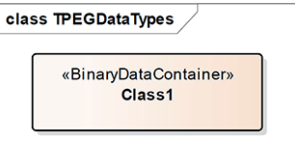
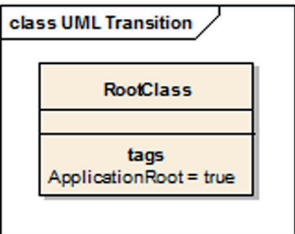
5.10.1 Rule 1 — Classes

A class shall be converted into an XSD ComplexType with the name of the class.

The class stereotype <<DataStructure>> shall be ignored.

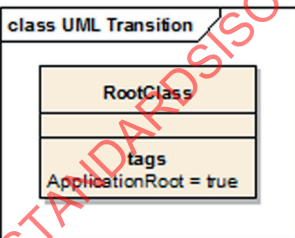
The complex type of a class with tagged value *ApplicationRoot* shall be derived from *ApplicationRootMessageML* defined in TPEG Service Framework specification ISO 21219-5.

UML	XSD definition
 <pre>classDiagram class UML_Transition { +Class1 } class Class1</pre>	<pre><xs:complexType name="Class1"> </xs:complexType></pre>

UML	XSD definition
 <pre> classDiagram class UMLTransition { <<DataStructure>> Class2 } </pre>	<pre> <xs:complexType name="Class2"> </xs:complexType> </pre>
 <pre> classDiagram class TPEGDataTypes { <<BinaryDataContainer>> Class1 } </pre>	<pre> <xs:element name="Class1" type="xs:base64Binary" /> </pre> XML Example: <pre> <Class1>QmluYXJ5IHRlc3RkYXRh</Class1> </pre>
 <pre> classDiagram class UMLTransition { RootClass tags ApplicationRoot = true } </pre>	<pre> <xs:complexType name="RootClass"> <xs:complexContent> <xs:extension base="sfw:ApplicationRootMessageML"> <xs:sequence> ... </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType> </pre>

5.10.2 Rule 2 — Root element

A class with tagged value *ApplicationRoot* shall be present in an XSD as a global element with the fixed name and the type of the corresponding XSD ComplexType. The name of the element shall be "ApplicationRootMessageML".

UML	XSD definition
 <pre> classDiagram class UMLTransition { RootClass tags ApplicationRoot = true } </pre>	<pre> <xs:element name="ApplicationRootMessageML" type="RootClass" /> </pre>

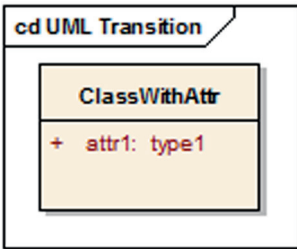
5.10.3 Rule 3 — Attributes

Rule 3 consists of four sub-rules, each specifying how to handle class attributes.

5.10.3.1 Rule 3a — Datatypes

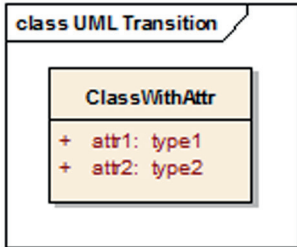
Attributes of primitive data type shall be converted into xml format as defined in [5.5](#) and [5.7](#).

Attributes of compound data type shall be converted into xml format according to the rules in this [clause \(5.10\)](#).

UML	XSD definition
	<pre><xs:complexType name="ClassWithAttr"> <xs:sequence> <xs:element name="attr1" type="type1"/> </xs:sequence> </xs:complexType></pre>

5.10.3.2 Rule 3b — Ordering

The order of the attributes as listed in the UML model shall be preserved.

UML	XSD definition
	<pre><xs:complexType name="ClassWithAttr"> <xs:sequence> <xs:element name="attr1" type="type1"/> <xs:element name="attr2" type="type2"/> </xs:sequence> </xs:complexType></pre>

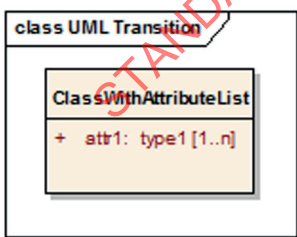
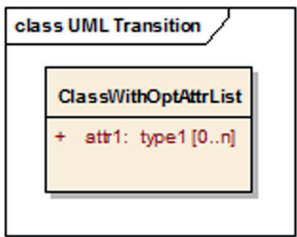
5.10.3.3 Rule 3c — Single multiplicity [minOccurs = 1..maxOccurs = 1]

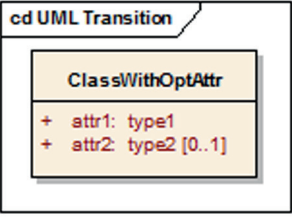
No additional rules apply.

