INTERNATIONAL STANDARD

ISO 24531

Second edition
2013-06-01

Intelligent transport systems — System architecture, taxonomy and terminology — Using XML in ITS standards, data registries and data dictionaries

Systèmes intelligents de transport — Architecture, taxinomie et terminologie des systèmes — Usage de XML dans les normes, registres de données et dictionnaires de données, en ITS

Reference number
ISO 24531:2013(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web 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. 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. 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.

The committee responsible for this document is ISO/TC 204, *Intelligent Transport Systems*.

This second edition cancels and replaces the first edition (ISO 24531:2006). Clause 7 onwards has been technically revised.

Introduction

As the exchange of information via the internet and other wired and wire-free networks develops and expands, the use of XML (Extended Mark-up Language) and its variants continues to grow and develop.

XML will be an important tool in the development and operation of “Intelligent Transport Systems” (ITS) services.

However, within XML and its variants there are options. In order to obtain maximum benefit, interoperability and re-use of data within the ITS sector, it is important to implement XML and its variants in a consistent manner.

This International Standard provides definitions of how to use XML and its variants in a consistent and interoperable manner within the ITS sector.

Intelligent transport systems — System architecture, taxonomy and terminology — Using XML in ITS standards, data registries and data dictionaries

1 Scope

This International Standard assists ITS standards developers and users of ITS standards who wish to use XML, by providing a consistent definition of the rules and rule references for the use of XML within ITS systems. This International Standard defines consistent rules and rule references to provide a framework to be used when implementing XML-based applications in ITS, and particularly in specifying XML in ITS standards, ITS data registries and ITS data dictionaries. This International Standard also provides guidance and examples in respect of the use of XML in ITS, and the elaboration of XML within the ASN.1 data definitions required by ISO 14813-6 and ISO 14817.

NOTE A table of language comparisons (XML, ASN.1, UML) can be found in ISO 14813-6:2009.

2 Conformance

This International Standard prescribes a conceptual model; it does not define any single physical implementation. It provides a consistent and interoperable means of achieving interoperability for the international exchange of information in XML application programs. Regional and national XML schema have the option of providing additional schema and variants for use in local situations.

In order to claim conformance with this International Standard, it is only required to design systems and exchange data consistently in accordance with the provisions of this International Standard. No external conformance procedures are proposed or defined in this International Standard, although regional, national and local implementations are free to, and may choose to, define and require local conformance procedures.

3 Normative references

The following referenced documents are indispensable for the application 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 14812:1999, *Transport information and control systems — Glossary standard terminologies for the transport information and control sector*

ISO 14817:2002, *Transport information and control systems — Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries*

ISO/IEC 19501:2005, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*

W3C Recommendation, *Extensible Mark-up Language (XML) 1.0* (Fifth Edition), 26 November 2008

W3C Recommendation, *Namespaces in XML 1.0* (Second Edition), 16 August 2006

W3C Recommendation, *XML Schema Part 1: Structures* (Second Edition), 28 October 2004

W3C Recommendation, *XML Schema Part 2: Datatypes* (Second Edition), 28 October, 2004

W3C Recommendation, *XML Linking Language (XLink)*, Version 1.0, 27 June 2001

W3C Recommendation, *XSL Transformations (XSLT)*, Version 2.0, 23 January 2007

OASIS, C*ode List Representation (Genericode)*, Version 1.0, December 2007

OASIS, *Context/value association using genericode 1.0*, April 2010

ISOC, RFC 5141, *A Uniform Resource Name (URN) Namespeace for the International Organization for Standardization (ISO)*, March 2008

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 14812 and the following apply.

4.1
application
<XML> program that reads XML documents and "does something useful" with them

Note 1 to entry: Applications will normally be interfaced to an XML parser, for example via DOM or SAX.

4.2
ASN.1 application
application that uses ASN.1 encodings for communication (except XER)

4.3
ASN.1 schema
definition of the content and structure of data using an ASN.1 type definition

4.4
association end
<UML> endpoint of an association, which connects the association to a classifier

4.5
attribute
<XML> property of an element

Note 1 to entry: It is additional information about a piece of data (element). Often attributes are used to pass information about the element and hence can be said to provide metadata for the element. An attribute is a value indicator (=) and the attribute value is specified within the tag (i.e. <H3 align="centre">). Attribute in XML is a name="value" pair that can be placed in the start tag of an element. For XML, all values have to be quoted with single or double quotes.

4.6
attribute
<UML> feature within a classifier that describes a range of values those instances of the classifier may hold

4.7
child element
<XML> element contained within another element

Note 1 to entry: The element containing other elements is a parent element.

4.8
class
<UML> description of a set of objects that share the same attributes, operations, methods, relationships, and semantics

4.9
class diagram
<UML> diagram that shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships

4.10
constraint
<UML> semantic condition or restriction

Note 1 to entry: Certain constraints are predefined in the UML, others may be user defined. Constraints are one of three extensibility mechanisms in UML.

4.11
content
<XML> all data between the start tag and end tag of an element

Note 1 to entry: Content may be made up of mark-up characters and character data.

4.12
content model
<XML> expression specifying what elements and data are allowed within an element

4.13
data concept
any of a group of data dictionary structures defined in ISO 14817 (i.e. object class, property, value domain, data element concept, data element, data frame, message, interface dialogue, association) referring to abstractions or things in the natural world that can be identified with explicit boundaries and meaning and whose properties and behaviour all follow the same rules

[ISO 14817:2002, definition 4.4]

4.14
Data Dictionary
organized and constructed (electronic data base) compilation of descriptions of data concepts that provides a consistent means for documenting, storing and retrieving the syntactical form (i.e. representational form) and the meaning and connotation of each data concept

[ISO 14817:2002, definition 4.6]

4.15
data element
data concept; some single unit of information of interest (such as a fact, proposition, observation, etc.) about some (entity) class of interest (e.g. a person, place, process, property, concept, association, state, event)

[ISO 14817:2002, definition 4.7]

Note 1 to entry: A data element is considered to be indivisible in a particular context.

4.16
data frame
data concept; grouping of data elements primarily for the purpose of referring to the group with a single name, and thereby efficiently reusing groups of data elements that commonly appears together (as an ASN.1 SEQUENCE, SEQUENCE OF, SET, SET OF or CHOICE) in a message specification

4.17
data registry
store of data, characterized in a consistent manner, as determined according to the provisions of this International Standard, used for a specific purpose (in this case ITS)

[ISO 14817:2002, definition 4.11]

4.18
data type
<XML, UML> type of content that an element contains in XML and UML

Note 1 to entry: An author can specify an element's data type.

4.19
declaration
<XML> create new types (both simple and complex)

4.20
definition
<XML> enable elements and attributes with specific names and types (both simple and complex) to appear in document instances

4.21
document type definition
<XML> rules that define the tags that can be used in an XML file and their valid values

4.22
element
<XML> logical data structure within an XML document, a piece of data within a file

Note 1 to entry: An XML element consists of a start tag, and end tag, and the information between the tags, which is often referred to as the contents. Start tags and end tags show the beginning and end of an element. A schema that can provide a description of the structure of the data describes elements used in an XML file.

4.23
element
<UML> atomic constituent of the UML model

4.24
end tag
<XML> element delimiter

Note 1 to entry: In: <foo>this is a bar</foo> the construct </foo> is the end-tag. End tags cannot include anything other than the element name and trailing space.

4.25
global
construct (e.g. element, group, attribute, attribute group, or data type) that is declared as a direct child of the schema root element

4.26
internet (uniform) resource identifier
IRI
compact string of characters for identifying an abstract or physical resource

4.27
lexical space
<XML> set of valid literals for a data type

4.28
local
<XML> element, group, attribute, attribute group, or data types that are not global

4.29
mark-up
<XML> identification of element types and structure within a document

Note 1 to entry: The mark-up is not actually part of the content, but identifies the components and their roles.

4.31
message
data concept; grouping of data elements and/or data frames as well as associated message metadata, that is used to convey a complete unit of information

[ISO 14817:2002, definition 4.19]

4.32
metadata
data that defines and describes other data

4.33
namespace
<XML> set of unique identifiers

Note 1 to entry: Namespace is a mechanism to resolve naming conflicts between elements in an XML document when each comes from a different vocabulary. It allows the commingling of like tag names from different namespaces. A namespace identifies an XML vocabulary defined within a URN. An attribute on an element, attribute, or entity reference associates a short name with the URN that defines the namespace; that short name is then used as a prefix to the element, attribute, or entity reference name to uniquely identify the namespace. Namespace references have scope. All child nodes beneath the node that specifies the namespace inherit that namespace. This allows nonqualified names to use the default namespace.

4.34
namespace
<UML> part of the model in which the names may be defined and used

Note 1 to entry: Within a namespace, each name has a unique meaning.

4.35
node
<XML> elements, comments, processing instructions, and text in an XML document

Note 1 to entry: An XML document has a hierarchical structure, described as a tree. The tree has branches connecting at the nodes.

4.36
object class
data concept; construct used to represent any kind of object (also referred to as an entity) within an ITS/TICS information environment

[ISO 14817:2002, definition 4.25]

4.37
OID
<ASN.1> globally unique value associated with an object to identify it unambiguously

4.38
package
<UML> general purpose mechanism for organizing elements into groups

Note 1 to entry: Packages may be nested within other packages.

4.39
parser (for XML)
<XML> processor that reads an XML document and determines the structure and properties of the data

Note 1 to entry: If the parser goes beyond the XML rules for conformance and validates the document against an XML schema, the parser is said to be a "validating" parser. A generalized XML parser reads XML files and generates a hierarchically structured tree, then hands off data to viewers and other applications for processing. A validating XML parser also checks the XML syntax and reports errors.

4.40
prefix
namespace prefix
<XML> short name to uniquely identify the namespace

4.41
profile
<UML> stereotyped package that contains model elements, which have been customized for a specific domain or purpose using extension mechanisms, such as stereotypes, tagged definitions and constraints

Note 1 to entry: A profile may also specify model libraries on which it depends and the metamodel subset that it extends.

4.42
property
<UML> named value denoting a characteristic of an element

Note 1 to entry: Certain properties are predefined in the UML; others MAY be user defined. See: tagged value.

Note 2 to entry: A property has semantic impact.

4.43
role
<UML> named specific behaviour of an entity participating in a particular context

4.44
schema
<XML, UML> system of representing an information model that defines the data's elements and attributes

4.45
schema processor
<UML> processor to validate schema

4.46
stereotype
<UML> new type of modelling element that extends the semantics of the metamodel

Note 1 to entry: Stereotypes have to be based on certain existing types or classes in the metamodel. Stereotypes may extend the semantics, but not the structure of pre-existing types and classes. Certain stereotypes are predefined in the UML, others may be user defined. Stereotypes are one of three extensibility mechanisms in UML.

4.47
tags
<XML> text structures that mark-up characters which mark the beginning and end of elements within the XML document

4.48
tagged value
<UML> explicit definition of a property as a name-value pair

Note 1 to entry: Certain tags are predefined in the UML; others MAY be user defined. Tagged values are one of three extensibility mechanisms in UML.

Note 2 to entry: In a tagged value, the name is referred as the tag.

4.49
value domain
data concept; expression of a specific and explicit representation of some information about something of interest within the ITS/TICS domain

[ISO 14817:2002, definition 4.29]

4.50
XMI
XML-based model interchange format for UML models

4.51
XML application
application that uses XML encoding

4.52
XML OID
XML representation of an ASN.1 OID

EXAMPLE In the following example, the ASN.1 OID delimiter (white space) changed by a designated delimiter. ASN.1 OID : iso standard 24531 schema 1; XML OID (delimiter "_"): iso_standard_24531_schema_1; XML OID (delimiter "/"): iso/standard/24531/schema/1

5 Abbreviated terms

ASN.1

abstract syntax notation one

DD

data dictionary

DR

data registry

HTML

hyper text markup language

IRI

Internationalized Resource Identifiers

ISO

International Organization for Standardization

ITS

intelligent transport system(s)

NDR

naming and design rules

OID

object identifier

OMG

object management group

RFC

request for comments

TICS

transport information and control system(s)

UBL

unifversal business language

URL

uniform resource locator

UML

unified modelling language (as defined by ISO 19501)

W3C

world wide web consortium

WSDL

web services description language

XHTML

extensible hyper text mark-up language

XMI

XML metadata interchange

XML

eXtensible mark-up language

6 Document convention

In this International Standard the following documentation conventions are used.

a) The term 'schema' refers specifically to schemas authored in accordance with the World Wide Web Consortium (W3C) XML schema recommendation, unless otherwise indicated.

b) For reasons of brevity, not all examples are full schemas. In all prose and examples, the mappings shown in Table 1 apply, even if no namespace declaration appears in the example.

Table 1 — Namespace prefix and associated namespace

Namespace prefix	Associated Namespace
xs	http://www.w3.org/2001/XMLSchema
xsi	http://www.w3.org/2001/XMLSchema-instance
xmi	http://www.omg.org/2001/XMI

EXAMPLE

```
<xs:simpleType name = "car"/>
```
In this case, it is understood that

```
<xs:schema xlmns:xs = "http://www.w3.org/2001/XMLSchema">
```
has already been declared.

c) Even if no end tag appears in the example, assume the end tag is declared at the appropriate place.

d) All examples are only for the purpose of explanation and are therefore informative. All IRIs in the examples are virtual with the exception of the namespaces listed in Table 1.

e) Throughout this International Standard, in accordance with ISO 31-0:1992, Amd. 2:2005, decimal separators will be a point on the line.

7 Requirements

Figure 1 shows the scope of XML functionality in the ITS sector.

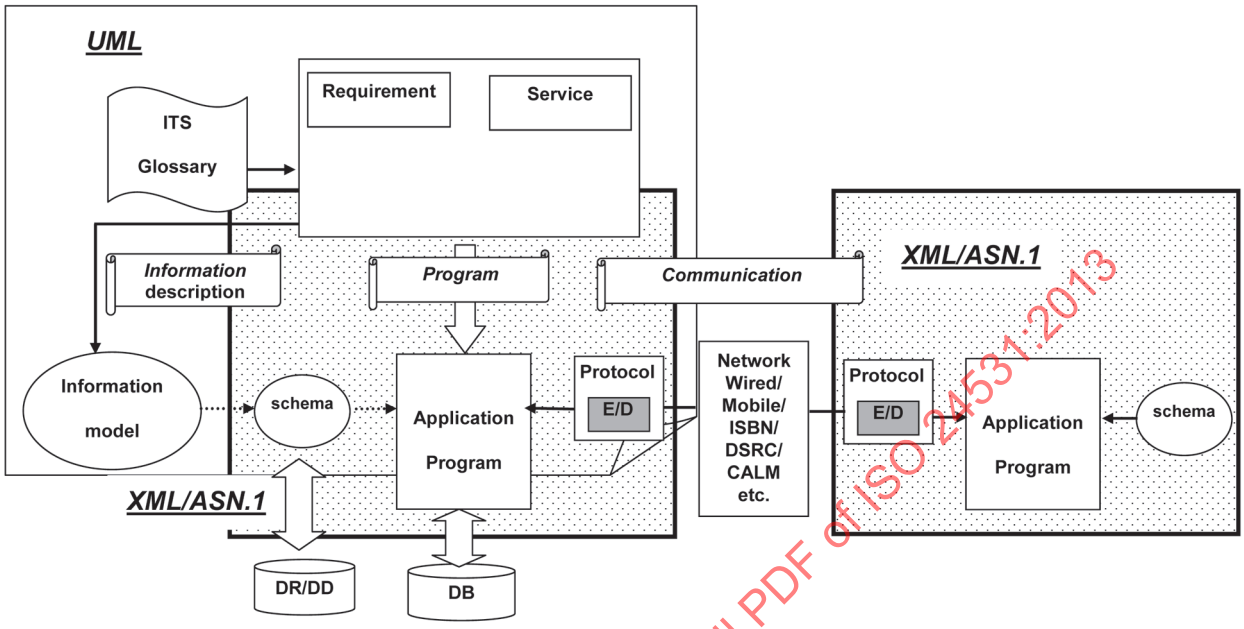


Figure 1 — XML functionality

7.1 Required conditions

Intelligent transport systems are evolving social infrastructure systems that offer various functions. To achieve their benefits, the following exchanges occur:

— information exchanges between various countries' organizations, including web services usages;

— information exchanges between different ITS functional areas;

— information exchanges between ITS-related industries' systems; and

— information exchanges through various networks.

7.2 Required items

From the viewpoint of ITS information technology, the following items are required:

— formal method to define precise and unambiguous ITS vocabularies;

— registration and management rules for XML components, management and maintenance rules (ITS data registries and ITS data dictionaries);

— formal method to define dialogues and messages;

— expandable and reusable vocabularies and programs;

— ways to support various networks (wired/mobile/DSRC/digital broadcast /CALM, etc.);

— efficient encoding method; and

— automatic generation of XML schemas from UML.

7.3 Rules for modelling data exchanges

NOTE Annex C contains a complete example of a modeled data exchange.

7.3.1 Model each data exchange

Every data exchange shall be modeled using a UML sequence diagram. See Figure 2 for an example.

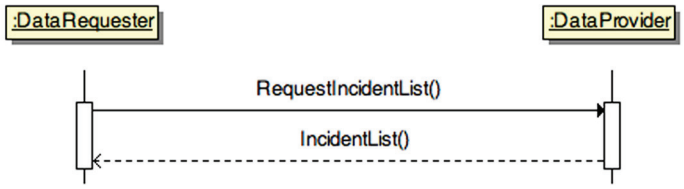


Figure 2 — Sample sequence diagram

NOTE The production of sequence diagrams also facilitates the generation of computer code in various languages and with reverse engineering tools, code/diagram synchronization becomes possible.

7.3.1.1 Identify the entities exchanging messages

The UML sequence diagram shall identify the sender and recipient of each message.

7.3.1.2 Identify the messages exchanged

The UML sequence diagram shall identify the name of each message exchanged in the sequence without the colon or "message" term.

EXAMPLE For a message with the Descriptive Name "IncidentList():message", the name of the UML message in the sequence diagram would be "IncidentList()".

7.3.2 Model each message

Every message shall be modeled using a UML class diagram. Figure 3 shows an example.

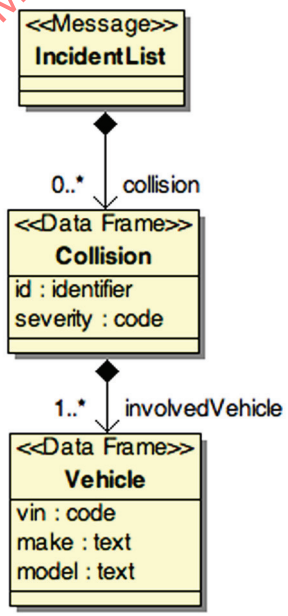


Figure 3 — Sample UML class diagram depicting a message

7.3.2.1 Use the <<Message>> stereotype

A message shall be modeled as a UML class with the stereotype <<Message>>.

7.3.2.2 Specialize from the Message class

A message shall be defined to be a specialization of the generic Message class. The Message class shall be as defined in Annex B.

7.3.2.3 Message name

The name of the UML class representing the message shall be the Descriptive Name of the message without the parenthesis, colon, or "message" term.