5.10.3.4 Rule 3d — Multiplicity [minOccurs = 0..maxOccurs = 1], Multiplicity [minOccurs = 0..maxOccurs > 1] and Multiplicity [minOccurs = 1..maxOccurs > 1]

The lower bound and upper bound of the attribute multiplicity shall be directly converted to “minOccurs” and “maxOccurs” XML schema statements.

If the upper bound is set to “*” value the “maxOccurs” value shall be set then to “unbounded”.

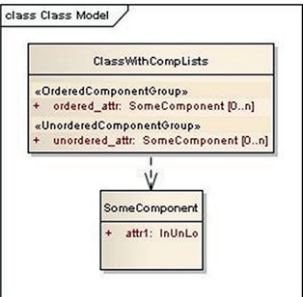
UML	XSD definition
	<pre><xs:complexType name="ClassWithAttributeList"> <xs:sequence> <xs:element name="attr1" type="type1" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType></pre>
	<pre><xs:complexType name="ClassWithOptAttributeList"> <xs:sequence> <xs:element name="attr1" type="type1" minOccurs="0" maxOccurs="unbounded"/> </xs:sequence> </xs:complexType></pre>

UML	XSD definition
 <pre> classDiagram class ClassWithOptAttr { +attr1: type1 +attr2: type2 [0..1] } </pre>	<pre> <xs:complexType name="ClassWithOptAttr"> <xs:sequence> <xs:element name="attr1" type="type1" /> <xs:element name="attr2" type="type2" minOccurs="0"/> </xs:sequence> </xs:complexType> </pre>

5.10.3.5 Rule 3e — Attributes stereotyped with <<OrderedComponentGroup>> or <<UnorderedComponentGroup>>

The attributes stereotyped with <<OrderedComponentGroup>> shall be converted as a normal attribute.

The attributes stereotyped as <<UnorderedComponentGroup>> shall be combined under the xs:choice element. Independent of the multiplicity of such attributes the defined element shall be always optional. Multiple occurrence of such attributes will be achieved by setting the occurrence of the xs:choice element to unbounded.

UML	XSD definition
 <pre> classDiagram class ClassWithCompLists { <<OrderedComponentGroup>> +ordered_attr: SomeComponent [0..n] <<UnorderedComponentGroup>> +unordered_attr: SomeComponent [0..n] } class SomeComponent { +attr1: InUnLo } ClassWithCompLists --> SomeComponent </pre>	<pre> <xs:complexType name="ClassWithCompLists"> <xs:sequence> <xs:element name="ordered_attr" type="SomeComponent" minOccurs="0" maxOccurs="unbounded"/> <xs:choice maxOccurs="unbounded"> <xs:element name="unordered_attr" type="SomeComponent" minOccurs="0" /> </xs:choice> </xs:sequence> </xs:complexType> </pre>

5.10.4 Rule 4 — Specialisations/Abstract classes

5.10.4.1 Rule 4a — Inheritance

The UML inheritance is resolved in tpegML format by copying all attributes from parent class into the child classes. The usage of XML inheritance feature is not allowed. The abstract parent class shall not be available in the resulting XML schema unless it will be used for class switching (see Rule 5b). The parent attributes shall occur before child attributes.

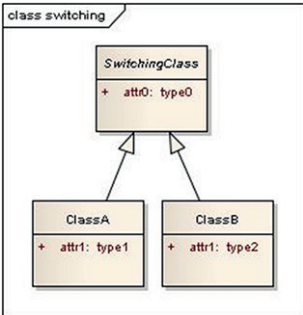
NOTE The use of inheritance is not allowed if a child class is stereotyped as <<external>>.

UML	XSD definition
<pre>classDiagram class ParentClass { +attr1: type1 } class ChildClass1 { +attr1_child1: type2 } class ChildClass2 { +attr1_child2: type3 } ParentClass < -- ChildClass1 ParentClass < -- ChildClass2</pre>	<pre><xs:complexType name="ChildClass1"> <xs:complexContent> <xs:sequence> <xs:element name="attr1" type="type1"/> <xs:element name="attr1_child1" type="type2"/> </xs:sequence> </xs:complexContent> </xs:complexType> <xs:complexType name="ChildClass2"> <xs:complexContent> <xs:sequence> <xs:element name="attr1" type="type1"/> <xs:element name="attr1_child2" type="type3"/> </xs:sequence> </xs:complexContent> </xs:complexType></pre>

5.10.4.2 Rule 4b — Switching

The switching mechanism allows the generalization to be used as a selector mechanism to select only one of the child components for instantiating. To convert this mechanism to tpegML notation, the UML inheritance shall first be resolved according to rule 4a. The abstract parent class will then build the switching class.

A complex type with name of the abstract parent class shall be created. This complex type shall contain only the xs:choice of the multiplicity [1..1]. For each derived child component ,a choice element with name and type of the derived child component and multiplicity of [1..1] shall be created. The name of the choice element shall be prefixed with string “option”.

UML	XSD definition
 <pre> classDiagram class SwitchingClass { +attr0: type0 } class ClassA { +attr1: type1 } class ClassB { +attr1: type2 } ClassA -- > SwitchingClass ClassB -- > SwitchingClass </pre>	<pre> < xs:complexType name = "SwitchingClass" > < xs:choice > < xs:element name = "optionClassA" type = "ClassA"/ > < xs:element name = "optionClassB" type = "ClassB"/ > < /xs:choice > < /xs:complexType > < xs:complexType name = "ClassA" > < xs:complexContent > < xs:sequence > < xs:element name = "attr0" type = "type0"/ > < xs:element name = "attr1" type = "type1"/ > < /xs:sequence > < /xs:complexContent > < /xs:complexType > < xs:complexType name = "ClassB" > < xs:complexContent > < xs:sequence > < xs:element name = "attr0" type = "type0"/ > < xs:element name = "attr1" type = "type2"/ > < /xs:sequence > < /xs:complexContent > < /xs:complexType > </pre>

To use the switch construct an element shall be defined with type of switching class name.

XSD example:

```

< xs:complexType name = "SomeClass"
  ...
  < xs:element name = "someattr" type = "SwitchingClass" / >
  ...
< /xs:complexType >
  
```

XML example:

```
< someattr >
  < optionClassA >
    < attr0 > 123 </attr0 >
    < attr1 > 12345 </attr1 >
  </optionClassA >
</someattr >
or
</someattr >
  < optionClassB >
    < attr0 > 123 </attr0 >
    < attr1 > abcdef </attr1 >
  </optionClassB >
</someattr >
```

5.11 Example

NOTE These examples are intended to be exemplary for the xml format and not for an actual application.

tpegML schema example:

```
< ?xml version = "1.0" encoding = "UTF-8"? >
< xs:schema xmlns = "http://www.tisa.org/TPEG/APP\_0\_0"
targetNamespace = "http://www.tisa.org/TPEG/APP\_0\_0"
xmlns:xs = "http://www.w3.org/2001/XMLSchema"
xmlns:swf = "http://www.tisa.org/TPEG/SFW\_0\_0"
xmlns:tdt = "http://www.tisa.org/TPEG/TPEGDataTypes\_0\_0"
xmlns:mmc = "http://www.tisa.org/TPEG/MMC\_0\_0"
xmlns:lrc = "http://www.tisa.org/TPEG/LRC\_0\_0"
elementFormDefault = "qualified"
attributeFormDefault = "qualified" >
< xs:import namespace = "http://www.tisa.org/TPEG/SFW\_0\_0" schemaLocation =
"http://www.tisa.org/TPEG/SFW\_0\_0"/>
< xs:import namespace = "http://www.tisa.org/TPEG/TPEGDataTypes\_0\_0" schemaLocation =
"http://www.tisa.org/TPEG/TPEGDataTypes\_0\_0"/>
< xs:import namespace = "http://www.tisa.org/TPEG/MMC\_0\_0" schemaLocation =
"http://www.tisa.org/TPEG/MessageManagementContainer\_0\_0"/>
< xs:import namespace = "http://www.tisa.org/TPEG/LRC\_0\_0" schemaLocation =
"http://www.tisa.org/TPEG/LRC\_0\_0"/>
< xs:element name = "APPMessage" type = "APPMessage"/ >
< xs:complexType name = "APPMessage" >
  < xs:complexContent >
```

```

< xs:extension base = "swf:ApplicationRootMessageML" >
  < xs:sequence >
    < xs:element name = "mmt" type = "MMCSwitch"/ >
    < xs:element name = "event" type = "A" minOccurs = "0"/ >
    < xs:element name = "loc" type = "lrc:LocationReferencingContainer" minOccurs = "0"/ >
  < /xs:sequence >
< /xs:extension >
< /xs:complexContent >
< /xs:complexType >
< xs:complexType name = "MMCSwitch" >
  < xs:sequence >
    < xs:choice >
      < xs:element name = "optionMasterMessageLink" type = "mmc:MMCMasterMessage"/ >
      < xs:element name = "optionMasterPartLink" type = "mmc:MMCMessagePart" / >
      < xs:element name = "optionMessageManagementContainerLink" type =
"mmc:MessageManagementContainer" / >
    < /xs:choice >
  < /xs:sequence >
< /xs:complexType >
< xs:complexType name = "B" >
  < xs:sequence >
    < xs:element name = "attr_enum1" type = "app001_Enum1"/ >
    < xs:element name = "attr_dstruct" type = "A_DS" minOccurs = "0" maxOccurs = "unbounded"/ >
    < xs:element name = "attr_enum2" type = "app002_Enum2" minOccurs = "0"/ >
    < xs:element name = "oc_1" type = "E" minOccurs = "0" maxOccurs = "unbounded"/ >
  < /xs:sequence >
< /xs:complexType >
< xs:complexType name = "A" >
  < xs:sequence >
    < xs:element name = "attr1" type = "tdt:DateTime" minOccurs = "0"/ >
    < xs:element name = "attr4" type = "tdt:LocalizedShortString" minOccurs = "1" maxOccurs = "10"/ >
    < xs:element name = "attr5" type = "B"/ >
    < xs:element name = "attr6" type = "tdt:LocalizedShortString" minOccurs = "0" maxOccurs =
"unbounded"/ >