EXAMPLE For a message with the Descriptive Name "IncidentList():message", the UML class name would be "IncidentList".

7.3.2.4 Message contents

All message contents shall be modeled as either properties or associated data frames.

7.3.3 Modeling properties

If a property of an object class is relevant to a message, it shall be modeled as a UML attribute of its parent data frame or message.

NOTE In other words, a message (or data frame) is really just a view of an object class. The object class may have additional properties and associations not included in the message (or data frame) definition, but the message (or data frame) may not include any properties or associations not defined for the parent object class.

Figure 4 provides an example.

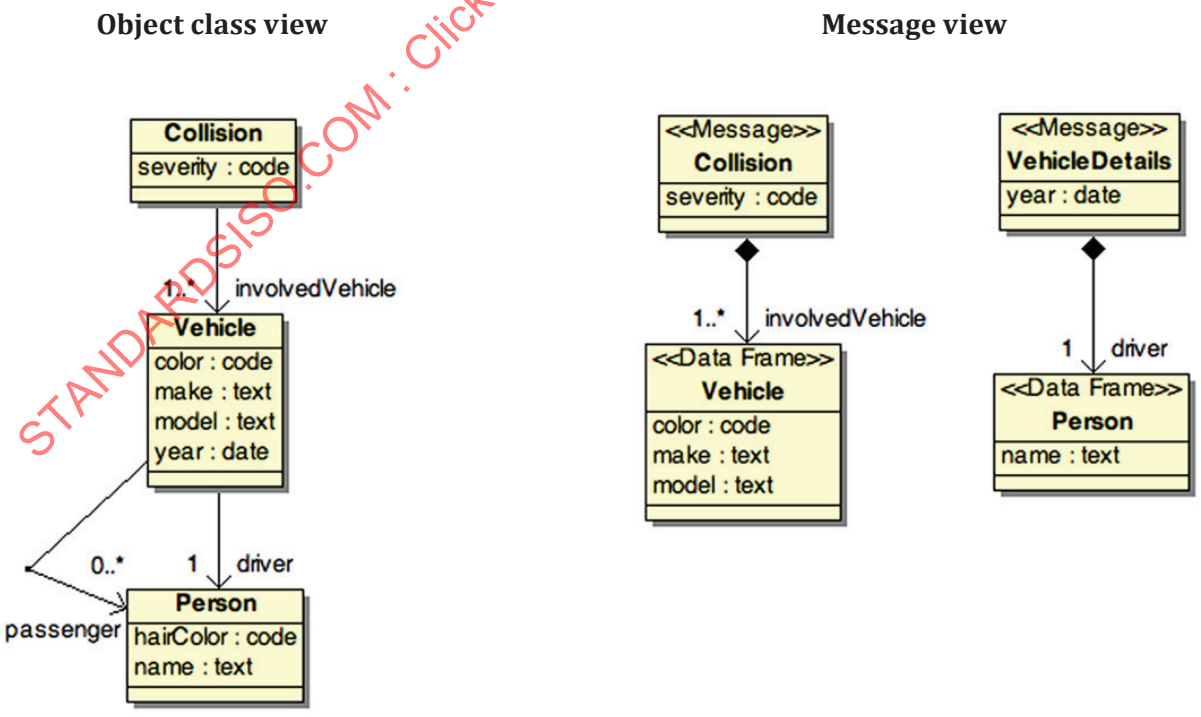


Figure 4 — Object class view vs. message view

7.3.4 Modeling associations

If an association of an object class is relevant to a message, it shall be modeled as a UML directional composition to a UML class with the stereotype <<DataFrame>>.

NOTE In other words, for the purposes of messaging, the message is comprised of the smaller data frames and if a message is lost, so are all of the data frames that it contains. However, the receiving system may loosen the association between the message and data frames such that when the parent entity is deleted from the system, the detailed data may be retained, if desired.

7.3.4.1 Contents of UML classes

A data frame or message shall only contain properties and associations from a single object class.

NOTE In other words, in the above example, Person.name:text is placed in its own data frame even though that data frame has only one attribute defined. This is because a parallel structure to the base data model is being created. Alternatively, the designer could modify the base data model to create a new property (Vehicle. driverName:text) and then place this new property as an attribute of the VehicleDetails message.

7.3.4.2 Role of associated data frame

The association role assigned to the associated UML class shall be the role of the ITS data dictionary/registry association.

NOTE This is often the name of the dataframe, but may be different. For example, the “Person” dataframe may be assigned a role of “user”, “driver”, “employee”, etc.

7.4 Rules for using XML in ITS standards

Rules for using XML and its variants in ITS standards are described in the subclauses that follow.

7.4.1 Rules for XML file organization

In order to promote reusability and code management, schemas shall be constructed of smaller reusable packages wherever possible. The packages shall follow the modular structure as defined by the following subclauses and depicted in Figure 5. Details about each file shown in this schema are provided in 7.4.2 through 7.4.16.

NOTE 1 The file organization and the format of each file is based on UBL Naming and Design Rules (NDR) 2.1 with slight variances to accommodate the ITS domain and management structures. A comparison between UBL NDR and this International Standard is provided in Annex W.

NOTE 2 Electronic versions of the files cited in the following clauses may be accessed at http://standards.iso.org/iso/24531/

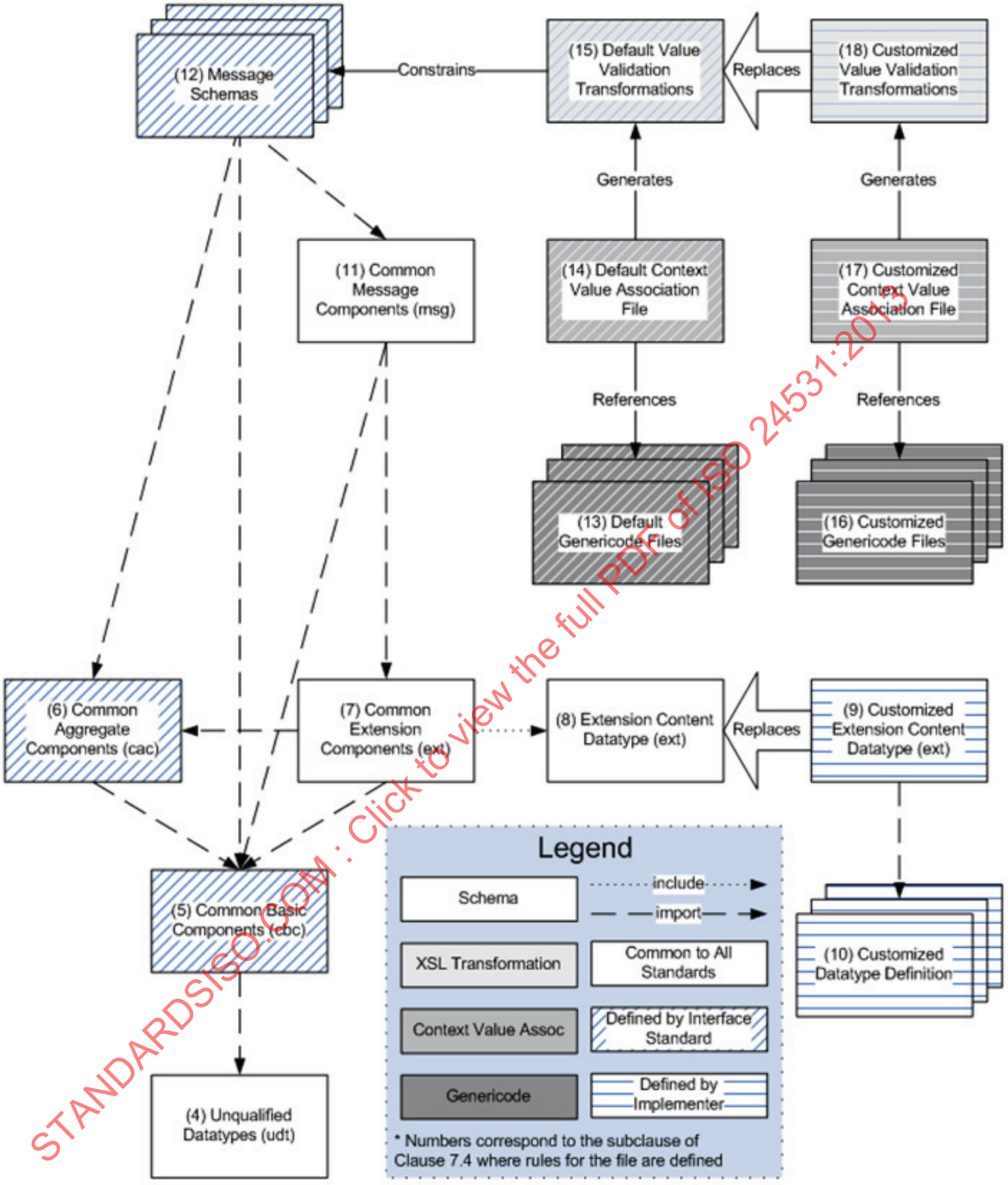


Figure 5 — Relationship among XML interface definition files

7.4.1.1 Unqualified data types schema

The unqualified data types schema provides the definition of each fundamental data type that can be used in ISO/TC 204 standards. All ISO/TC 204 message sets use the same unqualified data types schema, as defined in Annex D.

7.4.1.2 Common basic components schema

The common basic components schema provides the definition of all properties used by a particular set of messages. All ISO/TC 204 message sets shall be based on the same common aggregate components schema, as defined in Annex E. However, it is recognized that there may be a large number of properties within the full scope of ISO/TC 204. In order to accommodate manageable, modular deployments, individual working groups may modify the common aggregate components schema, but only to "include" additional aggregate components schema files.

7.4.1.3 Common aggregate components schema

The common aggregate components schema provides the definition of all data frames, associations, and data elements used by a particular set of messages. All ISO/TC 204 message sets shall use the same common aggregate components schema, as defined in Annex F. However, it is recognized that there may be a large number of data frames, associations, and data elements within the full scope of ISO/TC 204. In order to accommodate manageable, modular deployments, individual working groups may modify the common aggregate components schema, but only to "include" additional aggregate components schema files.

7.4.1.4 Common extension components schema

The common extension components schema provides the definition of how to create an extension to a standardized message. It defines the structure that contains the extension along with a variety of metadata, including the version of the extension, the agency who defined the extension, etc. All ISO/TC 204 message sets shall use the same default extension content data type schema, as defined in Annex G.

7.4.1.5 Extension content data type schema

The extension content data type schema is a standardized schema that serves as a placeholder to allow for extensions to the standard. All ISO/TC 204 message sets shall use the same default extension content data type schema, as defined in Annex H.

7.4.1.6 Customized extension content data type schema

The customized extension content data type schema is a schema produced by an implementer who wishes to extend the standard message set by referencing other schemas. An example of such a schema is provided in Annex S.

7.4.1.7 Customized data type definitions

The customized data type definitions schemas are the set of schemas that the implementer wishes to use to extend the standardized message set. An example of such a schema is provided in Annex T.

7.4.1.8 Common message component schema

The common message components schema provides the definition of the message structure that should be used by all ISO/TC 204 messages. It defines a structure that supports extensions and identifies each message with a version id. All ISO/TC 204 message sets shall use the same common message components schema, as defined in Annex I.

7.4.1.9 Message schemas

A message schema provides the definition of a single message by referencing elements previously defined in the common message components schema, the common aggregate components schema, and the common basic components schema. This file is unique to each message type. Examples for such a schema are provided in Annex J and Annex K.

7.4.1.10 Code list definition (genericode files)

Each code list shall be defined in its own genericode file, as per the rules defined by OASIS. Code lists shall be defined by the messaging standard with a clear indication whether support for each defined value is mandatory or optional. In the latter case, the code list may be customized by an implementation to define the actual code list supported. Example genericode files are provided in Annex L.

7.4.1.11 Context-value associations

The valid codes for a given context shall be defined in a context-value association file, per the rules defined by OASIS. The context-value association file, and any portion thereof, shall be defined by the messaging standard with a clear indication whether each rule is mandatory or advisory. In the latter case, the context-value associations may be customized by an implementation to define the actual values supported. An example context-value association file is provided in Annex M.

7.4.1.12 Value validation transformation files

A value validation transformation file is an EXtensible Stylesheet Language Transformation (XSLT) file produced by performing a standard XSLT transformation on the context-value association file (and referenced genericode files). The result is an XSLT file that can verify that the values contained in an XML document (i.e., an actual instance of a ITS message) are valid. Annex N provides a sample XSLT transformation file that can be used to transform a CVA file into the XSLT format. Annex O provides an example value validation transformation file produced by applying the Annex N stylesheet to the CVA file defined in Annex M.

7.4.1.13 Circular dependencies

There shall be a strictly linear dependency graph between sub schemas. Circular dependencies are not allowed among schema files.

7.4.2 General rules for XML schema

7.4.2.1 Use of the W3C XML schema language

The W3C XML schema language, as defined by the W3C Recommendations listed as normative references to this International Standard, shall be used as the descriptive language for schemas when using XML for International Standards.

NOTE Although there are other descriptive languages for schemas, such as RELAX NG, use of the W3C XML schema language is overwhelmingly predominant in applications written in XML. The W3C XML schema language is used to enable interoperability with other applications.

7.4.2.2 Availability of schemas

All referenced schema files shall be accessible while an application using a specific message schema is in use.

NOTE When an instance document is input, the schema syntax is designed such that validation is performed regardless of whether the processor references the schema or not. Ordinarily, however, a processor references the schema in performing the validation process, and it is thought that such a method shall normally be provided. The basis for doing that is the schema files. Accordingly, the schema file shall be accessible so long as the application is in use.

7.4.2.3 Naming uniqueness

Within any namespace, a name shall be used only once and shall not be repeated even if it is a different type of data.

NOTE Additional naming rules are defined in this International Standard that further specify how to produce data type names that are different from element names and that ensure that attributes are only defined in the unqualified data type schema.

7.4.2.4 Rules for schema types

7.4.2.4.1 Any

The *xsd:any* type shall not be used, except in the construct of the ExtensionComponentDataType.

7.4.2.4.2 Any Type

The *xsd:anyType* type shall not be used.

7.4.2.5 Rules for use of elements within schema definitions

7.4.2.5.1 All

The *xsd:all* element shall not be used.

7.4.2.5.2 App Info

The *xsd:appInfo* element should be avoided and used only to express non-normative information.

7.4.2.5.3 Choice

The *xsd:choice* element should not be used where customization and/or extensibility may be needed.

NOTE The list of options in a required choice statement cannot be extended without breaking backwards compatibility. For example, there may be a variety of ways to describe a location, and the number of ways may increase over time. Rather than using a choice element, it would be better to define one required location format and then allow a series of optional location formats to provide for alternative methods. That approach maximizes interoperability among systems. However, a choice element may be appropriate where the list of options is not expected to ever grow.

7.4.2.5.4 Import

The *xsd:import* element shall only be used to import a schema from a different namespace.

7.4.2.5.5 Include

The *xsd:include* element shall only be used to include a schema in the same namespace.

7.4.2.5.6 Notation

The *xsd:notation* element shall not be used.

7.4.2.5.7 Redefine

The *xsd:redefine* element shall not be used.

NOTE This is to avoid pervasive side-effects in reused components, and to increase clarity and readability.

7.4.2.5.8 Union

The *xsd:union* element shall not be used.

7.4.2.6 Rules for use of attributes within schema definitions

7.4.2.6.1 Final

The “final” attribute shall not be used, except in extensions where there is a desire to prohibit future extensions.

7.4.2.6.2 Mixed

The “mixed” attribute shall not be used. Mixed content is not allowed.

7.4.2.6.3 Nillable

The “nillable” attribute shall not be used.

7.4.2.6.4 Substitution Group

The “substitutionGroup” attribute shall not be used.

7.4.2.7 Attributes

Attributes shall only be defined within the unqualified data types schema to define the base data types used to represent all data.

7.4.2.8 No empty elements

Empty elements shall not be declared.

7.4.2.9 Range checking

ISO/TC 204 standards shall not define normative ranges for values within a schema.

NOTE The mechanisms within XML that define valid data ranges do not provide the necessary flexibility to allow for proper customization, localization, extension, and version control. Instead, this International Standard recommends defining valid values through a separate approach using context-value association files.

7.4.2.10 Documentation

Each schema shall be provided in two forms: an annotated version to assist in human consumption and a version without any annotation for run-time use. The annotated version shall contain annotation elements and XML comments as discussed within this International Standard and as otherwise deemed valuable to the human understanding of the schema contents; the non-annotated version shall not contain any annotation elements or comments.

EXAMPLE

```
<xsd:complexType name=”AreaLocationType”>
    <xsd:annotation>
        <xsd:documentation>
            <Definition>An area on the face of the earth.</Definition>
            <DataConceptType>Data Frame</DataConceptType>
        </xsd:documentation>
    </xsd:annotation>
```

7.4.2.10.1 Annotating elements and types

The documentation element of the annotation element shall be used to record relevant text that relates to a specific element or type.

NOTE The "documentation" element contains a human readable annotation that can be further refined by using the "source" attribute and/or the "*xml:lang*" attribute to support multiple natural language descriptions.

7.4.2.10.2 XML comments

XML comments should only be used to provide supplemental information that does not relate to any specific element or type. For example, XML comments may be used to provide visual separators between different major portions of the schema.

7.4.2.10.3 Annotating messages and data frames

Each complexType representing a message or a data frame shall include an annotation element that documents the following ISO 14817 meta-attributes, as a minimum:

— Definition

— Data concept type

7.4.2.10.4 Annotating properties and associations

Each element contained within a message or data frame complexType that represents a property or association shall include an annotation element that documents the following ISO 14817 meta-attributes, as a minimum, when defined:

— Definition

— Data concept type

— Format, for properties

— Unit of measure, for properties

— Valid value rule, for properties

7.4.3 Layout of XML schema

Every schema shall present information in the following order:

— XML Declaration

— Schema element

— Header

— Imports

— Includes

— Aggregate element declarations

— Aggregate type definitions

— Basic element declarations

— Type definitions

— Copyright statement

7.4.3.1 XML declaration statement

XML schemas shall include the XML declaration statement, complete with the XML version number attribute and encoding attribute explicitly indicated.

EXAMPLE

```
<?xml version = '1.0' encoding = 'UTF-8' ?>
```

7.4.3.2 Schema element

The schema element shall follow the following layout:

— Namespace identifier attributes

— Target namespace attribute

— Element form default attribute

— Attribute form default attribute

— ID attribute (optional)

— Version attribute

EXAMPLE

```
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="urn:iso:std:iso:24531:tech:CommonComponents-1"
    targetNamespace="urn:iso:std:iso:24531:tech:CommonComponents-1"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    version="1.1">
```

7.4.3.2.1 Declare all namespaces

The schema element shall explicitly declare a namespace identifier for every namespace used in the schema.

7.4.3.2.1.1 The "xs" namespace

The schema element shall explicitly declare the schema namespace with the namespace identifier set to "xs". The schema namespace shall be the first namespace defined in the schema element.

EXAMPLE

```
xlmns:xs="http://www.w3c.org/2001/XMLSchema"
```

7.4.3.2.2 Target namespace attribute

The schema element shall explicitly include the target namespace attribute.