```

```

< xs:element name = "attr7" type = "tdt:IntUnLoMB" minOccurs = "0"/ >
< xs:element name = "attr8" type = "tdt:LocalizedShortString"
minOccurs = "0" maxOccurs = "unbounded"/ >
< xs:element name = "attr9" type = "tdt:LocalizedShortString" minOccurs = "0" maxOccurs = "15"/ >
< xs:element name = "oc_1" type = "B" minOccurs = "0" maxOccurs = "unbounded"/ >
< xs:element name = "oc_2" type = "D" minOccurs = "0"/ >
< xs:element name = "oc_3" type = "B" minOccurs = "0"/ >
< /xs:sequence >
< /xs:complexType >
< xs:complexType name = "D" >
< xs:sequence >
< xs:element name = "attr1" type = "tdt:IntUnLoMB" minOccurs = "0"/ >
< xs:element name = "attr2" type = "tdt:DateTime" minOccurs = "0"/ >
< xs:element name = "oc_1" type = "E_Parent" maxOccurs = "unbounded"/ >
< /xs:sequence >
< /xs:complexType >
< xs:complexType name = "E_Parent" >
< xs:sequence >
< xs:choice >
< xs:element name = "optionE" type = "E"/ >
< xs:element name = "optionF" type = "F"/ >
< /xs:choice >
< /xs:sequence >
< /xs:complexType >
< xs:complexType name = "E" >
< xs:sequence >
< xs:element name = "attr1_parent" type = "tdt:DateTime" minOccurs = "0"/ >
< xs:element name = "attr2_parent" type = "tdt:Duration"/ >
< xs:element name = "attr1" type = "tdt:IntUnTi" minOccurs = "0"/ >
< xs:element name = "attr2" type = "tdt:Boolean" minOccurs = "0"/ >
< xs:element name = "attr3" type = "tdt:Boolean" maxOccurs = "unbounded"/ >
< xs:element name = "attr4" type = "tdt:Boolean"/ >
< xs:element name = "attr5" type = "tdt:Boolean" minOccurs = "0" maxOccurs = "unbounded"/ >
< xs:element name = "attr6" type = "app001_Enum1"/ >

```

```

    < xs:element name = "attr7" type = "SubTableType1"/ >
  < /xs:sequence >
< /xs:complexType >
< xs:complexType name = "F" >
  < xs:sequence >
    < xs:element name = "attr1_parent" type = "tdt:DateTime" minOccurs = "0"/ >
    < xs:element name = "attr2_parent" type = "tdt:Duration"/ >
    < xs:element name = "attr1" type = "tdt:Float"/ >
  < /xs:sequence >
< /xs:complexType >
< xs:complexType name = "A_DS" >
  < xs:sequence >
    < xs:element name = "attr1" type = "tdt:Boolean" minOccurs = "0"/ >
    < xs:element name = "attr2" type = "tdt:IntUnTi"/ >
    < xs:element name = "attr3" type = "tdt:IntUnTi" minOccurs = "0"/ >
    < xs:element name = "attr4" type = "tdt:IntUnLoMB" minOccurs = "0"/ >
    < xs:element name = "attr5" type = "tdt:IntUnLoMB" minOccurs = "0"/ >
  < /xs:sequence >
< /xs:complexType >
< xs:complexType name = "SubTableType1" >
  < xs:attribute name = "table" use = "required" >
    < xs:simpleType >
      < xs:restriction base = "xs:string" >
        < xs:enumeration value = "app101_SubEnum1"/ >
        < xs:enumeration value = "app102_SubEnum2"/ >
        < xs:enumeration value = "app103_SubEnum3"/ >
      < /xs:restriction >
    < /xs:simpleType >
  < /xs:attribute >
  < xs:attribute name = "code" type = "xs:unsignedByte" use = "required"/ >
< /xs:complexType >
< xs:complexType name = "app001_Enum1" >
  < xs:attribute name = "table" type = "xs:string" fixed = "app001_Enum1" use = "required"/ >

```

```

< xs:attribute name = "code" use = "required" >
  < xs:simpleType >
    < xs:restriction base = "xs:unsignedByte" >
      < xs:minInclusive value = "0"/ >
      < xs:maxInclusive value = "6"/ >
    < /xs:restriction >
  < /xs:simpleType >
< /xs:attribute >
< /xs:complexType >
< xs:complexType name = "app002_Enum2" >
  < xs:attribute name = "table" type = "xs:string" fixed = "app002_Enum2" use = "required"/ >
  < xs:attribute name = "code" use = "required" >
    < xs:simpleType >
      < xs:restriction base = "xs:unsignedByte" >
        < xs:minInclusive value = "0"/ >
        < xs:maxInclusive value = "24"/ >
      < /xs:restriction >
    < /xs:simpleType >
  < /xs:attribute >
< /xs:complexType >
< xs:complexType name = "app101_SubEnum1" >
  < xs:attribute name = "table" type = "xs:string" fixed = "app101_SubEnum1" use = "required"/ >
  < xs:attribute name = "code" use = "required" >
    < xs:simpleType >
      < xs:restriction base = "xs:unsignedByte" >
        < xs:minInclusive value = "0"/ >
        < xs:maxInclusive value = "2"/ >
      < /xs:restriction >
    < /xs:simpleType >
  < /xs:attribute >
< /xs:complexType >
< xs:complexType name = "app102_SubEnum2" >
  < xs:attribute name = "table" type = "xs:string" fixed = "app102_SubEnum2" use = "required"/ >

```

```

< xs:attribute name = "code" use = "required" >
  < xs:simpleType >
    < xs:restriction base = "xs:unsignedByte" >
      < xs:minInclusive value = "0"/ >
      < xs:maxInclusive value = "5"/ >
    < /xs:restriction >
  < /xs:simpleType >
< /xs:attribute >
< /xs:complexType >
< xs:complexType name = "app103_SubEnum3" >
  < xs:attribute name = "table" type = "xs:string" fixed = "app103_SubEnum3" use = "required"/ >
  < xs:attribute name = "code" use = "required" >
    < xs:simpleType >
      < xs:restriction base = "xs:unsignedByte" >
        < xs:minInclusive value = "0"/ >
        < xs:maxInclusive value = "3"/ >
      < /xs:restriction >
    < /xs:simpleType >
  < /xs:attribute >
< /xs:complexType >
< /xs:schema >