7.4.3.2.2.1 Target namespace format

The target namespace for ISO standard ITS schema shall be a URN that conforms to RFC 5141, with the following restrictions:

— The optional status, edition, docversion, and language elements will be omitted

— The optional techdefined addition field is required per this International Standard and shall indicate the name of the schema followed by a hyphen and the major version number.

7.4.3.2.2.2 Target namespace versioning

The value assigned to the target namespace shall include the major version number of the schema as defined in the version attribute of the schema element.

NOTE 1 The value assigned to the target namespace will only change when a non-backward compatible version is released.

NOTE 2 The namespace system facilitates use of the same names in many different fields without any naming collisions. The use of XML in various ITS fields may lead to the same names being used even though they have different meanings. The *targetNamespace* shall therefore be explicitly defined to facilitate the use of names without any collisions.

EXAMPLE targetNamespace="urn:iso:std:iso:24531:tech:CommonComponents-1"

NOTE 3 This example shows an ISO URN (i.e., "urn:iso") that is an international standard recognized by ISO (i.e., not jointly developed) with the number 24531 (i.e. "std:iso:24531") and that it is a draft of a major version 1 release.

7.4.3.2.3 Element form default attribute

The schema element shall explicitly include the "elementFormDefault" attribute.

7.4.3.2.4 Attribute form default attribute

The schema element shall explicitly include the "attributeFormDefault" attribute.

7.4.3.2.5 ID attribute

The schema element may include the id attribute to indicate the identity of the schema. If present, the id attribute shall be the OID of the major version of the subject schema using the XML OID, which uses a delimiter of "_" rather than ".". The OID of the major version of the subject schema shall be defined as follows:

"iso" . "standard" . <standard number> . "schema" . <schema name> . <major version>

EXAMPLE

```
id="iso_standard_24531_schema_common_components_1"
```

7.4.3.2.6 Schema version attribute

The schema element shall explicitly include the version attribute.

NOTE 1 It is highly likely that applications that use XML will be revised, including extension and deletion of elements. In such cases, the schema is generally revised. It is necessary to indicate clearly which version of a schema is currently being used.

NOTE 2 The major and minor version numbers of a schema do not necessarily correspond to either the ISO edition or the ISO docversion. The version numbering of the schema is solely dependent upon the types of changes made to the schema and whether these changes are backwards compatible with previous versions and revisions.

7.4.3.2.6.1 Schema versioning rule

Released schemas shall provide a version number in the form n.m (e.g. 1.2), and drafts also shall provide a version number of the form n.ma (e.g. 1.2b).

7.4.3.2.6.2 Major version number rule

The major version number (n) shall be a sequentially assigned, non-zero positive integer that shall be revised upwards when the change from the previous version of schema will cause existing documents to fail to validate or when the new version adds semantic content that must be processed.

EXAMPLE A group may want to add a new optional element to a message so that message instances can be flagged as being test messages. However, this type of information changes the entire semantics of the message and cannot be safely ignored. Therefore, even though this could be treated as a minor version change from a syntactic perspective, it is important to update the major version number for semantic reasons.

7.4.3.2.6.3 Minor version number rule

The minor version number (m) shall be a sequentially assigned, positive integer that shall be revised upwards when the change to the schema will result in all existing documents continuing to validate and all new documents being semantically backwards-compatible when any new elements are stripped from the document.

NOTE Some new documents, which validate against the new schema version, may fail against the old schema version due to the inclusion of newly defined elements. However, if one can create a pre-processor that strips out any newly defined elements without changing the underlying semantics of the message, the change can still be considered a minor version change.

The minor version number shall be reset to zero when the major version number increments.

7.4.3.2.6.4 Version letter rule

The version letter (a) shall be a sequentially assigned English alphabetical character that shall be revised upwards every time a new draft is issued.

The version letter shall be reset to “a” when the major or minor version number increments.

NOTE Indicating the version of a schema is good practice and helps prevent problems caused by people accidentally working with incorrect schema versions. Therefore a common versioning rule and indicating the schema version attribute is required.

EXAMPLE 1 A possible chronological sequence of version numbers:

1.0a

1.0b

1.0

1.1a

1.1

1.2a

1.2

2.0a

2.0

EXAMPLE 2 In the first version of the example schema, a complexType called VersionExample is defined.

Listing 7-1 Version 1.0 document

```
<xs:schema
    xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
    xmlns=”urn:iso:std:iso:24531:tech:CommonComponents-1”
    targetNamespace=”urn:iso:std:iso:24531:tech:CommonComponents-1”
    elementFormDefault=”qualified”
```

```
    attributeFormDefault="unqualified"
    version="1.0">
    <xs:complexType name="VersionExample">
        <xs:sequence>
            <xs:element name='temperature' type='xs:decimal'/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

In the next version, the VersionExample complexType is modified to include an additional element; however, this additional information does not change the meaning of the previously defined data and older systems can safely ignore the added data without any detrimental effects. As a result, the change is considered a minor version change.

Listing 7-2 Version 1.1 document

```
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
    xmlns="urn:iso:std:iso:24531:tech:CommonComponents-1"
    targetNamespace="urn:iso:std:iso:24531:tech:CommonComponents-1"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    version="1.1">
    <xs:complexType name="VersionExample">
        <xs:sequence>
            <xs:element name='temperature' type='xs:decimal'/>
            <xs:element name='humidity' type='xs:decimal'/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

NOTE The 'humidity' element was added to the schema, therefore a new conformant document containing the 'humidity' element would not validate with the version 1.0 schema. So the minor version number is incremented (in this example to 1.1). However, any version 1.0 deployment could readily accept such a message by adding a XSLT transformation pre-processor that removed any data not recognized by the deployment.

In the next version, an additional field is added to allow for the temperature to be provided in different units. This results in a semantic change to the previously defined elements and therefore requires a major version change.

Listing 7-3 Version 2.0 document

```
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
    xmlns="urn:iso:std:iso:24531:tech:CommonComponents-2"
    targetNamespace="urn:iso:std:iso:24531:tech:CommonComponents-2"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified"
    version="2.0">
    <xs:complexType name="VersionExample">
        <xs:sequence>
            <xs:element name='temperature' type='xs:decimal' />
            <xs:element name='humidity' type='xs:decimal' minOccurs="0" />
            <xs:element name='tempUnits' type='xs:text'/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

7.4.3.2.6.5 Always update version

The version attribute shall be updated whenever a change is made to the schema and distributed for review.

7.4.3.3 Schema header

In annotated versions of the schema, the schema element shall contain an annotation element as its first child element. The annotation element shall document information about the schema as defined in Table 2. This annotation element shall be called the header.

Table 2 — Schema header item

Header Item	Description
General information	
Schema name	Described by id attribute
Description	Plain text description of the type of information described by the schema
Current status	(Released/ Draft)
Standard title	Title of the associated standard
Standard number	Number of the associated standard
Validated by	Schema validation software name
Release Package	The URI of the location where the schema set can be downloaded
Change history	
Version number	Release or draft version number
Date	Version number change date
Description of change	Plain text description
Contact	
Name	Name of person to contact with questions regarding the schema
e-mail address	e-mail address to contact with questions regarding the schema

NOTE 1 By adhering a header to the schema, the schema becomes an interoperability tool and promotes reuse of the schema, because analysts can read it and understand the meaning and derivation of various XML components. It also expected to increase the maintainability of the system.

NOTE 2 See samples in annexes.

7.4.3.4 Import element

A schema may include “xs:import” elements.

7.4.3.4.1 Imported namespace

Each “xs:import” element shall include a “namespace” attribute that indicates the namespace of the imported schema.

7.4.3.4.2 Imported schemaLocation

Each “xs:import” element shall include a “schemaLocation” attribute that indicates the relative location of the imported schema in the release package.

EXAMPLE

```
<xsd:import namespace=”urn:iso:std:iso:24531:tech:CommonComponents-1”
   schemaLocation=”24531-CommonComponents-1.0.xsd”/>
```

7.4.3.5 Include element

A schema may include “xs:include” elements.

7.4.3.5.1 Included schemaLocation

Each “xs:include” element shall include a “schemaLocation” attribute that indicates the relative location of the included schema in the release package.

EXAMPLE

```
<xsd:include schemaLocation=”../../ext/ExtensionContentDataType.xsd”/>
```

7.4.3.6 Elements

Schemas shall be designed so that elements are the main holders of information content in the XML instances.

NOTE Attributes are more suited to holding ancillary metadata – simple items providing more information about the element content. Unlike elements, attributes cannot hold structured data. For this reason, elements are preferred as the principal holders of information content. However, attributes are used by the Core Component Types Schema to allow the inclusion of metadata about an element’s content (for example, the format of a date, a unit of measure or the identification of a value set).

7.4.3.7 Complex types

The following data concepts shall be represented by a corresponding complex data type definition:

— Message

— Data frame

7.4.4 Rules for unqualified data types schema

The unqualified data types schema defines the XML data types used as the basis of all ISO 14817 value domain terms and defines appropriate XML attributes for each of these data types.

7.4.4.1 Names of unqualified data types

All unqualified data types shall have names in UpperCamelCase and shall terminate with the term “Type”.

7.4.4.2 Unqualified data type definitions

All ISO/TC 204 standards shall use the unqualified data types schema as defined in Annex D of this International Standard.

7.4.4.3 Filename

The filename of the unqualified data types schema shall be “24531-UnqualifiedDataTypes-2.0.xsd”, where the 2.0 will be periodically updated to reflect the actual version.

7.4.4.4 Namespace identifier

The unqualified data types schema shall be assigned the namespace identifier of “udt” when imported into other schemas.

7.4.5 Rules for common basic components schema

The common basic components schema defines or includes the properties that are used by the subject message set.

7.4.5.1 Basic component definitions

All ISO/TC 204 message sets shall use a common basic components schema that is derived from the one defined in Annex E of this International Standard. Message set standards may only modify this common basic components schema in order to "include" additional basic components schemas.

NOTE It is recognized that there may be a large number of properties within the full scope of ISO TC 204. The intent is to define all generic properties within the common basic components schema, but to allow individual working groups to extend this set to meet their own needs in a modular fashion.

7.4.5.2 Elements representing properties

The common basic components schema for a message set standard shall either directly define or include a definition for a global element for each property included in the message set in order to promote the reuse of data concepts.

NOTE This allows the data concept to be referenced (i.e. "ref") by multiple data elements defined in the aggregate components schemas.

7.4.5.2.1 Name of property element

The name of the property element shall be the name of the property, as defined in the ITS Data Registry, followed by the name of its unqualified data type with the word "Type" removed.

7.4.5.2.2 Special rule for text elements

If the unqualified data type of the property is "TextType", the unqualified data type shall not be added to the end of the element name.

7.4.5.2.3 Special rule for identifier elements

If the unqualified data type of the property is "IdentifierType", the term "ID" shall be added to the end of the element name rather than the full name of the unqualified data type.

7.4.5.2.4 Removal of duplicate terms

When constructing the name of a property element, duplicate terms shall be removed.

EXAMPLE For the property "firstName:name", the XML element name would be firstName, not firstNameName.

7.4.5.2.5 Type of the property element

The type of the property element shall be one of the unqualified data types defined in the unqualified data types schema.

EXAMPLE

```
<xs:element name="colorCode" type="CodeType">
```

7.4.5.3 Filename

The filename of the common basic components schema shall indicate the ISO standard number of the message set, a hyphen, "CommonBasicComponents", another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be ".xsd".

EXAMPLE

```
24531-CommonBasicComponents-2.0a.xsd
```

The filename of an included basic components schema shall indicate the ISO standard number of the message set, a hyphen, an indication of the specialized set of basic components, the string "BasicComponents",

another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be ".xsd".

EXAMPLE

```
24531-ExampleBasicComponents-2.0a.xsd
```

7.4.5.4 Namespace identifier

The common basic components schema shall be assigned the namespace identifier of "cbc" when imported into another schema.

7.4.6 Rules for common aggregate components schema

The common aggregate components schema defines or includes the data frames, associations, and data elements that are used by the subject message set.

7.4.6.1 Aggregate component definitions

All ISO/TC 204 message sets shall use a common aggregate components schema that is derived from the one defined in Annex F of this International Standard. Message set standards may only modify this common aggregate components schema in order to "include" additional aggregate components schemas.

NOTE It is recognized that there may be a large number of data frames within the full scope of ISO/TC 204. The intent is to minimize the size of the common aggregate components schema to those entities common to all ISO/TC 204 efforts, but to allow individual working groups to extend this set to meet their own needs in a modular fashion.

7.4.6.2 Elements representing data frames

The aggregate components schemas shall define a global element for each data frame included in a message set in order to promote the reuse of data concepts.

NOTE This allows the data concept to be referenced (i.e. "ref") by multiple association elements.

7.4.6.2.1 Name of data frame element

The name of the data frame element is the name of the data frame, as defined in the ITS Data Registry, except that the initial character is lowercase.

NOTE This will produce a name that is in lowerCamelCase.

7.4.6.2.2 Type of data frame element

The type of the data frame element is the name of the corresponding complexType as defined in 7.4.6.4.1 with the term "Structure" removed.

EXAMPLE

```
<xs:element name="incident" type="IncidentStructure">
```

7.4.6.3 Elements representing associations

The aggregate components schemas shall define a global element for each association that has a name that is different than its associated data frame.

EXAMPLE If a message was associated with the data frame "location" and the name of the association was "location", there is not a need to define a separate "location" element for the association. However, if the association was named "startLocation", a separate element would be defined for the association and that element would be referenced from the parent message.

7.4.6.3.1 Name of association elements

The name of the association element shall be the role name of the associated object class, as defined in the ITS Data Registry, followed by the name of the associated data frame. The initial character of the role name shall be lower case.

7.4.6.3.2 Removal of duplicate terms within an association name

When constructing the name of an association element, duplicate terms shall be removed.

EXAMPLE For the association "startLocation:Location", the element name would be startLocation, not startLocationLocation.

7.4.6.3.3 Type of association elements

The type of the association element is the name of its associated data frame complexType, as defined in 7.4.6.4.1.

7.4.6.4 Complex types representing data frames

The aggregate components schemas shall define a complexType for each data frame in order to promote reuse of the data concept.

7.4.6.4.1 Name of data frame complex type

The name of the data frame complexType shall be the name of the data frame, as defined in the ITS Data Registry, followed by the term "Structure".

NOTE This will result in the name being in UpperCamelCase with an ending of "Structure".

7.4.6.4.2 Sequence type

The data frame complexType shall be defined as an "xs:sequence" of inherited data frame information (i.e. properties and associations from generalized data frames) along with its own properties and associations.

7.4.6.4.3 Elements within the sequence

Each local element of a sequence shall reference (i.e. "ref") a globally defined element (i.e. either a property or an associated data frame).

7.4.6.4.4 Ordering of data frame members

The sequence of contained elements within a data frame element shall start with the properties and associated data frames from inherited data frames, followed by the properties that are specific to this data frame, followed by the associated data frames that are specific to this data frame.

7.4.6.4.5 Ordering of data element concepts

The ordering of the properties within the sequence shall be identical to the ordering of properties in UML class diagrams.

7.4.6.4.6 Use of choice

The data frame sequence may contain a "choice" in order to model an "XOR" constraint for multiple discretely identified associations or data element concepts. The list of choices shall not change in a minor version release.

EXAMPLE A data frame might allow a location to be represented in any one of multiple forms. In UML, this could be depicted an XOR constraint between two associations; in XML, this XOR constraint would be presented as a "xs:choice" statement.

7.4.6.5 Filename

The filename of the common aggregate components schema shall indicate the ISO standard number of the message set, a hyphen, "CommonAggregateComponents", another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be ".xsd".

EXAMPLE

```
24531-CommonAggregateComponents-2.0a.xsd
```

The filename of an included aggregate components schema shall indicate the ISO standard number where the file is defined, a hyphen, an indication of the specialized set of aggregate components, the string "AggregateComponents", another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be ".xsd".

NOTE It is recommended that a separate aggregate component schema be created for each data frame.

EXAMPLE

```
24531-AgencyAggregateComponents-2.0a.xsd
```

7.4.6.6 Namespace identifier

The common aggregate components schema shall be assigned the namespace identifier of "cac" when imported into another schema.

7.4.7 Rules for common extension components schema

The common extension components schema defines the overall structure to be used when including a customized extension to a standardized message. It defines a placeholder for the extension itself and a set of meta-data about the extension, such as its version. All ISO TC 204 message sets shall use the same default extension content data type schema, as defined in Annex G.

7.4.7.1 Filename

The filename of the common extension components schema shall be "24531-CommonExtensionComponents-2.0.xsd", where the 2.0 will be periodically updated to reflect the actual version.

7.4.7.2 Namespace identifier

The common extension components schema shall be assigned the namespace identifier of "ext" when imported into other schemas.

7.4.8 Rules for extension content data type schema

The extension content data type schema shall define the standard mechanism by which ISO/TC 204 XML messages can be extended to support customized extensions.

7.4.8.1 Extension content data type definition

All ISO/TC 204 standards shall use the extension content data type schema as defined in Annex H of this International Standard.

7.4.8.2 Filename

The filename shall be “ExtensionContentDataType.xsd”.

7.4.8.3 Namespace identifier

The extension content data type schema shall use the namespace identifier of “ext” when included into other schemas.

7.4.9 Rules for customizing the extension content data type schema

A deployment may customize the extension content data type schema by adding import statements to import customized data type definition schemas. An example of a customized extension content data type schema is provided in Annex S.

7.4.9.1 Filename

The filename shall be “ExtensionContentDataType.xsd”.

NOTE This is the same filename assigned to the standard extension content data type schema. By using the same filename, the customized version of the file can overwrite the standard file in the directory structure and the 24531-CommonExtensionComponent-1.0.xsd schema will automatically recognize all of the namespaces defined in the customized file.

7.4.9.2 Namespace identifier

The customized extension content data type schema shall use the namespace identifier of “ext” when included into other schemas.

7.4.9.3 Restrictions on Modifications

The only changes allowed to the standard extension content data type schema is the addition of additional import statements. No other changes shall be allowed.

7.4.10 Rules for defining customized data type definition schemas

Those deploying the standard may develop customized data type definition schemas in order to define elements that can be used to extend the standard messages to support custom and/or local requirements for information exchange. An example of a customized data type definition schema is provided in Annex T.

7.4.10.1 Requirements for customized data type definition schema

A conformant customized data type definition schema shall conform to all of the general rules for XML Schema as defined in 7.4.2.

7.4.10.2 Preferred rules for customized data type definition schema

A customized data type definition schema should conform to the rules for the common component schema and message set components schema.

7.4.11 Rules for common message components schema

The common message components schema defines the overall message structure that should be inherited by every ISO/TC 204 message. All ISO TC 204 standards shall use the common message components schema as defined in Annex I of this International Standard.

7.4.11.1 Filename

The filename of the common message components schema shall be "24531-CommonMessageComponents-2.0.xsd", where the 2.0 will be periodically updated to reflect the actual version.

7.4.11.2 Namespace identifier

The common message components schema shall be assigned the namespace identifier of "msg" when imported into other schemas.

7.4.12 Rules for message schemas

Each message shall be defined in its own message schema. Example message schemas are provided in Annex J and Annex K.

7.4.12.1 Filename

The filename shall indicate the ISO standard number, a hyphen, the name of the message, another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be ".xsd".

EXAMPLE

```
24531-IncidentList-1.0a.xsd
```

7.4.12.2 Single global element

The message schema shall contain a single global element, which represents the message itself.

7.4.12.3 Name of element

The name of the message element shall be the Descriptive Name of the message as defined in the ITS Data Registry, but without the parenthesis, colon, or term "message".

NOTE This means that the message shall always start with an uppercase letter. The XML element name for the message will always be the same as its UML class name.

EXAMPLE For the Descriptive Name "MyMessage():message", the element name would be "MyMessage".

7.4.12.4 Type of element

The type of the message element shall be the name of its corresponding complexType as defined in 7.4.6.4.

7.4.12.5 Single complex type

The message schema shall contain a single complexType, which shall define the message structure.

7.4.12.6 Structure of complex type

The message complexType shall be constructed according to the same rules defined for data frame complex types, as defined in 7.4.6.4.

7.4.12.7 Extensibility of messages

Every message shall inherit the Message structure as defined in Annex B.

NOTE Including the extension as the first element of the message allows streaming XML processors to buffer the extension and process the contents at the correct location in the stream. If the extension is placed at the end of the XML message, the processor would be forced to buffer the entire message before processing any portion of the message.

7.4.13 Rules for default code lists

Default code lists shall be defined by the ISO/TC 204 working group for every element that uses codes. An example default code list is provided in Annex L.

7.4.13.1 Code lists defined using genericode

Each code list shall be defined using the genericode file format.

7.4.13.2 Keys

The code list table shall include one or more columns defined as keys. Each key column shall be defined as a "Column" and as a "Key" in the "ColumnSet".

7.4.13.3 Status Columns

The code list table shall include a column that indicates whether support for the code defined in the row is mandatory or optional.

7.4.13.3.1 Mandatory codes

A conforming implementation shall support every value identified as mandatory in contexts that reference the code list.

7.4.13.3.2 Optional codes

A conforming implementation may support values that are defined as optional in contexts that reference the code list.

7.4.13.3.3 Conditional codes

A standard may define more complex rules to indicate when values must be supported.

NOTE For example, a standard might want to indicate that an implementation must support at least one of a group of three codes. Genericode is flexible enough to handle such definitions, but this International Standard does not dictate the exact format for handling such conditional statements.

7.4.13.4 Name column

The code list table should include at least one column to provide a short name for the code, unless the code itself provides a useful name.

7.4.13.5 Definition column

The code list table should include at least one column to provide a definition of the code, unless this information is self-explanatory from the other information given in the code.

7.4.14 Rules for default context-value associations

The ISO/TC 204 working group shall define a default definition of the valid value ranges allowed for each occurrence of each element throughout all possible message instances of all messages defined in their message set. An example default context-value association file is provided in Annex M.

7.4.14.1 Filename

The filename shall indicate the ISO standard number, a hyphen, "DefaultValidation", another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be ".cva".

EXAMPLE

```
24531-DefaultValidation-1.0a.cva
```

7.4.14.2 Context-value association file format

The valid value range for each occurrence of each element shall be defined using the context/value association file format.

NOTE The context/value association file format allows the identification of value lists (e.g. by referencing the genericode files) and value tests (e.g. to ensure values are within a defined range, within a defined length, follow a specific format, etc.) and to then associate these conditions with specific locations of data within messages by using XPath statements.

7.4.14.3 Value test conformance

Each value test in the context/value association file shall be annotated to indicate conformance requirements for implementations.

NOTE For some elements, the standard may want to define absolute minimal ranges that must be supported by all implementations while allowing much larger ranges as an option. For other elements, the standard may want to define the maximal range that is allowed while allowing implementations to only support a sub-range. Other elements may be associated with an exact range that must be supported, while still other elements may be associated with a recommended or suggested range while allowing for either larger or smaller range support. This International Standard merely requires that the context/value association file be explicit as to the conformance rules for the implementation.

7.4.14.4 Value list conformance

Each value list in the context/value association file shall be annotated to indicate conformance requirements for implementations for the list as a whole.

NOTE While the context/value association file will identify conformance requirements for specific rows in the code list, the reference to the list in the context/value association file will identify issues such as whether the list can be expanded to include other code lists.

7.4.15 Rules for default value validation transformation file

The default genericode and context/value association files shall be transformed into the default extensible stylesheet language transformation (XSLT) file. Annex N provides a sample CVA transformation file and Annex O provides an example default value validation transformation file that would be produced by applying the Annex N transformation to the CVA file provided in Annex M.

NOTE The value validation transformation file is a powerful tool that can be used to validate the information contained in an XML instance document, but the XSLT file format can be somewhat complex to understand. The genericode and context/value association files define the value validation rules in a format that can readily be transformed (via XSLT) into either a user-friendly table format (e.g. in HTML) or into the default value validation transformation file (i.e. also in XSLT).

7.4.15.1 Filename

The filename shall indicate the ISO standard number, a hyphen, “DefaultValidation”, another hyphen, and the major and minor version number; if the document is not yet published, the revision letter shall also be appended to the end of this name. The file extension shall be “.xsl”.

EXAMPLE