```

XML example:

```

<?xml version = "1.0" encoding = "UTF-8"? >
< app:APPMessage xmlns:app = "http://www.tisa.org/TPEG/APP\_0\_0"
xmlns:mmc = "http://www.tisa.org/TPEG/MMC\_0\_0"
xmlns:tdt = "http://www.tisa.org/TPEG/TPEGDataTypes\_0\_0"
xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance">
  < app:mmt >
    < app:optionMasterMessageLink >
      < mmc:messageID > 0 < /mmc:messageID >
      < mmc:versionID > 255 < /mmc:versionID >
      < mmc:messageExpiryTime > 2001-12-17T09:30:47.0Z < /mmc:messageExpiryTime >
      < mmc:cancelFlag > true < /mmc:cancelFlag >
      < mmc:messageGenerationTime > 2001-12-17T09:30:47.0Z < /mmc:messageGenerationTime >
      < mmc:priority tdt:code = "0" tdt:table = "typ007_Priority"/ >
      < mmc:multiPartMessageDirectory >
        < mmc:partID > 255 < /mmc:partID >
        < mmc:partType mmc:code = "255" mmc:table = "mmc001_PartType"/ >
      < /mmc:multiPartMessageDirectory >
    < /app:optionMasterMessageLink >
  < /app:mmt >
  < app:event >
    < app:attr1 > 2001-12-17T09:30:47.0Z < /app:attr1 >
    < app:attr4 >
      < tdt:languageCode tdt:code = "0" tdt:table = "typ001_LanguageCode"/ >
      < tdt:value > abcdef < /tdt:value >
    < /app:attr4 >
    < app:attr5 >
      < app:attr_enum1 app:code = "0" app:table = "app001_Enum1"/ >
      < app:attr_dstruct >
        < app:attr1 > true < /app:attr1 >
        < app:attr2 > 255 < /app:attr2 >
        < app:attr3 > 255 < /app:attr3 >
        < app:attr4 > 0 < /app:attr4 >
        < app:attr5 > 0 < /app:attr5 >
      < /app:attr_dstruct >
    < /app:attr5 >
  < /app:event >
< /app:APPMessage >

```

Annex A (normative)

TPEG Data Types schema definition

A.1 Schema definition

The “TPEG Data Types” schema contains the XML schema definition for all TPEG abstract data types and TPEG standard tables. The schema has the following declarations:

```
< xs:schema xmlns = "http://www.tisa.org/TPEG/TPEGDataTypes 2 1"
  xmlns:xs = "http://www.w3.org/2001/XMLSchema"
  targetNamespace = "http://www.tisa.org/TPEG/TPEGDataTypes 2 1"
  elementFormDefault = "qualified"
  attributeFormDefault = "qualified" >
  ...
< /xs:schema >
```