```
24531-DefaultValidation-1.0a.xsl
```

7.4.16 Rules for customizing genericode files

An implementer should publish a customized genericode file that represents only values that the implementation supports. A sample customized genericode file is provided in Annex P.

7.4.16.1 Removing codes

An implementer may remove optional codes from the genericode files.

7.4.16.2 No new codes

An implementer shall not add codes to a genericode file defined by a standards organization.

NOTE An implementer may create a separate genericode file to document custom and/or local codes; however, each list should be uniquely named such that the pairing of a code with its version-specific list will produce a globally and temporally unique interpretation of the code. The new genericode file can be added to the customized context/value association file, if allowed by the rules of that file.

7.4.17 Rules for customizing context/value association files

An implementer may edit the default context/value association file to the extent allowed by the rules contained in the file. Annex Q provides a sample customized context-value association file.

7.4.18 Rules for customizing the value validation transformation file

An implementer shall produce a new value validation transformation file based on the customized genericode files and the customized context/value association file. Annex R provides a sample customized value validation transformation file.

7.4.19 Rules for XML instance documents

XML instance documents shall conform to the W3C Extensible Mark-up Language (XML). Sample XML instance documents are provided in Annex U and Annex V.

7.4.19.1 XML declaration

Every conformant instance document shall start with an XML declaration element.

7.4.19.1.1 XML version

The XML declaration element shall include a version attribute.

7.4.19.1.2 XML encoding

The XML declaration element shall include encoding attribute that specifies UTF-8 encoding.

7.4.19.2 Namespaces

Every conformant instance document shall explicitly define all referenced namespaces.

NOTE This will typically include the namespace of the message schema and the namespace of the component schema(s) (i.e. “cc” and “msc”).

7.4.19.3 Conformant instance document

A conformant instance document shall validate against its standardized message schema.

NOTE This allows for an instance document to include extensions using the standardized extensions mechanism, but does not require a receiving system to support handling of this extended data. It does not include any range checking of values since it is assumed that local implementation may extend or restrain valid values differently than the standard.

7.4.19.3.1 Indication of code list

A conformant instance document shall explicitly include the listName and listVersionID attributes element that references a code list other than the default code list for the element.

7.4.19.4 Compliant instance document

A compliant instance document shall be a conformant instance document that can be transformed by the receiving system’s customized value validation transformation file without generating any error or warning messages.

NOTE A compliant instance document may contain unrecognized extensions that will be ignored by the receiving system. However, values for all data fields will be validated as acceptable by the receiving system.

7.4.19.5 Null values

Conformant instance documents shall not contain any elements that contain null values or are devoid of content.

7.4.19.6 Omitted data

The absence of an element or attribute shall not contain meaning, unless the schema standard explicitly defines a default value for the element or attribute.

7.4.19.7 Interpretation of attributes

An attribute defined for a structural element shall apply to all the descendant elements contained within the structural element.

8 Rules for registration and management of XML schema constructs in data registry (DR) and/or data dictionaries (DDs)

8.1 Objectives of schema constructs registration and management

Schemas should be viewed as one of the final artifacts of a system’s engineering process. The process starts by identifying a set of user needs and refining these needs into requirements. The requirements then lead to a design that identifies a set of necessary data exchanges between two or more entities where each exchange can be defined as consisting of a set of known information and well-defined information. If there is a desire to use XML to achieve this exchange, an XML schema should be created to formally represent this exchange; however, all data concepts defined in the schema should be previously defined as a part of the system’s engineering process.

8.2 Why use ISO 14817 data registry/ data dictionary (DR/DD)?

ISO 14817 defines a variety of data concepts that can be registered within the ITS industry to promote the reuse of data. This facilitates efforts to standardize terminology and data formats. For example, one application area might define a person to have a first, middle, and last name while another might also define a person's name to have a salutation and a suffix. A data registry or data dictionary can assist identifying and harmonizing these differences so that the same constructs can be used throughout the industry.

Although ISO 14817 is based on ASN.1, the existence of well-defined mapping rules (such as ISO 8825-4 and ISO 8825-5) allows for ready conversion between the different representational forms. This allows data harmonization among all of the various implementation languages.

Figure 6 depicts relationships between XML schema constructs and ISO 14817 DR/DD.

NOTE ASN.1<->XML transformation is done if necessary.

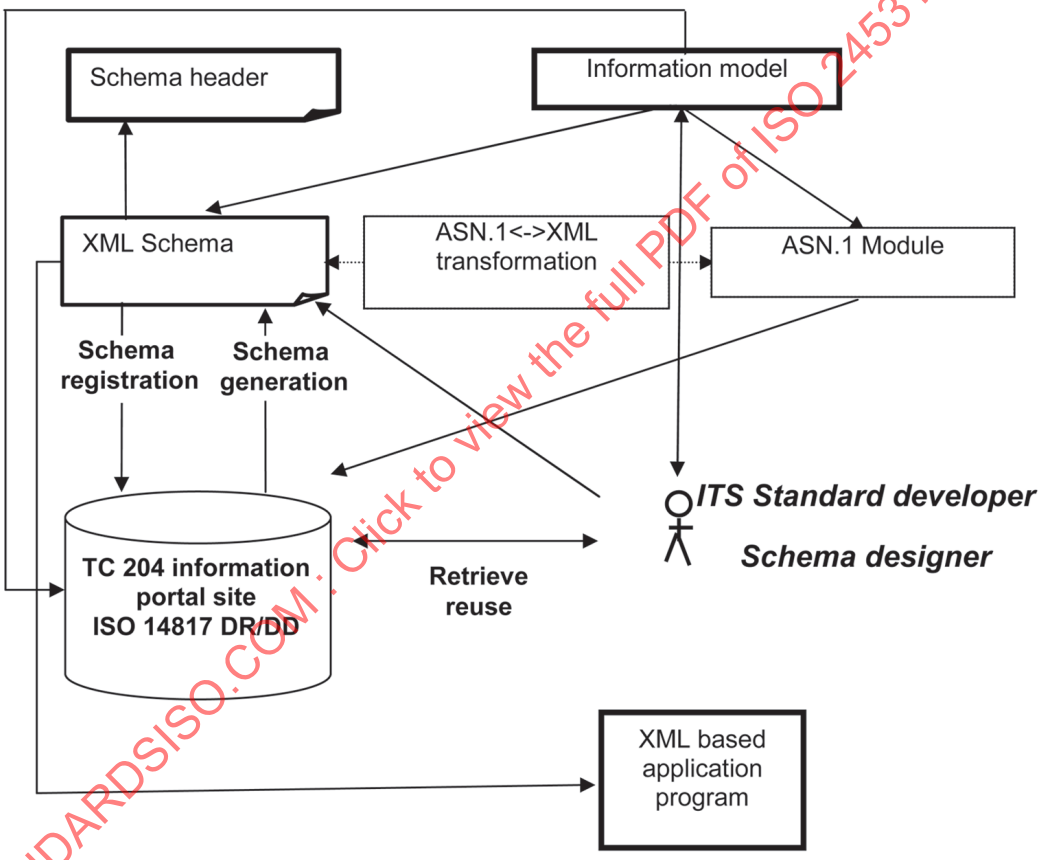


Figure 6 — Relationships of XML schema constructs and ISO 14817 DR/DD

8.3 Schema constructs mapping to the ISO 14818 constructs

Table 3 identifies how ISO 14817 data concepts are represented in schemas and UML. Annex A provides additional information as to how data concepts may be mapped among various forms.

Table 3 — Schema constructs mapping scheme for ISO 14817 constructs

ISO 14817 data concept	Schema representation	UML representation
Object Class	N/A - schemas do not attempt to portray the Object Class View, only the Message view; see 7.3.3.	Class (no stereotype)
Property	A globally defined xs:element	N/A – UML always shows attributes as a part of a class
Value Domain	xs:complexType (should be defined in Unqualified Data Types schema)	Type
Data Element Concept	N/A – all schema concepts have a representational form.	Attribute
Data Element	A referenced (i.e. "ref") xs:element within a Message or Data Frame construct.	Attribute with type
Data Frame	An xs:element and an xs:complexType. The xs:element references the xs:complexType. These are defined in either the common components schema or the message set components schema.	Class (stereotype of "Data Frame")
Message	An xs:element and an xs:complexType defined in a dedicated message schema file. The xs:element references the xs:complexType.	Operation
Interface Dialog	N/A (can be separately defined as a WSDL file)	Sequence diagram
Association Role	An xs:element that references an xs:complexType of another Data Frame.	Role

8.4 Registration and management rules

8.4.1 Referencing of DR or DD when creating schemas

When creating schemas, a check shall be made to verify that the schema elements are already registered in the ISO 14817 data registry and data dictionary.

NOTE This procedure is necessary in order to enable portal sites to keep the vocabulary.

8.4.2 Registering items not registered in DR/DD

If an XML component identified in Table 3 has not been registered, the component shall be registered in the appropriate DR/DD.

Annex A
(informative)

Model/document transformation

A.1 Model/document transformation among XML, UML, ASN.1

UML, XML schema, and ASN.1 have their own target area of usage, and no language is superior for all domains. Mastering every language is impractical, so it is helpful to use standardized language transformation supported by software tools. Figure A.1 shows various language transformation standards.

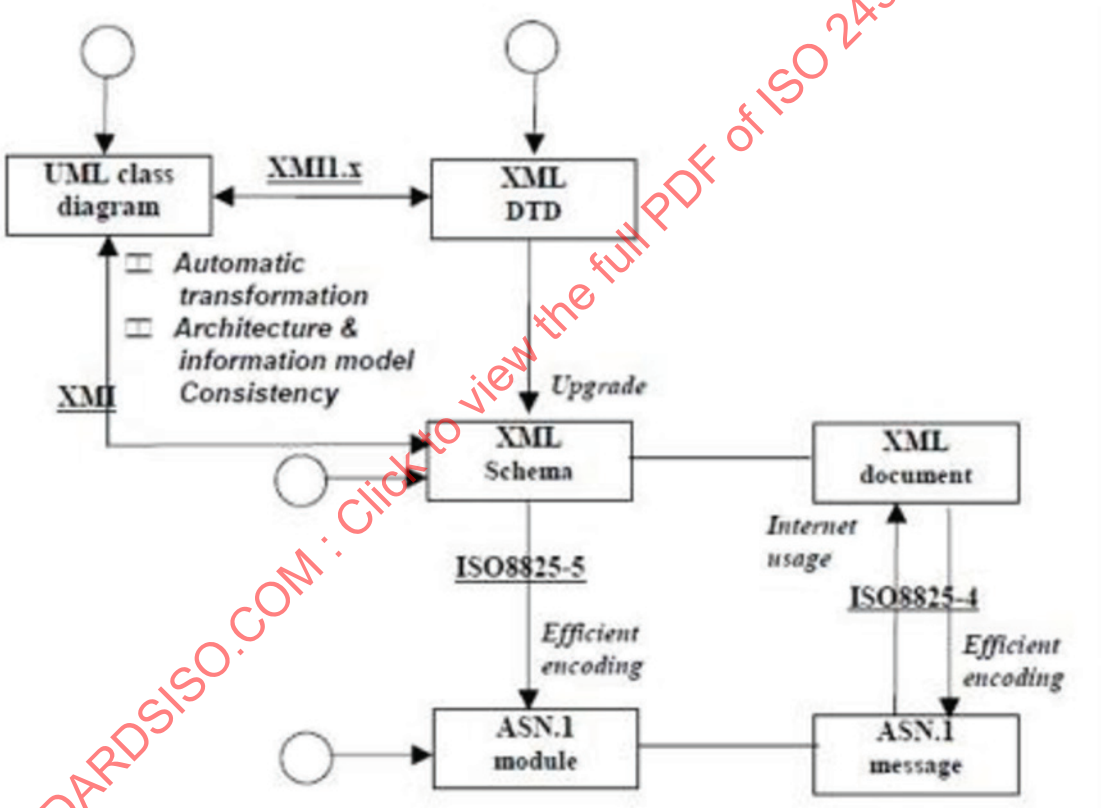


Figure A.1 — Language transformation standards and typical objectives of transformation

Key

◯	designing start point
italics	typical objectives
underlined	related standards

A.2 XML applications in ITS

Figure A.2 depicts representative XML applications in ITS.

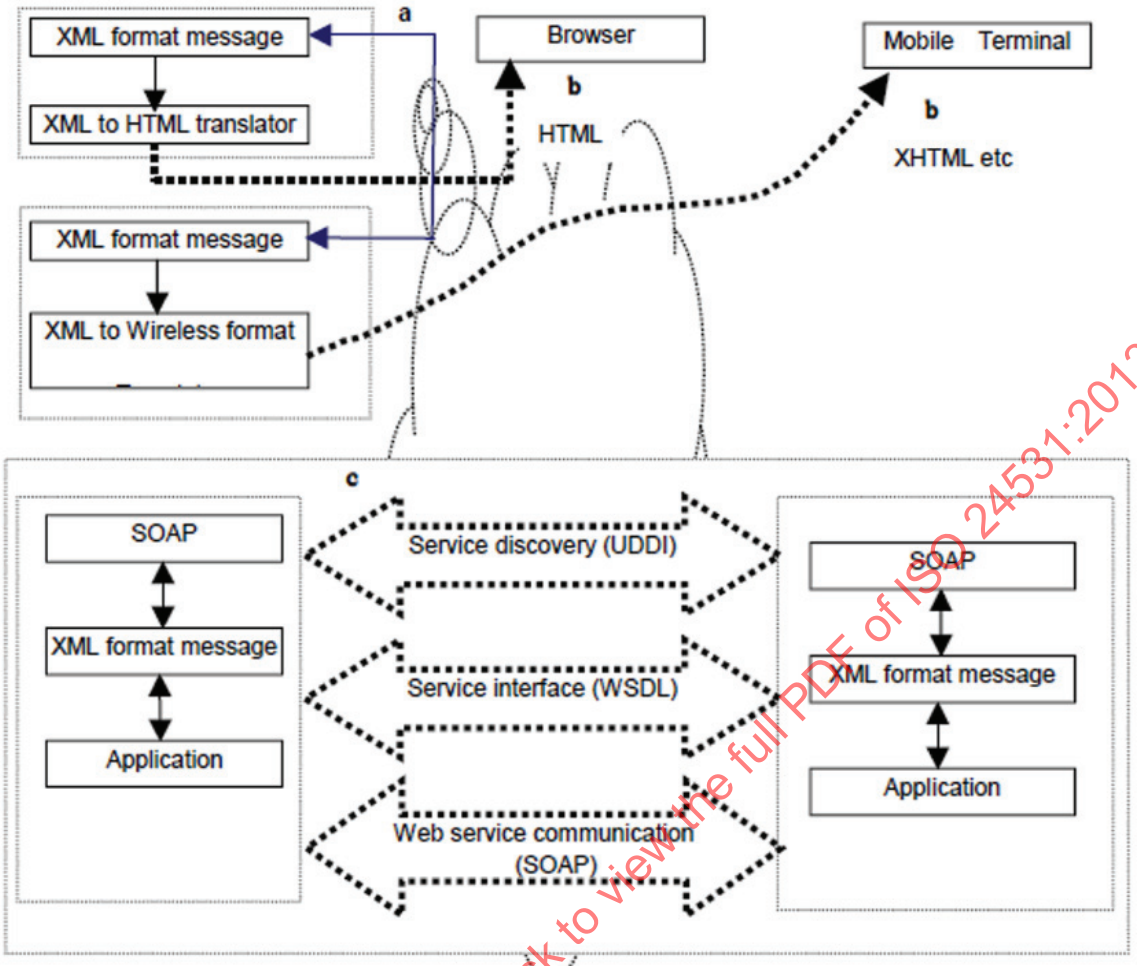


Figure A.2 — XML application in ITS

a) Message interchange (Figure A.2, a)

XML does not depend on hardware and operating systems. It is frequently used as a middleware in heterogeneous circumstances. An XML schema allows rigorous validation of the message structure in a pre-processor; as a result, the ITS application program can be simplified since it does not need to perform much of this data validation.

However, once an interface has been designed and implemented, any changes can break interoperability and require program modifications; thus, it is important to carefully design the schema so that it is expandable with minimal interoperability impacts. To achieve this requirement, this International Standard defines a set of processes and rules for both the interface design and the schema design.

The interface design recommends the use of UML because it eases the understanding among the stakeholders. If a national ITS architecture is described by UML in a sufficient detail of granularity, it is possible to create XML schema automatically, and this enables strict reconciliation with the architecture, and reverse engineering maintenance of the architecture is possible.

By cooperating with other ITS applications and ITS adjacent applications (e.g. e-commerce, e-government, etc.), the effect is increased. Messages using validated schema make application cooperation possible. If relevant XML schema already exist, this International Standard recommends such reuse because it brings

interoperability between applications. Once the schema is registered, it can be reused by the "include" or "import" mechanisms of XML. A UML class diagram helps more easily to find reusable schema.

NOTE XML message interchange is used for various ITS applications. For example, message interchange between traffic control centres, weather information, event information, telemetric services, and operational data exchange for commercial vehicles, are typical usages of XML messages.

b) Internet browsing (Figure A.2, b)

XML documents are easily transformed to HTML or XHTML by using the XML application interface "*DOM*" or "*XSLT*". A typical ITS application of Internet browsing is a traveller information system.

c) Web services (Figure A.2, c)

Web services are anticipated to be one of the main applications for the use of XML. The W3C defines web services as follows: *"A web service is a software system identified by a URI (RFC 2396), whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols*."

The term "web service" describes a standardized way of integrating web-based applications using the XML, SOAP, WSDL, and UDDI open standards over an Internet protocol backbone. XML is used to make message, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available. Used primarily as a means for businesses to communicate with each other and with clients, web services allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall.

Web services allow different applications from different sources to communicate with each other without time consuming custom coding, and because all communication is in XML, web services are not tied to any one operating system or programming language. Comparing EDI or CORBA it is said to be easier, and flexible coordination is possible.

Annex B
(normative)

Definition of the Message class

The Message class is the generalized class that serves as the parent to all ITS messages to ensure certain capabilities are provided by all messages. Figure B.1 provides a UML depiction of the Message class and the ISO 14817 definition is provided afterwards.

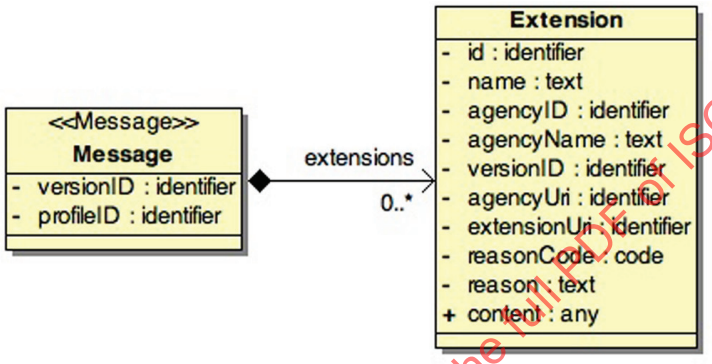


Figure B.1 — The Message class

Table B.1 — Definition of the Message class

Message	
Descriptive Name	Message():message
Descriptive Name Context	ITS
Definition	A generic message container; used as the generalized class for all messages.

Message Body	SEQUENCE {
	versionID Identifier OPTIONAL,
	profileID Identifier OPTIONAL,
	extensions SEQUENCE OF Extension OPTIONAL
	}

Standard	ISO 24531
Data Concept Type	Message
Meta-data Source	direct
Message Identifier	iso(1) standard(0) iso24531(24531) annex(1) b(8)

B.1 versionID

The definition of Message.versionID:identifier is provided in Table B.2.

Table B.2 — Definition of Message.versionID:identifier

Message.versionID:identifier	
Descriptive Name	Message.versionID:identifier
Descriptive Name Context	ITS
Definition	The earliest version of the message that defines all of the elements that might be encountered in the current instance.
Data Type	Identifier
Format	#.#
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.2 profileID

The definition of Message.profileID:identifier is provided in Table B.3.

Table B.3 — Definition of Message.profileID:identifier

Message.profileID:identifier	
Descriptive Name	Message.profileID:identifier
Descriptive Name Context	ITS
Definition	Identifies a user-defined customized profile of the message being used.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.2.1 extensions

The definition of Message->extensionsExtension is provided in Table B.4.

Table B.4 — Definition of Message->extensionsExtension

Message->extensionsExtension	
Descriptive Name	Message->extensionsExtension
Descriptive Name Context	ITS
Definition	A container for all extensions present in the document.
Roles	extensions
Multiplicity	0..*
Referenced Object Classes	Extension
Standard	ISO 24531
Data Concept Type	Association

B.3 Extension

The definition of Extension is provided in Table B.5.

Table B.5 — Definition of Extension

Extension	
Descriptive Name	Extension
Descriptive Name Context	ITS
Definition	A single extension for custom or local use.

Message Body	SEQUENCE {
	id Identifier OPTIONAL,
	name Text OPTIONAL,
	agencyID Identifier OPTIONAL,
	agencyName Text OPTIONAL,
	versionID Identifier OPTIONAL,
	agencyUri Identifier OPTIONAL,
	extensionUri Identifier OPTIONAL,
	reasonCode Code OPTIONAL,
	reason Text OPTIONAL,
	content Any
	}

Standard	ISO 24531
Data Concept Type	Data Frame

B.3.1 Extension.id:identifier

The definition of Extension.id:identifier is provided in Table B.6.

Table B.6 — Definition of Extension.id:identifier

Extension.id:identifier	
Descriptive Name	Extension.id:identifier
Descriptive Name Context	ITS
Definition	An identifier for the Extension assigned by the creator of the extension.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.2 Extension.name:text

The definition of Extension.name:text is provided in Table B.7.

Table B.7 — Definition of Extension.name:text

Extension.name:text	
Descriptive Name	Extension.name:text
Descriptive Name Context	ITS
Definition	A name for the Extension assigned by the creator of the extension.
Data Type	Text
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.3 Extension.agencyID:identifier

The definition of Extension.agencyID:identifier is provided in Table B.8.

Table B.8 — Definition of Extension.agencyID:identifier

Extension.agencyID:identifier	
Descriptive Name	Extension.agencyID:identifier
Descriptive Name Context	ITS
Definition	An agency that maintains one or more Extensions.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.4 Extension.agencyName:text

The definition of Extension.agencyName:text is provided in Table B.9.

Table B.9 — Definition of Extension.agencyName:text

Extension.agencyName:text	
Descriptive Name	Extension.agencyName:text
Descriptive Name Context	ITS
Definition	The name of the agency that maintains the Extension.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.5 Extension.versionID:identifier

The definition of Extension.versionID:identifier is provided in Table B.10.

Table B.10 — Definition of Extension.versionID:identifier

Extension.versionID:identifier	
Descriptive Name	Extension.versionID:identifier
Descriptive Name Context	ITS
Definition	The version of the Extension.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.6 Extension.agencyUri:identifier

The definition of Extension.agencyUri:identifier is provided in Table B.11.

Table B.11 — Definition of Extension.agencyUri:identifier

Extension.agencyUri:identifier	
Descriptive Name	Extension.agencyUri:identifier
Descriptive Name Context	ITS
Definition	A URI for the Agency that maintains the Extension.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.7 Extension.extensionUri:identifier

The definition of Extension.extensionUri:identifier is provided in Table B.12.

Table B.12 — Definition of Extension.extensionUri:identifier

Extension.extensionUri:identifier	
Descriptive Name	Extension.extensionUri:identifier
Descriptive Name Context	ITS
Definition	A URI for the Extension.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.8 Extension.reasonCode:code

The definition of Extension.reasonCode:code is provided in Table B.13.

Table B.13 — Definition of Extension.reasonCode:code

Extension.reasonCode:code	
Descriptive Name	Extension.reasonCode:code
Descriptive Name Context	ITS
Definition	A code for reason the Extension is being included.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.9 Extension.reason:text

The definition of Extension.reason:text is provided in Table B.14.

Table B.14 — Definition of Extension.reason:text

Extension.reason:text	
Descriptive Name	Extension.reason:text
Descriptive Name Context	ITS
Definition	A description of the reason for the Extension.
Data Type	Identifier
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

B.3.10 Extension.content:any

The definition of Extension.content:any is provided in Table B.15.

Table B.15 — Definition of Extension.content:any

Extension.content:any	
Descriptive Name	Extension.content:any
Descriptive Name Context	ITS
Definition	The definition of the extension content.
Data Type	Any
Format	
Unit of Measure	
Valid Value Rule	
Standard	ISO 24531
Data Concept Type	Data Element

Annex C
(informative)

Example Message Exchange: Model

C.1 Background

In many cases, complex ideas are best explained through a complete example. The remaining annexes of this International Standard are intended to provide such an example so that users can more fully appreciate the design principles contained in this International Standard.

The example is intentionally kept as simple as possible while also attempting to show many of the more common issues a real-world scenario may encounter. The example is based on the concept that one system has a need to obtain a listing of traffic incidents from a remote system. This example is chosen because it is a topic that is widely understood, at least at a conceptual level. However, any real-world analysis of this need would result in a much more complex and robust data model. It is not the intent of this International Standard to completely document an Incident, rather, the example has specifically been simplified in order to keep the sample as brief as possible while still describing common situations. The example is strictly informative, except as follows:

— Annex B provides a normative definition of a Message Class for use by all ISO/TC 204 XML standards

— Annex D provides a normative definition of the Unqualified Data Type schema for use by all ISO/TC 204 XML standards

— Annex E provides a normative definition of the Common Basic Components schema that is to be used as a basis for all ISO/TC 204 XML standards

— Annex F provides a normative definition of the Common Aggregate Components schema for use by all ISO/TC 204 XML standards

— Annex G provides a normative definition of the Common Extension Components schema that is to be used as a basis for all ISO/TC 204 XML standards

— Annex H provides a normative definition of the Extension Content Data Type schema that is to be used as a basis for all ISO/TC 204 XML standards

— Annex I provides a normative definition of the Common Message Components schema for use by all ISO/TC 204 XML standards

C.2 Data exchange definition

Every message exchange should be defined in a UML sequence diagram. The design of the diagram is based on an analysis of the user needs and requirements, that are not the focus of this example. However, to keep the example simple, it will be assumed that the exchange is relatively simple and consists of a request for an incident list followed by the incident list being transmitted. This is depicted in Figure C.1.

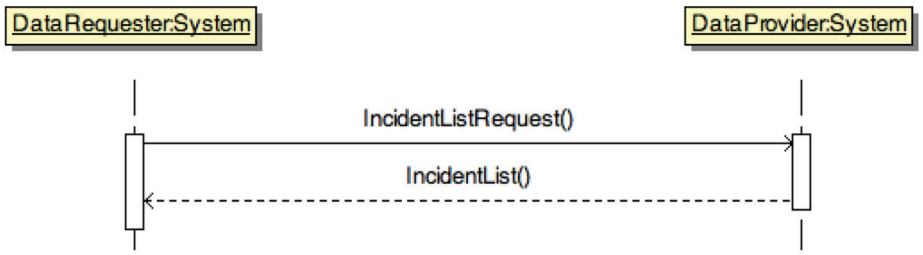


Figure C.1 — Example Message Exchange

Real-world exchanges may be this simple or might be more complex. They may include multiple steps and/or multiple entities exchanging data in a pattern.

C.3 Logical Model of the IncidentListRquest

The request for the example will contain information as determined from an analysis of user needs. It will be assumed that this analysis indicates the need to base the request on an area of interest. This is depicted in Figure C.2.

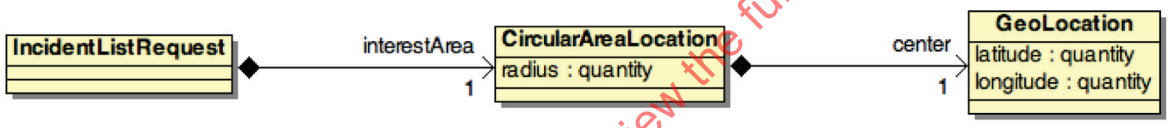


Figure C.2 — A sample logical model for the IncidentListRequest

Each entity defined in the logical data model would be formally defined in the data dictionary or data registry according to the rules of ISO 14817.

C.4 Message definition for IncidentListRequest

The logical model shown above can also be presented as a model depicting messages. In this case, all of the logical information is included in the proposed messages, so the two models appear to be identical, except for the fact that the entities have now been stereotyped as either a <<Message>> or a <<Data Frame>>.

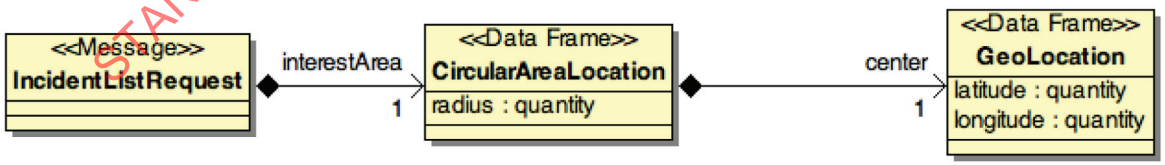


Figure C.3 — A sample message model of IncidentListRequest

C.5 Logical Model for IncidentList

The response is based on the logical model for an IncidentList. The analysis of user needs compared with previous work done by others might result in a complex model as shown in Figure C.4.

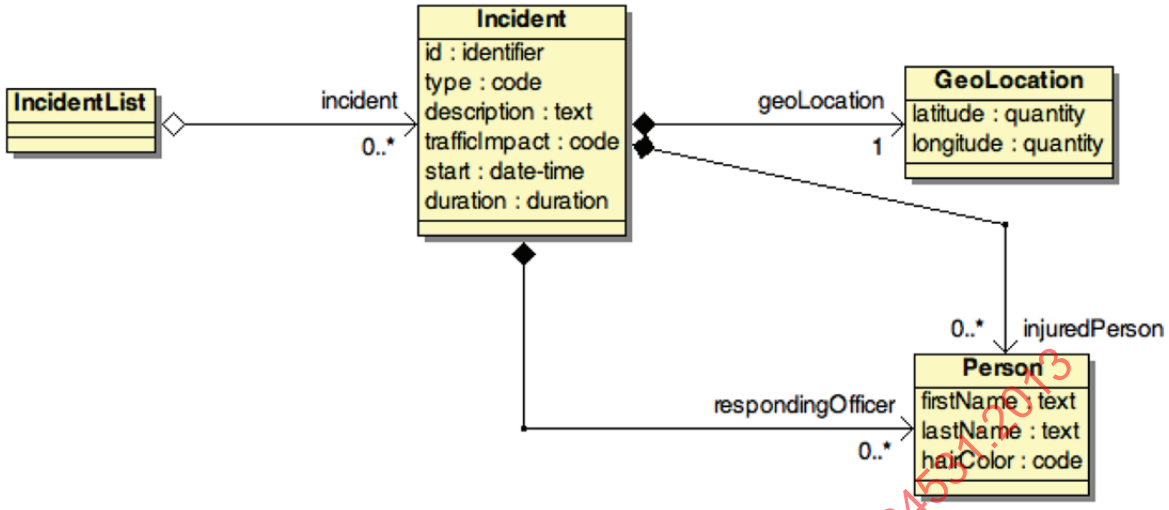
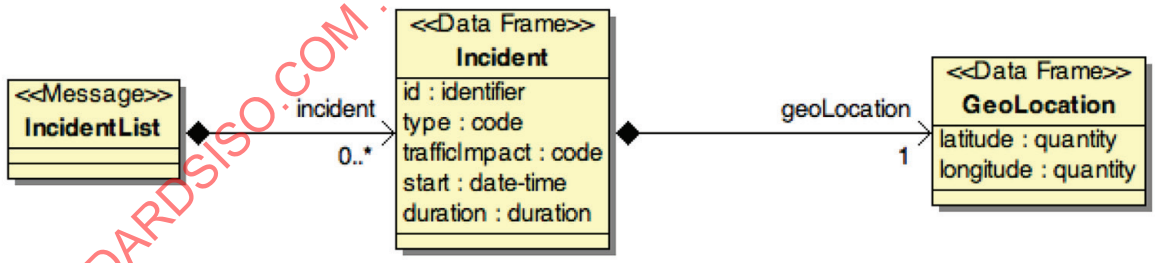


Figure C.4 — A sample logical model for IncidentList

Once again, this each data concept shown in this model would have a corresponding entry in the data dictionary or data registry.

C.6 Message definition for IncidentList

However, it may be decided that it is not necessary to exchange some of the more detailed information for these particular purpose, or at least not in the current high-level exchange that is currently being defined. For example, most users of incident data might not be interested in or might not have privileges to see some of the data such as the injured persons or responding officer data. Thus, in this case, the message model is a simplified version of the logical model as shown in Figure C.5.



Annex D
(normative)

Unqualified data types schema

D.1 General

This annex provides the formal definition of the ISO/TC 204 unqualified data types schema that should be referenced by all ISO/TC 204 XML schema standards.

D.2 Unqualified data types overview

The unqualified data types schema defines 16 data types that should be used to represent all ISO/TC 204 data elements. The following clauses summarize these data types.

D.2.1 Amount type

The amount type is based on the ISO 14817 amount value domain and the UBL amount data type. It is defined as a decimal value with attributes that describe the currency of the amount.

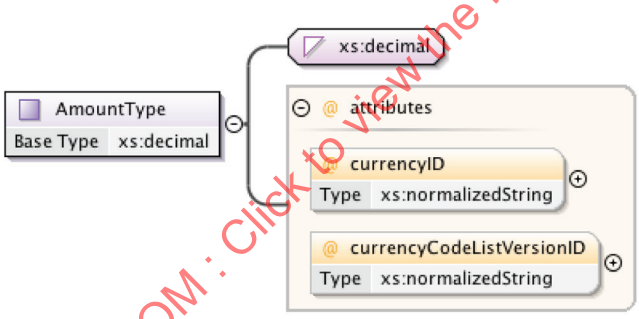


Figure D.1 — Amount type

D.2.2 Binary object type

The binary object type is based on the UBL binary object data type and serves as a foundation for the image and sound value domains as defined in ISO 14817. It is defined as a base64binary value with several attributes to describe the format, encoding, and file information.

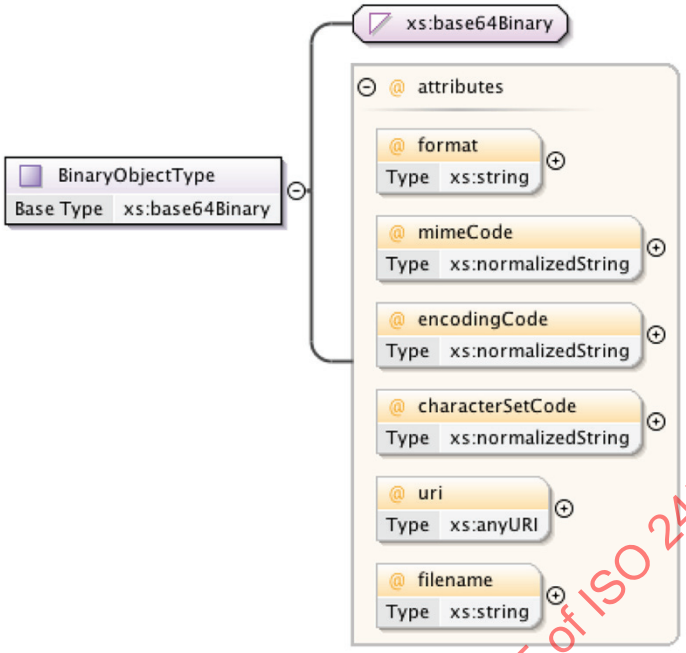


Figure D.2 — Binary object type

D.2.3 Code type

The code type is based on the ISO 14817 code value domain and the UBL code data type. It is defined as a normalized string with attributes that describe the code list from which the code is derived.

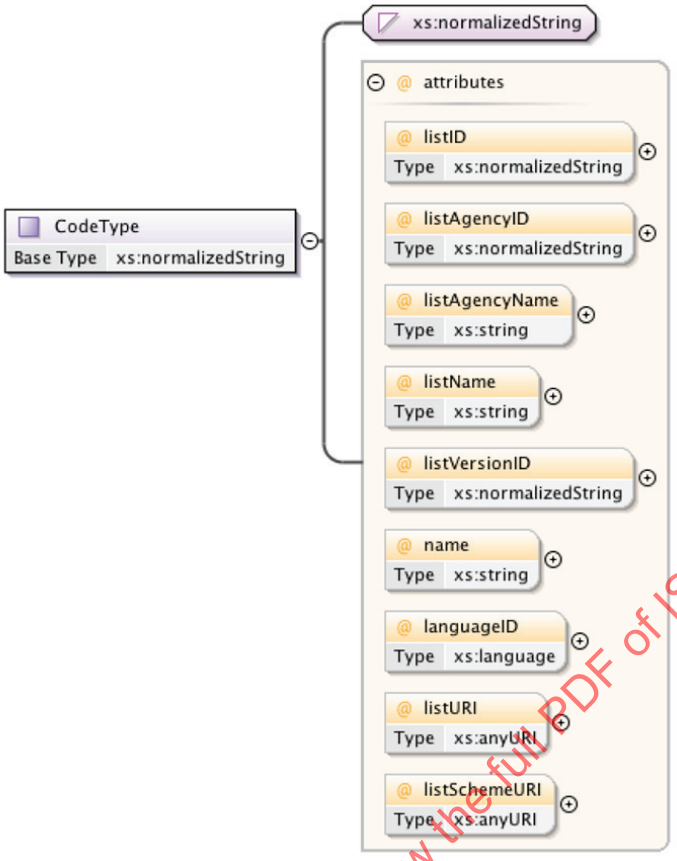


Figure D.3 — Code type

D.2.4 Date-time type

The date-time type is based on the ISO 14817 GPS value domain and the UBL date-time data type. It is defined as a date-time value (i.e. a formatted string) with no attributes.

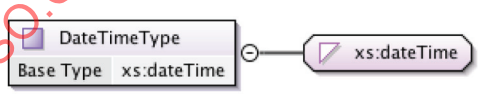


Figure D.4 — Date-time type

D.2.5 Date type

The date type is based on the ISO 14817 date value domain and the UBL date data type. It is defined as a date with no attributes.

Figure D.5 — Date type

D.2.6 Duration type

The duration type is based on the ISO 14817 value domain of UTC, but is not limited to a 24 hour period. It is defined as a duration with no attributes.

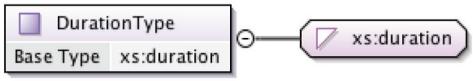


Figure D.6 — Duration type

D.2.7 Graphic type

The graphic type is based on the UBL graphic data type, which is a specialization of the binary object type.

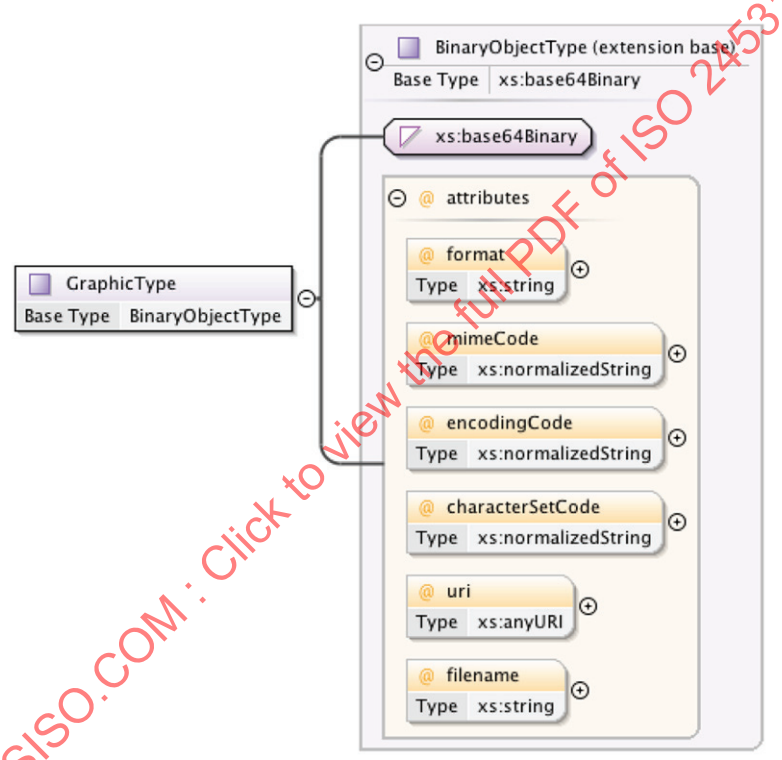


Figure D.7 — Graphic type

D.2.8 Identifier type

The identifier type is based on the ISO 14817 identifier value domain and the UBL identifier data type. It is defined as a normalized string with attributes that describe the identification scheme from which the identifier is derived.

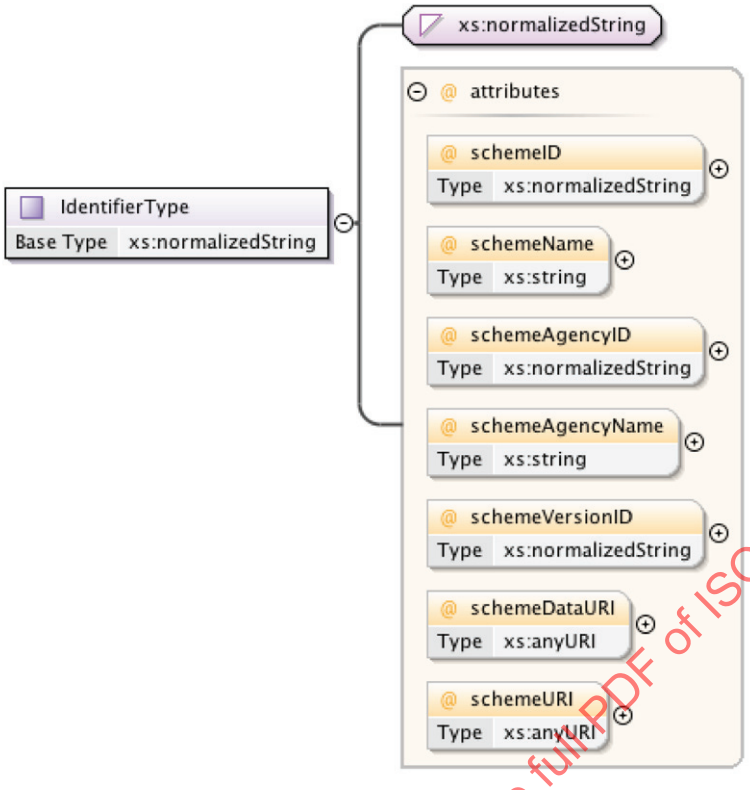


Figure D.8 — Identifier type

D.2.9 Indicator type

The indicator type is based on the UBL indicator data type; its closest match within ISO 14817 is the code value domain, although it is explicitly restricted to two values. It is defined as a Boolean value.

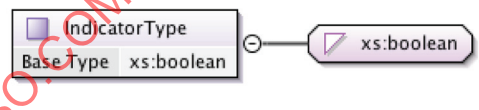


Figure D.9 — Indicator type

D.2.10 Measure type

The measure type is based on the UBL measure type; its closest match within ISO 14817 is the quantity value domain.

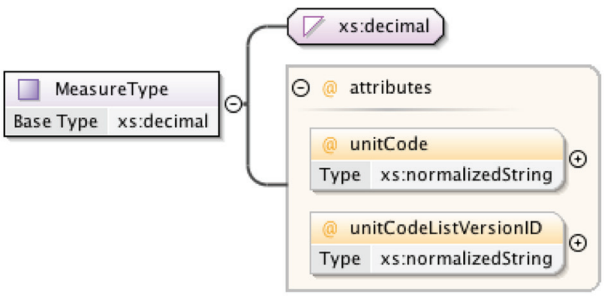


Figure D.10 — Measure type

D.2.11 Name type

The name type is based on the UBL name type; its closest match within ISO 14817 is the text value domain.

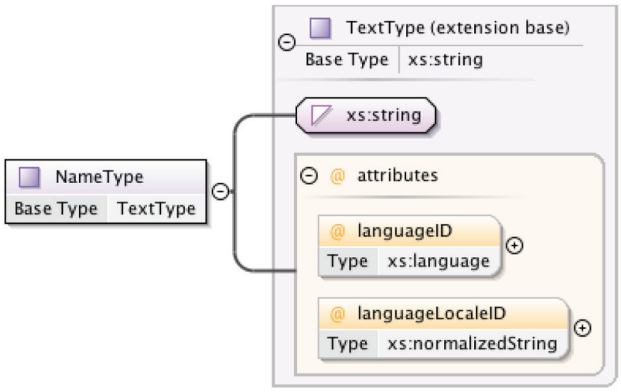


Figure D.11 — Name type

D.2.12 Numeric type

The numeric type is based on the ISO 14817 number value domain and the UML numeric data type. It is defined as a decimal value with an attribute to describe its format.

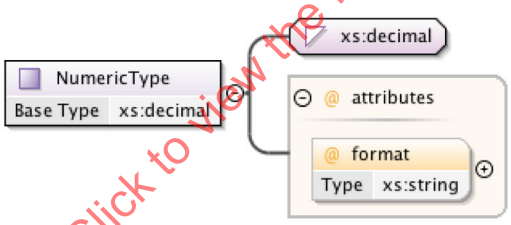


Figure D.12 — Number type

D.2.13 Percent type

The percent type is based on the ISO 14817 percent value domain and the UBL percent data type. It is defined as a number type.

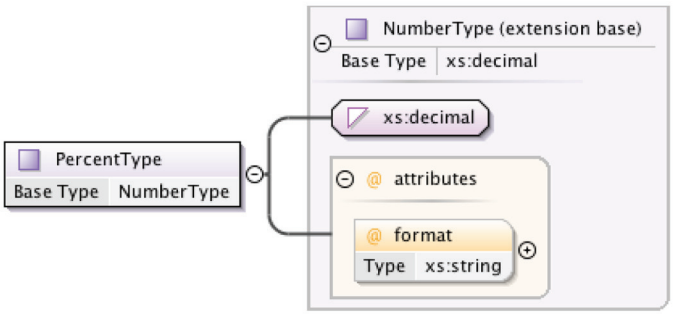


Figure D13 — Percent type

D.2.14 Picture type

The picture type is based on the UBL picture type; the closest ISO 14817 value domain is the image type.

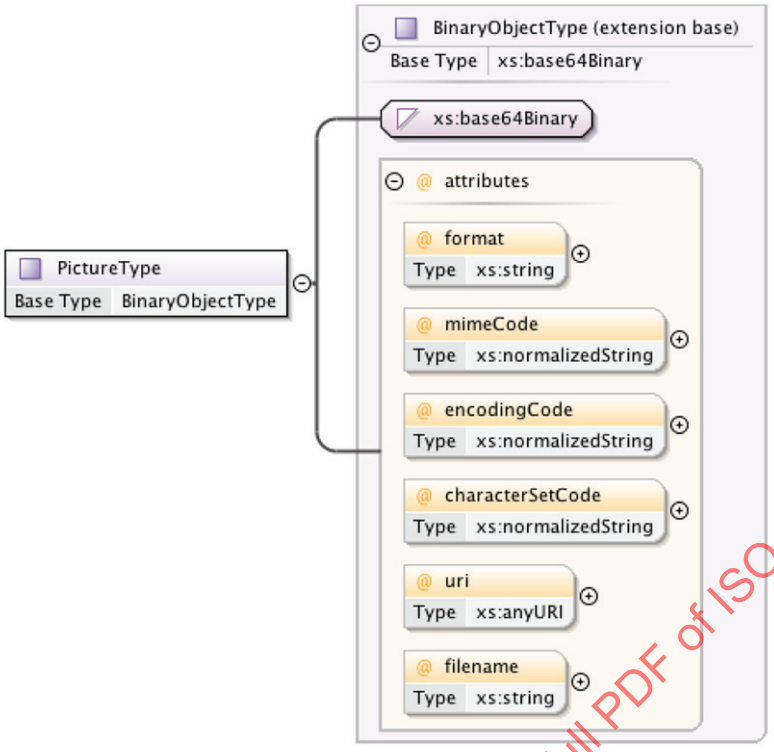


Figure D.14 — Picture type

D.2.15 Quantity type

The quantity type is based on the ISO 14817 quantity value domain and the UBL quantity data type. It is defined as a decimal value with attributes to describe the units that the quantity is measured in.

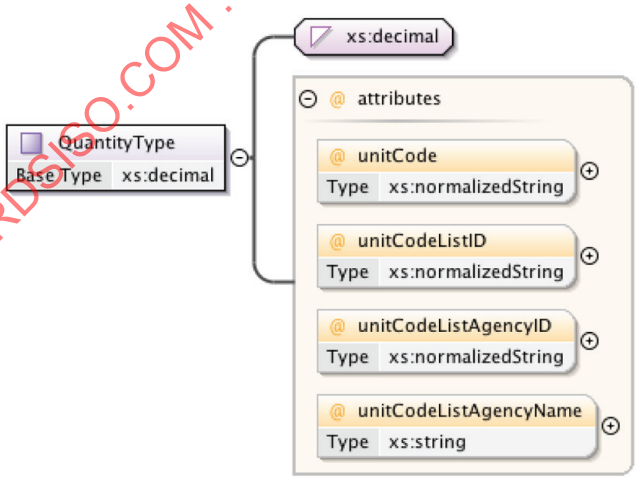


Figure D.15 — Quantity type

D.2.16 Rate type

The number type is based on the ISO 14817 rate value domain and the UBL rate data type. It is defined as a quantity type.

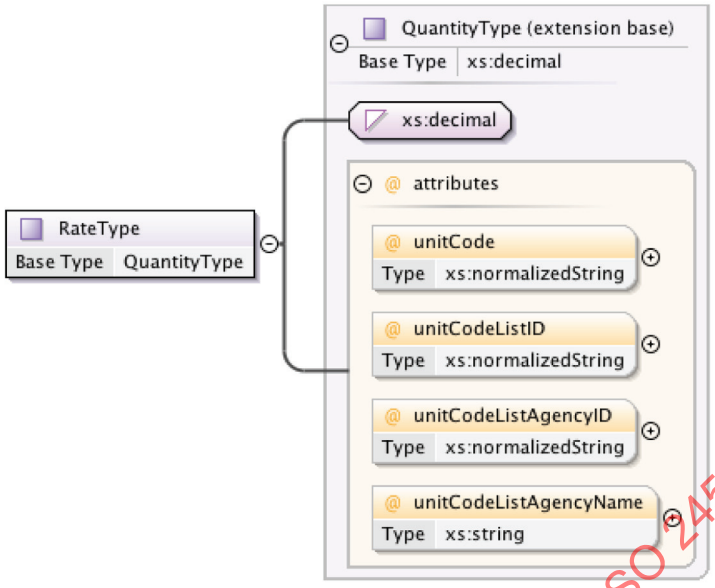


Figure D.16 — Rate type

D.2.17 Sound type

The sound type is based on the ISO 14817 sound value domain and the UBL sound data type. It is defined as a binary object type.

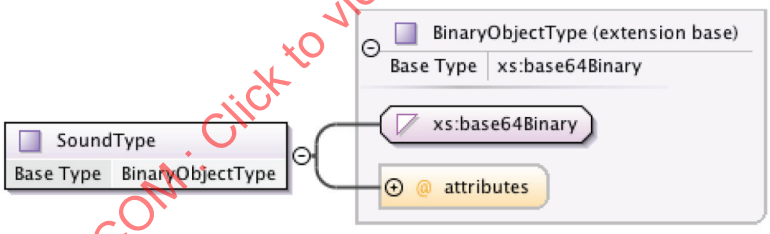


Figure D.17 — Sound type

D.2.18 Text type

The text type is based on the ISO 14817 text value domain and the UBL text data type. It is defined as a string with attributes to describe the language.

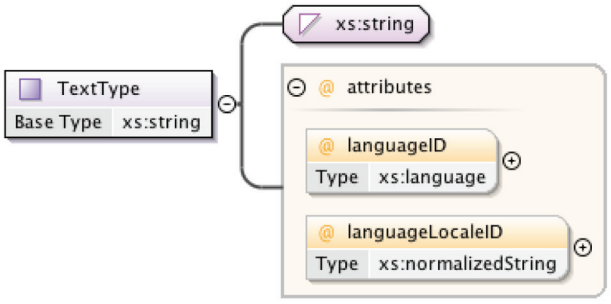


Figure D.18 — Text type

D.2.19 Time type

The time type is based on the ISO 14817 GPS value domain and the UBL time data type. It is defined as a time value.

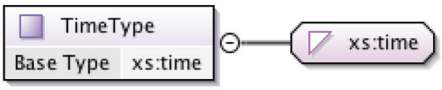


Figure D.19 — Time type

D.2.20 Value type

The value type is based on the UBL value data type; the closest ISO 14817 value domain is the number type.

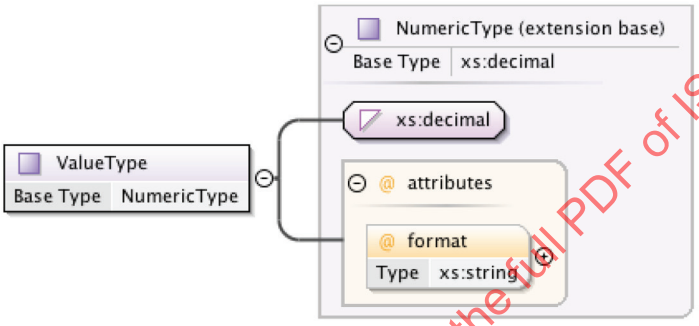


Figure D.20 — Value type

D.2.21 Video type

The video type is based on the UBL video data type; the closest ISO 14817 value domain is the image type.

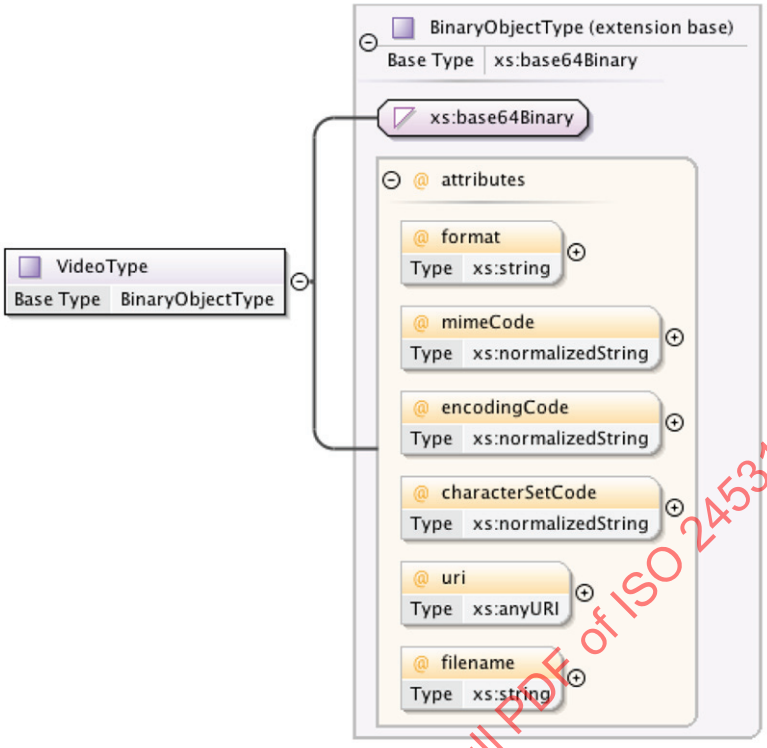


Figure D.21 — Video type

D.3 Unqualified data types schema