A.2 Abstract data types

Data type	XML format definition (XML element type)
BitArray	The bits in a BitArray are encoded as a sequence of '0' or '1' characters. The first character in the string is the bit 1. If in a BitArray all bits after a certain set bit are not set, the remaining characters containing only unset bits may be removed. EXAMPLE BitArray = 0A hex (00001010b); Bit 5 and bit 7 are set. Bit 7 is the last set bit, therefore the BitArray string is “0000101”. <u>XSD definition:</u> <pre>< xs:simpleType name = "BitArray" > < xs:restriction base = "xs:string" > < xs:pattern value = "(0 1)*" / > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:BitArray" / ></pre> where <i>tdt</i> is the prefix of the “TPEG Data Types” namespace. <u>XML example:</u> <pre>< UMLAttributeName > 011011010101 < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
Boolean	<u>XSD definition:</u> <pre>< xs:simpleType name = "Boolean" > < xs:restriction base = "xs:boolean"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: — for mandatory booleans <pre>< xs:element name = "UMLAttributeName" type = "tdt:Boolean"/ ></pre> — for multiple mandatory booleans <pre>< xs:element name = "UMLAttributeName" type = "tdt:Boolean" maxOccurs = "unbounded"/ ></pre> — for one single optional boolean <pre>< xs:element name = "UMLAttributeName" type = "tdt:Boolean" minOccurs = "0"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > true < /UMLAttributeName ></pre>
DateTime	A time field in UTC. It is based on the ISO 8601 extended: YYYY-MM-DDThh:mm:ssZ, where: — "YYYY" the four-digit representation of the year — "MM" the two-digit representation of the month [1 .. 12] — "DD" the two-digit representation of the day [1 .. 31] — "T" the date/time separator — "hh" the two-digit representation of the hour [0 .. 24] — "mm" the two-digit representation of the minute [0 .. 59] — "ss" the two-digit representation of the second [0 .. 59] The representation is immediately followed by a "Z" to indicate Coordinated Universal Time (UTC). EXAMPLE message_generation_time = "2001-02-12T12:01:13Z" NOTE Midnight can be represented both as 24:00:00 and as 00:00:00, the former is used at the end of a calendar day and the latter is used at the start of a calendar day. <u>XSD definition:</u> <pre>< xs:simpleType name = "DateTime" > < xs:restriction base = "xs:dateTime"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:DateTime"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 2009-01-01T01:02:03Z < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
DaySelector	<u>XSD definition:</u> <pre>< xs:complexType name = "DaySelector" > < xs:sequence > < xs:element name = "Monday" minOccurs = "0" / > < xs:element name = "Tuesday" minOccurs = "0" / > < xs:element name = "Wednesday" minOccurs = "0" / > < xs:element name = "Thursday" minOccurs = "0" / > < xs:element name = "Friday" minOccurs = "0" / > < xs:element name = "Saturday" minOccurs = "0" / > < xs:element name = "Sunday" minOccurs = "0" / > < /xs:sequence > < /xs:complexType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:DaySelector" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName xmlns:tdt = "http://www.tisa.org/TPEG/TPEGDataTypes_2_1"> < tdt:Monday / > < tdt:Friday / > < /UMLAttributeName ></pre>
DistanceMetres	<u>XSD definition:</u> <pre>< xs:simpleType name = "DistanceMetres" > < xs:restriction base = "xs:unsignedInt" > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:DistanceMetres" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 12345 < /UMLAttributeName ></pre>
DistanceCentiMetres	<u>XSD definition:</u> <pre>< xs:simpleType name = "DistanceCentiMetres" > < xs:restriction base = "xs:unsignedInt" / > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:DistanceCentiMetres" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 12345 < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
Duration	<u>XSD definition:</u> <pre>< xs:simpleType name = "Duration" > < xs:restriction base = "xs:unsignedInt"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:Duration"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 12345 < /UMLAttributeName ></pre>
FixedPercentage	<u>XSD definition:</u> <pre>< xs:simpleType name = "FixedPercentage" > < xs:restriction base = "xs:unsignedByte" > < xs:minInclusive value = "0"/ > < xs:maxInclusive value = "100"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:FixedPercentage"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 99 < /UMLAttributeName ></pre>
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. <u>XSD definition:</u> <pre>< xs:simpleType name = "FixedPointNumber" > < xs:restriction base = "xs:decimal" > < xs:minInclusive value = "-2147483648.99"/ > < xs:maxInclusive value = "2147483648.99"/ > < xs:fractionDigits value = "2"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:FixedPointNumber"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 12.34 < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
Float	<u>XSD definition:</u> <pre>< xs:simpleType name = "Float" > < xs:restriction base = "xs:float" / > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:Float" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 12.34 < /UMLAttributeName ></pre>
IntSiTi	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntSiTi" > < xs:restriction base = "xs:byte" / > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntSiTi" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > -123 < /UMLAttributeName ></pre>
IntSiLi	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntSiLi" > < xs:restriction base = "xs:short" / > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntSiLi" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > -12345 < /UMLAttributeName ></pre>
IntSi24	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntSi24" > < xs:restriction base = "xs:int" > < xs:minInclusive value = "-8388608" / > < xs:maxInclusive value = "8388607" / > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntSi24" / ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > -5432109 < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
IntSiLo	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntSiLo" > < xs:restriction base = "xs:int"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntSiLo"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > -123456789 < /UMLAttributeName ></pre>
IntSiLoMB	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntSiLoMB" > < xs:restriction base = "xs:int"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntSiLoMB"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > -123456789 < /UMLAttributeName ></pre>
IntUnTi	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntUnTi" > < xs:restriction base = "xs:unsignedByte"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntUnTi"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 123 < /UMLAttributeName ></pre>
IntUnLi	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntUnLi" > < xs:restriction base = "xs:unsignedShort"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntUnLi"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 12345 < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
IntUn24	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntUn24" > < xs:restriction base = "xs:unsignedInt" > < xs:maxInclusive value = "16777215"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntUn24"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 16432100 < /UMLAttributeName ></pre>
IntUnLo	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntUnLo" > < xs:restriction base = "xs:unsignedInt"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntUnLo"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 123456789 < /UMLAttributeName ></pre>
IntUnLoMB	<u>XSD definition:</u> <pre>< xs:simpleType name = "IntUnLoMB" > < xs:restriction base = "xs:unsignedInt"/ > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:IntUnLoMB"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 123456789 < /UMLAttributeName ></pre>
ShortString	<u>XSD definition:</u> <pre>< xs:simpleType name = "ShortString" > < xs:restriction base = "xs:string" > < xs:maxLength value = "255"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:ShortString"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > abcdef < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
LongString	<u>XSD definition:</u> <pre>< xs:simpleType name = "LongString" > < xs:restriction base = "xs:string" > < xs:maxLength value = "65535"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:LongString"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > abcdef < /UMLAttributeName ></pre>
LocalisedShortString	<u>XSD definition:</u> <pre>< xs:complexType name = "LocalisedShortString" > < xs:sequence > < xs:element name = "languageCode" type = "typ001_LanguageCode"/ > < xs:element name = "value" type = "ShortString"/ > < /xs:sequence > < /xs:complexType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:LocalizedShortString"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName xmlns:tdt = "http://www.tisa.org/TPEG/TPEGDataTypes_2_1"> < tdt:languageCode tdt:table = "typ001_LanguageCode" tdt:code = "33" / > < tdt:value > abcdef < /tdt:value > < /UMLAttributeName ></pre>
LocalisedLongString	<u>XSD definition:</u> <pre>< xs:complexType name = "LocalisedLongString" > < xs:sequence > < xs:element name = "languageCode" type = "typ001_LanguageCode"/ > < xs:element name = "value" type = "LongString"/ > < /xs:sequence > < /xs:complexType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:LocalizedLongString"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName xmlns:tdt = "http://www.tisa.org/TPEG/TPEGDataTypes_2_1"> < tdt:languageCode tdt:table = "typ001_LanguageCode" tdt:code = "33" / > < tdt:value > abcdef < /tdt:value > < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
MajorMinorVersion	The constituting elements MajorVersion and MinorVersion are each encoded as a string with value "[0-9]" or "1[0-5]" for a total of 16 values The MajorMinorVersion is then encoded as the concatenation of the MajorVersion string and the MinorVersion string with a "." separator character in between Example: MajorVersion is "3" and MinorVersion is "15": MajorMinorVersion is encoded as "3.15" <u>XSD definition:</u> <pre>< xs:simpleType name = "MajorMinorVersion" > < xs:restriction base = "xs:string" > < xs:pattern value = "([0-9] 1[0-5])\.([0-9] 1[0-5])"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:MajorMinorVersion"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 3.15 < /UMLAttributeName ></pre>
Probability	<u>XSD definition:</u> <pre>< xs:simpleType name = "Probability" > < xs:restriction base = "xs:decimal" > < xs:minInclusive value = "0.00"/ > < xs:maxInclusive value = "1.00"/ > < xs:fractionDigits value = "2"/ > < /xs:restriction > < /xs:simpleType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:Probability"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > 0.12 < /UMLAttributeName ></pre>

Data type	XML format definition (XML element type)
ServiceIdentifier	<u>XSD definition:</u> <pre>< xs:complexType name = "ServiceIdentifier" > < xs:sequence > < xs:element name = "SID_A" type = "xs:unsignedByte"/ > < xs:element name = "SID_B" type = "xs:unsignedByte"/ > < xs:element name = "SID_C" type = "xs:unsignedByte"/ > < /xs:sequence > < /xs:complexType ></pre> To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:ServiceIdentifier"/ ></pre> where <i>tdt</i> is the prefix the of "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName xmlns:tdt = "http://www.tisa.org/TPEG/TPEGDataTypes_2_1"> < tdt:SID_A > 1 < /tdt:SID_A > < tdt:SID_B > 2 < /tdt:SID_B > < tdt:SID_C > 3 < /tdt:SID_C > < /UMLAttributeName ></pre>
TimeInterval	<u>XSD definition:</u> <pre>< xs:simpleType name = "TimeInterval" > < xs:restriction base = "xs:duration"/ > < /xs:simpleType ></pre> The XSD type duration has the following syntax PnYnMnDnHnMnS, where: — P start character — nY represents the number of years — nM the number of months — nD the number of days — T is the date/time separator — nH the number of hours — nM the number of minites — nS number of seconds EXAMPLE A time interval of 10 years, 3 months, 21 days, 3 h, 43 min and 2 s is represented by: P10Y3M21DT3H43M2S. To use an UML attribute of this type, the application tpegML schema shall have the following definition: <pre>< xs:element name = "UMLAttributeName" type = "tdt:TimeInterval"/ ></pre> where <i>tdt</i> is the prefix of the "TPEG Data Types" namespace. <u>XML example:</u> <pre>< UMLAttributeName > P10Y3M21DT3H43M2S < /UMLAttributeName ></pre>