```
 <?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   targetNamespace="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Unqualified Data Types</doc:schemaName>
             <doc:description>
                    Definition of the data types to be used in ISO TC 204 schemas
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                 <doc:version>2.0a</doc:version>
                 <doc:date>2011-08-07</doc:date>
                 <doc:description>Original version</doc:description>
             </doc:change>
             <doc:change>
```

```
                <doc:version>2.0b</doc:version>
                <doc:date>2011-10-13</doc:date>
                <doc:description>Revised documentation to conform to common
                   format. Modified types to match those used by UBL so
                   that instance documents could be made conformant to both
                   standards.
                </doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
schemaLocation="14817-Documentation-2.0b.xsd"/>
   <!-- ===== Type Definitions ===== -->
   <xs:complexType name="AmountType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>amount</doc:name>
             <doc:definition>A numeric quantification of a monetary value expressed in
monetary units, such as dollars and cents.</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:decimal">
             <xs:attribute name="currencyID" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The currency of the amount as defined
by the 3-letter alphabetic code from UNECE Rec 9.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="currencyCodeListVersionID" type="xs:normalizedString"
use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The VersionID of the UN/ECE Rec9 code
list.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
          </xs:extension>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="BinaryObjectType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>binary-object</doc:name>
             <doc:definition>A binary value used to transport complex data.</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:base64Binary">
             <xs:attribute name="format" type="xs:string" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The format of the binary content.</
xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="mimeCode" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The mime type of the binary object.</
xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="encodingCode" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">Specifies decoding algorithm of the
```

```
binary object.</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="characterSetCode" type="xs:normalizedString"
use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The character set of the binary object
if the mime
                            type is text.</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="uri" type="xs:anyURI" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The Uniform Resource Identifier that
identifies where
                            the binary object is located.</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="filename" type="xs:string" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The filename of the binary object.</
xs:documentation>
                </xs:annotation>
            </xs:attribute>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="CodeType">
      <xs:annotation>
         <xs:documentation xml:lang="en">
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>code</doc:name>
            <doc:definition>An alphanumeric character or symbol (or a string of characters
or symbols) that represents a specific meaning, e.g., "T" for "True" and "F" for "False".
The explicit Representation Term applicable to a data element using this Representation
Term term should be specified in its Representation Term meta-attribute.</doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:simpleContent>
         <xs:extension base="xs:normalizedString">
            <xs:attribute name="listID" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The identification of a list of
codes.</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="listAgencyID" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">An agency that maintains one or more
lists of codes.</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="listAgencyName" type="xs:string" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">Name of the agency that maintains the
list of codes.</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="listName" type="xs:string" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The name of a list of codes.</
xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="listVersionID" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The version of the list of codes.</
xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="name" type="xs:string" use="optional">
```

```
                <xs:annotation>
                   <xs:documentation xml:lang="en">The textual equivalent of the code
content component.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="languageID" type="xs:language" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The identifier of the language used in
the code name.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="listURI" type="xs:anyURI" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The Uniform Resource Identifier that
identifies where
                            the code list is located.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="listSchemeURI" type="xs:anyURI" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The Uniform Resource Identifier that
identifies where the code list scheme is located.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
          </xs:extension>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="DateTimeType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>date-time</doc:name>
             <doc:definition>A specific point in the time continuum adequate to identify
both a specific calendar day and a time within the day.</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:dateTime"/>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="DateType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>date</doc:name>
             <doc:definition>A specific calendar day</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:date"/>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="DurationType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>duration</doc:name>
             <doc:definition>A duration of time.</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:duration"/>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="GraphicType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>graphic</doc:name>
             <doc:definition>A graphical item, such as a map or icon. The explicit format
applicable to a data element using this representation term should be specified in the
```

```
meta-attribute (e.g., jpeg, mpeg, gif).</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:complexContent>
          <xs:extension base="BinaryObjectType"/>
       </xs:complexContent>
   </xs:complexType>
   <xs:complexType name="IdentifierType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>identifier</doc:name>
             <doc:definition>A value used to uniquely identify a Data Concept Instance of
an object class.</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:normalizedString">
             <xs:attribute name="schemeID" type="xs:normalizedString" use="optional">
             <xs:annotation>
                    <xs:documentation xml:lang="en">The identification of the identification
scheme.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
             <xs:attribute name="schemeName" type="xs:string" use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The name of the identification
scheme.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
             <xs:attribute name="schemeAgencyID" type="xs:normalizedString"
use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The identification of the agency that
maintains the identification scheme.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
             <xs:attribute name="schemeAgencyName" type="xs:string" use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The name of the agency that maintains
the identification scheme.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
             <xs:attribute name="schemeVersionID" type="xs:normalizedString"
use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The version of the identification
scheme.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
             <xs:attribute name="schemeDataURI" type="xs:anyURI" use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The Uniform Resource Identifier that
identifies where the identification scheme data is located.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
             <xs:attribute name="schemeURI" type="xs:anyURI" use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The Uniform Resource Identifier that
identifies where the identification scheme is located.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
          </xs:extension>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="IndicatorType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>indicator</doc:name>
             <doc:definition>A list of two mutually exclusive Boolean values that express
```

```
the only possible states of a property.</doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:boolean"/>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="MeasureType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>measure</doc:name>
             <doc:definition>A numeric value representing a measurement along with the
specified unit of measure.</doc:definition>
             <doc:dataType>decimal</doc:dataType>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:decimal">
             <xs:attribute name="unitCode" type="xs:normalizedString" use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The units used for the measurement.</
xs:documentation>
                 </xs:annotation>
             </xs:attribute>
             <xs:attribute name="unitCodeListVersionID" type="xs:normalizedString"
use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">The version of units of measure.</
xs:documentation>
                 </xs:annotation>
             </xs:attribute>
          </xs:extension>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="NameType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>name</doc:name>
             <doc:definition>A short textual language identifier for an entity.</
doc:definition>
             <doc:dataType>string</doc:dataType>
          </xs:documentation>
       </xs:annotation>
       <xs:complexContent>
          <xs:extension base="TextType"/>
       </xs:complexContent>
   </xs:complexType>
   <xs:complexType name="NumericType">
       <xs:annotation>
          <xs:documentation xml:lang="en">
             <doc:dataConceptType>Representation Term</doc:dataConceptType>
             <doc:name>numeric</doc:name>
             <doc:definition>A non-computational numeric string used to designate an item,
e.g., a serial number, telephone number, street number, apartment number, or social
security number. </doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:simpleContent>
          <xs:extension base="xs:decimal">
             <xs:attribute name="format" type="xs:string" use="optional">
                 <xs:annotation>
                    <xs:documentation xml:lang="en">Whether the number is an integer,
decimal, real number or percentage.</xs:documentation>
                 </xs:annotation>
             </xs:attribute>
          </xs:extension>
       </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="PercentType">
```

```
      <xs:annotation>
         <xs:documentation xml:lang="en">
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>percent</doc:name>
            <doc:definition>A ratio of two quantities expressed in numeric format as a
decimal number multiplied by 100.</doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:complexContent>
         <xs:extension base="NumericType"/>
      </xs:complexContent>
   </xs:complexType>
   <xs:complexType name="PictureType">
      <xs:annotation>
         <xs:documentation xml:lang="en">
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>picture</doc:name>
            <doc:definition>A pictorial item, such as a diagram or picture. The explicit
format applicable to a data element should be specified as a meta-attribute (e.g., jpeg,
mpeg, gif). </doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:complexContent>
         <xs:extension base="BinaryObjectType"/>
      </xs:complexContent>
   </xs:complexType>
   <xs:complexType name="QuantityType">
      <xs:annotation>
         <xs:documentation xml:lang="en">
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>quantity</doc:name>
            <doc:definition>A non-monetary numeric value subject to computational
manipulations.</doc:definition>
         </xs:documentation>
      </xs:annotation>
          <xs:simpleContent>
         <xs:extension base="xs:decimal">
            <xs:attribute name="unitCode" type="xs:normalizedString" use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The unit of the quantity</
xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="unitCodeListID" type="xs:normalizedString"
use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The quantity unit code list.</
xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="unitCodeListAgencyID" type="xs:normalizedString"
use="optional">
                <xs:annotation>
                   <xs:documentation xml:lang="en">The identification of the agency that
maintains the quantity unit code list</xs:documentation>
                </xs:annotation>
            </xs:attribute>
            <xs:attribute name="unitCodeListAgencyName" type="xs:string" use="optional">
                      <xs:annotation>
                   <xs:documentation xml:lang="en">The name of the agency which maintains
the quantity unit code list.</xs:documentation>
                </xs:annotation>
             </xs:attribute>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name="RateType">
      <xs:annotation>
         <xs:documentation xml:lang="en">
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>rate</doc:name>
```

```
            <doc:definition>A numeric unit of measure expressing the ratio of a quantity
to another quantity, for example, “miles per hour,” “gallons per hour,” and “dollars per
day.” </doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:simpleContent>
         <xs:extension base=”NumericType”/>
      </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name=”SoundType”>
      <xs:annotation>
         <xs:documentation xml:lang=”en”>
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>sound</doc:name>
            <doc:definition>An audio sequence with explicit beginning and end. The
explicit format applicable to a data element should be specified in the meta-attribute,
e.g., “wav.” </doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:simpleContent>
         <xs:extension base=”BinaryObjectType”/>
      </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name=”TextType”>
      <xs:annotation>
         <xs:documentation xml:lang=”en”>
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>text</doc:name>
            <doc:definition>An alphanumeric string (formatted or unformatted), e.g., a
street name, or the contents of a document, message, or other file.</doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:simpleContent>
         <xs:extension base=”xs:string”>
            <xs:attribute name=”languageID” type=”xs:language” use=”optional”>
               <xs:annotation>
                  <xs:documentation xml:lang=”en”>The identifier of the language used in
the content component.</xs:documentation>
               </xs:annotation>
            </xs:attribute>
            <xs:attribute name=”languageLocaleID” type=”xs:normalizedString”
use=”optional”>
               <xs:annotation>
                  <xs:documentation xml:lang=”en”>The identification of the locale of the
language.</xs:documentation>
               </xs:annotation>
            </xs:attribute>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name=”TimeType”>
      <xs:annotation>
         <xs:documentation xml:lang=”en”>
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>time</doc:name>
            <doc:definition>A specific time during a day</doc:definition>
         </xs:documentation>
      </xs:annotation>
      <xs:simpleContent>
         <xs:extension base=”xs:time”/>
      </xs:simpleContent>
   </xs:complexType>
   <xs:complexType name=”ValueType”>
      <xs:annotation>
         <xs:documentation xml:lang=”en”>
            <doc:dataConceptType>Representation Term</doc:dataConceptType>
            <doc:name>value</doc:name>
            <doc:definition>Numeric information that is assigned or is determined by
calculation, counting, or sequencing. It does not require a unit of quantity or unit of
measure.</doc:definition>
         </xs:documentation>
```

```
        </xs:annotation>
        <xs:complexContent>
           <xs:extension base=”NumericType”/>
        </xs:complexContent>
    </xs:complexType>
    <xs:complexType name=”VideoType”>
        <xs:annotation>
           <xs:documentation xml:lang=”en”>
              <doc:dataConceptType>Representation Term</doc:dataConceptType>
              <doc:name>video</doc:name>
              <doc:definition>A changing visual item, such as a motion picture. The explicit
format applicable to a data element should be specified in its meta-attribute (e.g.,
mov).</doc:definition>
           </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
           <xs:extension base=”BinaryObjectType”/>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

Annex E
(normative)

Common basic components schema

E.1 General

This annex provides the formal definition of the ISO/TC 204 common basic components schema that should be used as a base by all ISO/TC 204 XML schema standards.

Standards referencing this schema should only modify it to add "include" statements. E.2 provides the standard base schema and E.3 provides a modified schema for the example message set by adding one "include" statement in bold. E.4 provides the included schema that was added as a reference in E.3.

E.2 Standard common basic components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:udt="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
elementFormDefault="qualified"                    attributeFormDefault="unqualified"
version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
      <xs:documentation>
         <doc:general>
            <doc:schemaName>Common Basic Components</doc:schemaName>
            <doc:description>
                  Properties that are common to many ITS messages, plus
                  included files containing all other properties referenced
                  by the message set.
               </doc:description>
            <doc:status>Draft</doc:status>
            <doc:standardTitle>
                  Using XML in ITS standards, data registries, and data
                  dictionaries
               </doc:standardTitle>
            <doc:standardNumber>ISO 24531</doc:standardNumber>
            <doc:validatedBy>Oxygen</doc:validatedBy>
            <doc:releasePackage>
                  http://www.trevilon.com/iso/24531/
               </doc:releasePackage>
         </doc:general>
         <doc:changeHistory>
            <doc:change>
                <doc:version>2.0b</doc:version>
                <doc:date>2011-10-13</doc:date>
                <doc:description>Original version</doc:description>
            </doc:change>
         </doc:changeHistory>
         <doc:contact>
            <doc:name>Ken Vaughn</doc:name>
            <doc:email>kvaughn@trevilon.com</doc:email>
         </doc:contact>
      </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
```

```
schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
schemaLocation="24531-UnqualifiedDataTypes-2.0b.xsd"/>
   <!-- ===== Includes ===== -->
   <!-- Add Message Set Specific Files Here -->
   <!-- ===== Basic Element Definitions ===== -->
   <xs:element name="description" type="udt:TextType"/>
   <xs:element name="id" type="udt:IdentifierType"/>
   <xs:element name="latitudeMeasure" type="udt:MeasureType"/>
   <xs:element name="longitudeMeasure" type="udt:MeasureType"/>
   <xs:element name="name" type="udt:NameType"/>
   <xs:element name="profileID" type="udt:IdentifierType"/>
   <xs:element name="reason" type="udt:TextType"/>
   <xs:element name="reasonCode" type="udt:CodeType"/>
   <xs:element name="type" type="udt:TextType"/>
   <xs:element name="versionID" type="udt:IdentifierType"/>
   <xs:element name="duration" type="udt:DurationType"/>
   <xs:element name="radiusQuantity" type="udt:QuantityType"/>
   <xs:element name="startDateTime" type="udt:DateTimeType"/>
   <xs:element name="typeCode" type="udt:CodeType"/>
   <xs:element name="uri" type="udt:IdentifierType"/>
</xs:schema>
```

E.3 Modified common basic components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:udt="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
elementFormDefault="qualified"                    attributeFormDefault="unqualified"
version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Common Basic Components</doc:schemaName>
             <doc:description>
                    Properties that are common to many ITS messages, plus
                    included files containing all other properties referenced
                    by the message set.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                 <doc:version>2.0b</doc:version>
                 <doc:date>2011-10-13</doc:date>
                 <doc:description>Original version</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
schemaLocation="14817-Documentation-2.0b.xsd"/>
```

```
   <xs:import namespace="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
schemaLocation="24531-UnqualifiedDataTypes-2.0b.xsd"/>
   <!-- ===== Includes ===== -->
   <!-- Add Message Set Specific Files Here -->
   <xs:include schemaLocation="24531-ExampleBasicComponents-2.0b.xsd"/>
   <!-- ===== Basic Element Definitions ===== -->
   <xs:element name="description" type="udt:TextType"/>
   <xs:element name="id" type="udt:IdentifierType"/>
   <xs:element name="latitudeMeasure" type="udt:MeasureType"/>
   <xs:element name="longitudeMeasure" type="udt:MeasureType"/>
   <xs:element name="name" type="udt:NameType"/>
   <xs:element name="profileID" type="udt:IdentifierType"/>
   <xs:element name="reason" type="udt:TextType"/>
   <xs:element name="reasonCode" type="udt:CodeType"/>
   <xs:element name="type" type="udt:TextType"/>
   <xs:element name="versionID" type="udt:IdentifierType"/>
   <xs:element name="duration" type="udt:DurationType"/>
   <xs:element name="radiusQuantity" type="udt:QuantityType"/>
   <xs:element name="startDateTime" type="udt:DateTimeType"/>
   <xs:element name="typeCode" type="udt:CodeType"/>
   <xs:element name="uri" type="udt:IdentifierType"/>
</xs:schema>
```

E.4 Included basic components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:udt="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   elementFormDefault="qualified"    attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Example Basic Components</doc:schemaName>
             <doc:description>
                    Properties that are specific to the 24531 example message set.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
              <doc:change>
                  <doc:version>2.0b</doc:version>
                  <doc:date>2011-10-13</doc:date>
                  <doc:description>Original version</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import              namespace="urn:iso:std:iso:14817:tech:Documentation-2"
schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
schemaLocation="24531-UnqualifiedDataTypes-2.0b.xsd"/>
   <!-- ===== Basic Element Definitions ===== -->
```

```
   <xs:element name="trafficImpactCode" type="udt:CodeType"/>
</xs:schema>
```

Annex F
(normative)

Common aggregate components schema

F.1 General

This annex provides the formal definition of the ISO TC/204 common aggregate components schema that should be used as a base by all ISO/TC 204 XML schema standards.

Standards referencing this schema should only modify it to add "include" statements. F.2 provides the standard base schema and F.3 provides a modified schema for the example message set by adding two "include" statements in bold. F.4 provides the included schemas, one of which is included in the standard common aggregate components schema, and two that are included as a part of this example.

F.2 Standard common aggregate components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Common Aggregate Components</doc:schemaName>
             <doc:description>
                    The definition of the standard aggregate components common
                    to most ITS deployments plus includes for additional
                    aggregate components required by the subject message set
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                 <doc:version>2.0b</doc:version>
                 <doc:date>2011-10-13</doc:date>
                 <doc:description>Original version; added to more closely
                    mimic UBL structure.</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
```

```
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
   <!-- ===== Includes ===== -->
   <xs:include schemaLocation="24531-AgencyAggregateComponents-2.0b.xsd"/>
   <!-- Add Message Set Specific Files Here -->
   <!-- ===== Aggregate Element Definitions ===== -->
   <!-- ===== Aggregate Type Definitions ===== -->
</xs:schema>
```

F.3 Modified common aggregate components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Common Aggregate Components</doc:schemaName>
             <doc:description>
                    The definition of the standard aggregate components common
                    to most ITS deployments plus includes for additional
                    aggregate components required by the subject message set
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                 <doc:version>2.0b</doc:version>
                 <doc:date>2011-10-13</doc:date>
                 <doc:description>Original version; added to more closely
                    mimic UBL structure.</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
   <!-- ===== Includes ===== -->
   <xs:include schemaLocation="24531-AgencyAggregateComponents-2.0b.xsd"/>
   <!-- Add Message Set Specific Files Here -->
   <xs:include schemaLocation="24531-LocationAggregateComponents-2.0b.xsd"/>
   <xs:include schemaLocation="24531-IncidentAggregateComponents-2.0b.xsd"/>
   <!-- ===== Aggregate Element Definitions ===== -->
   <!-- ===== Aggregate Type Definitions ===== -->
</xs:schema>
```

F.4 Agency aggregate components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:udt="urn:iso:std:iso:24531:tech:UnqualifiedDataTypes-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Common Extension Components</doc:schemaName>
             <doc:description>
                    The definition of the standard components to extend the standard.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>XML Spy</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                 <doc:version>2.0b</doc:version>
                 <doc:date>2011-08-07</doc:date>
                 <doc:description>Original version; added file to prevent circular refer-
ences</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
       schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
       schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
   <!-- ===== Aggregate Element Definitions ===== -->
   <xs:element name="Agency" type="AgencyStructure"/>
   <!-- ===== Aggregate Type Definitions ===== -->
   <xs:complexType name="AgencyStructure">
       <xs:sequence>
          <xs:element ref="cbc:id" minOccurs="0">
             <xs:annotation>
                 <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Agency.id:identifier</doc:name>
                    <doc:definition>A system-wide unique identifier for the agency</
doc:definition>
                    <doc:multiplicity>1</doc:multiplicity>
                 </xs:documentation>
             </xs:annotation>
          </xs:element>
          <xs:element minOccurs="0" ref="cbc:name">
             <xs:annotation>
                 <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Agency.name:name</doc:name>
                    <doc:definition>A textual name for the agency</doc:definition>
                    <doc:multiplicity>1</doc:multiplicity>
```

```
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element minOccurs="0" ref="cbc:uri">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Agency.uri:identifier</doc:name>
                    <doc:definition>A universal resource identifier for the agency</
doc:definition>
                    <doc:multiplicity>1</doc:multiplicity>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

F.5 Location aggregate components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
  xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
  xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
  targetNamespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
  elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
  <!-- ===== Header ===== -->
  <xs:annotation>
    <xs:documentation>
      <doc:general>
        <doc:schemaName>Location Aggregate Components</doc:schemaName>
        <doc:description>
            Aggregate Components for the Incident object class.
          </doc:description>
        <doc:status>Draft</doc:status>
        <doc:standardTitle>
            Using XML in ITS standards, data registries, and data
            dictionaries
          </doc:standardTitle>
        <doc:standardNumber>ISO 24531</doc:standardNumber>
        <doc:validatedBy>Oxygen</doc:validatedBy>
        <doc:releasePackage>
            http://www.trevilon.com/iso/24531/index.html
          </doc:releasePackage>
      </doc:general>
      <doc:changeHistory>
        <doc:change>
          <doc:version>2.0a</doc:version>
          <doc:date>2011-08-07</doc:date>
          <doc:description>Original version</doc:description>
        </doc:change>
      </doc:changeHistory>
      <doc:contact>
        <doc:name>Ken Vaughn</doc:name>
        <doc:email>kvaughn@trevilon.com</doc:email>
      </doc:contact>
    </xs:documentation>
  </xs:annotation>
  <!-- ===== Imports ===== -->
  <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
    schemaLocation="../common/14817-Documentation-2.0b.xsd"/>
  <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
    schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
  <!-- ===== Aggregate Element Definitions ===== -->
  <xs:element name="centerGeoLocation" type="GeoLocationStructure"/>
  <xs:element name="circularAreaLocation" type="CircularAreaLocationStructure"/>
  <xs:element name="geoLocation" type="GeoLocationStructure"/>
  <!-- ===== Aggregate Type Definitions ===== -->
  <xs:complexType name="CircularAreaLocationStructure">
    <xs:annotation>
```

```
        <xs:documentation>
           <doc:dataConceptType>Data Frame</doc:dataConceptType>
           <doc:name>CircularAreaLocation
              </doc:name>
           <doc:definition>An area defined by a logical circle.
              </doc:definition>
        </xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element ref="cbc:radiusQuantity">
           <xs:annotation>
               <xs:documentation>
                  <doc:dataConceptType>Data Element</doc:dataConceptType>
                  <doc:name>CircularAreaLocation.radius:quantity
                      </doc:name>
                  <doc:definition>The radius of the logical circle.
                      </doc:definition>
                  <doc:unit>meters, unless otherwise specified
                      </doc:unit>
               </xs:documentation>
           </xs:annotation>
        </xs:element>
        <xs:element ref="centerGeoLocation">
           <xs:annotation>
               <xs:documentation>
                  <doc:dataConceptType>Association</doc:dataConceptType>
                  <doc:name>
                          CircularAreaLocation.center:GeoLocation
                      </doc:name>
                  <doc:definition>The center of the logical circle.
                      </doc:definition>
                  <doc:multiplicity>1</doc:multiplicity>
               </xs:documentation>
           </xs:annotation>
        </xs:element>
     </xs:sequence>
  </xs:complexType>
  <xs:complexType name="GeoLocationStructure">
     <xs:annotation>
        <xs:documentation>
           <doc:dataConceptType>Data Frame</doc:dataConceptType>
           <doc:name>GeoLocation</doc:name>
           <doc:definition>A point location defined by a latitude and a
                  longitude.
              </doc:definition>
        </xs:documentation>
     </xs:annotation>
     <xs:sequence>
        <xs:element ref="cbc:latitudeMeasure">
           <xs:annotation>
               <xs:documentation>
                  <doc:dataConceptType>Data Element</doc:dataConceptType>
                  <doc:name>GeoLocation.latitude:quantity</doc:name>
                  <doc:definition>The latitude of the point location.
                      </doc:definition>
                  <doc:unit>Degrees</doc:unit>
               </xs:documentation>
           </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:longitudeMeasure">
           <xs:annotation>
               <xs:documentation>
                  <doc:dataConceptType>Data Element</doc:dataConceptType>
                  <doc:name>GeoLocation.longitude:quantity</doc:name>
                  <doc:definition>The longitude of the point location.
                      </doc:definition>
                  <doc:unit>Degrees</doc:unit>
               </xs:documentation>
           </xs:annotation>
        </xs:element>
     </xs:sequence>
```

```
   </xs:complexType>
</xs:schema>
```

F.6 Incident aggregate components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Incident Aggregate Components</doc:schemaName>
             <doc:description>
                    Aggregate Components for the Incident object class.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/index.html
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                 <doc:version>2.0a</doc:version>
                 <doc:date>2011-08-07</doc:date>
                 <doc:description>Original version</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
       schemaLocation="../common/14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
       schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
   <!-- ===== Includes ===== -->
   <xs:include schemaLocation="24531-LocationAggregateComponents-2.0b.xsd"/>
   <!-- ===== Aggregate Element Definitions ===== -->
   <xs:element name="incident" type="IncidentStructure"/>
   <xs:element name="interestAreaLocation" type="CircularAreaLocationStructure"/>
   <!-- ===== Aggregate Type Definitions ===== -->
   <xs:complexType name="IncidentStructure">
       <xs:annotation>
          <xs:documentation>
             <doc:dataConceptType>Data Frame</doc:dataConceptType>
             <doc:name>Incident</doc:name>
             <doc:definition>A condition that is affecting the transportation
                    system.
                </doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:sequence>
          <xs:element ref="cbc:id">
             <xs:annotation>
                 <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Incident.id:identifier</doc:name>
```

```
                <doc:definition>An identifier that can be used to
                        uniquely identify the incident in the host system.
                    </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:typeCode">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Incident.type:code</doc:name>
                    <doc:definition>A code indicating the type of incident.
                    </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:trafficImpactCode">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Incident.trafficImpact:code</doc:name>
                    <doc:definition>A code indicating the severity of the
                        incident.
                    </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:startDateTime">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Incident.start:dateTime</doc:name>
                    <doc:definition>The date and time at which the incident
                        began to affect traffic.
                    </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:duration">
            <xs:annotation>
                    <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Incident.duration:duration</doc:name>
                    <doc:definition>The expected duration of the incident as
                        measured from the start time.
                    </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="geoLocation">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Association</doc:dataConceptType>
                    <doc:name>Incident.geoLocation:GeoLocation</doc:name>
                    <doc:definition>The location of the incident.
                    </doc:definition>
                    <doc:multiplicity>1</doc:multiplicity>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
      </xs:sequence>
   </xs:complexType>
</xs:schema>
```

Annex G
(normative)

Common extension components schema

G.1 General

This annex provides the formal definition of the ISO/TC 204 common extension components schema that should be referenced by all ISO/TC 204 XML schema standards.

G.2 Common extension components schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonExtensionComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:cac="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonExtensionComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Common Extension Components</doc:schemaName>
             <doc:description>
                    The definition of the standard components to extend the standard.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>XML Spy</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                <doc:version>2.0b</doc:version>
                <doc:date>2011-08-07</doc:date>
                <doc:description>Original version; added file to prevent circular refer-
ences</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
       schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
       schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
       schemaLocation="24531-CommonAggregateComponents-2.0b.xsd"/>
   <!-- ===== Includes ===== -->
   <xs:include schemaLocation="ExtensionContentDataType.xsd"/>
```

```
<!-- ===== Aggregate Element Definitions ===== -->
<xs:element name="extension" type="ExtensionStructure"/>
<xs:element name="extensions" type="ExtensionsStructure"/>
<!-- ===== Aggregate Type Definitions ===== -->
<xs:complexType name="ExtensionsStructure">
    <xs:sequence>
       <xs:element ref="extension" minOccurs="0" maxOccurs="unbounded">
          <xs:annotation>
              <xs:documentation>
                 <doc:dataConceptType>Association</doc:dataConceptType>
                 <doc:name>Extensions.extension:Extension
                     </doc:name>
                 <doc:definition>An extension included in the
                         message.
                     </doc:definition>
                 <doc:multiplicity>0..*</doc:multiplicity>
              </xs:documentation>
          </xs:annotation>
       </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="ExtensionStructure">
    <xs:sequence>
       <xs:element ref="cbc:id" minOccurs="0">
          <xs:annotation>
              <xs:documentation>
                 <doc:dataConceptType>Data Element</doc:dataConceptType>
                 <doc:name>Extension.id:id</doc:name>
                 <doc:definition>A unique identifier for the extension.
                     </doc:definition>
              </xs:documentation>
          </xs:annotation>
       </xs:element>
       <xs:element ref="cbc:name" minOccurs="0">
          <xs:annotation>
              <xs:documentation>
                 <doc:dataConceptType>Data Element</doc:dataConceptType>
                 <doc:name>Extension.name:text</doc:name>
                 <doc:definition>A unique name for the extension.
                     </doc:definition>
              </xs:documentation>
          </xs:annotation>
       </xs:element>
       <xs:element ref="cbc:versionID" minOccurs="0">
          <xs:annotation>
              <xs:documentation>
                 <doc:dataConceptType>Data Element</doc:dataConceptType>
                 <doc:name>Extension.versionID:id</doc:name>
                 <doc:definition>The earliest version of the extension
                         that defines all of the elements that might be
                         encountered in the current instance.
                     </doc:definition>
              </xs:documentation>
          </xs:annotation>
       </xs:element>
       <xs:element ref="cbc:uri" minOccurs="0">
          <xs:annotation>
              <xs:documentation>
                 <doc:dataConceptType>Data Element</doc:dataConceptType>
                 <doc:name>Extension.extensionURI:id</doc:name>
                 <doc:definition>A universal resource identifier for the
                         extension.
                     </doc:definition>
              </xs:documentation>
          </xs:annotation>
       </xs:element>
       <xs:element ref="cac:agency" minOccurs="0">
          <xs:annotation>
              <xs:documentation>
                 <doc:dataConceptType>Data Element</doc:dataConceptType>
                 <doc:name>Extension.agencyID:id</doc:name>
```

```
                    <doc:definition>A unique identifier for the agency.
                        </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:reasonCode" minOccurs="0">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Extension.reasonCode:code</doc:name>
                    <doc:definition>A code that desribes why the extension
                            has been included.
                        </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="cbc:reason" minOccurs="0">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Extension.reason:text</doc:name>
                    <doc:definition>A natural text description explaining
                            why the extension is included.
                        </doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element ref="extensionContent">
            <xs:annotation>
                <xs:documentation>
                    <doc:dataConceptType>Data Frame</doc:dataConceptType>
                    <doc:descriptiveName>Extensions</doc:descriptiveName>
                    <doc:definition>The content of the local or custom
                            extension.</doc:definition>
                </xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

Annex H
(normative)

Extension content data type schema

H.1 General

This annex provides the formal definition of the ISO/TC 204 extension content data type schema that should be referenced by all ISO/TC 204 XML schema standards.

H.2 Extension content data type schema

```
 <?xml version="1.0" encoding="UTF-8"?>
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonExtensionComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonExtensionComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <!-- Note: This file may be changed in a production system, but only by importing addi-
tional namespaces. -->
   <xs:annotation>
      <xs:documentation>
         <doc:general>
            <doc:schemaName>Extension Content Data Type</doc:schemaName>
            <doc:description>
                  The definition of the standard mechanism to extend
                  standardized schemas.
               </doc:description>
            <doc:status>Sample</doc:status>
            <doc:standardTitle>
                  Using XML in ITS standards, data registries, and data
                  dictionaries
               </doc:standardTitle>
            <doc:standardNumber>ISO 24531</doc:standardNumber>
            <doc:validatedBy>XML Spy</doc:validatedBy>
            <doc:releasePackage>
                  http://www.trevilon.com/iso/24531/
               </doc:releasePackage>
         </doc:general>
         <doc:changeHistory>
            <doc:change>
               <doc:version>2.0a</doc:version>
               <doc:date>2011-08-07</doc:date>
               <doc:description>Original version</doc:description>
            </doc:change>
            <doc:change>
                <doc:version>2.0b</doc:version>
                <doc:date>2011-10-13</doc:date>
                <doc:description>Moved all components to Common Extension Components file
so that extensions could use common components and message set components without circular
references.</doc:description>
            </doc:change>
         </doc:changeHistory>
         <doc:contact>
            <doc:name>Ken Vaughn</doc:name>
            <doc:email>kvaughn@trevilon.com</doc:email>
         </doc:contact>
      </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
```

```
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
schemaLocation="14817-Documentation-2.0b.xsd"/>
   <!-- ===== Extension Definition ===== -->
   <xs:complexType name="ExtensionContentType">
       <xs:sequence>
          <xs:any namespace="##any" processContents="lax" minOccurs="0">
             <xs:annotation>
                 <xs:documentation>The definition of the extension content.</
xs:documentation>
             </xs:annotation>
          </xs:any>
       </xs:sequence>
   </xs:complexType>
   <!-- ===== Basic Element Definitions ===== -->
   <xs:element name="extensionContent" type="ExtensionContentType"/>
</xs:schema>
```

Annex I
(normative)

Common message components schema

I.1 General

This annex provides the common message components schema that should be used by all ISO/TC 204 messages.

I.2 Message set components schema

```
 <?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2008 rel. 2 sp1 (http://www.altova.com) by Kenneth Vaughn
(Trevilon Corporation) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:CommonMessageComponents-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:ext="urn:iso:std:iso:24531:tech:CommonExtensionComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:CommonMessageComponents-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
      <xs:documentation>
         <doc:general>
            <doc:schemaName>Common Message Components</doc:schemaName>
            <doc:description>
                  The definition of the standard components to extend the standard.
               </doc:description>
            <doc:status>Draft</doc:status>
            <doc:standardTitle>
                  Using XML in ITS standards, data registries, and data
                  dictionaries
               </doc:standardTitle>
            <doc:standardNumber>ISO 24531</doc:standardNumber>
            <doc:validatedBy>XML Spy</doc:validatedBy>
            <doc:releasePackage>
                  http://www.trevilon.com/iso/24531/
               </doc:releasePackage>
         </doc:general>
         <doc:changeHistory>
            <doc:change>
               <doc:version>2.0b</doc:version>
               <doc:date>2011-08-07</doc:date>
               <doc:description>Original version; added file to prevent circular refer-
ences</doc:description>
            </doc:change>
         </doc:changeHistory>
         <doc:contact>
            <doc:name>Ken Vaughn</doc:name>
            <doc:email>kvaughn@trevilon.com</doc:email>
         </doc:contact>
      </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
       schemaLocation="14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
       schemaLocation="24531-CommonBasicComponents-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonExtensionComponents-2"
       schemaLocation="24531-CommonExtensionComponents-2.0b.xsd"/>
```

```
   <!-- ===== Aggregate Element Definitions ===== -->
   <xs:element name="message" type="MessageStructure"/>
   <!-- ===== Aggregate Type Definitions ===== -->
   <xs:complexType name="MessageStructure">
       <xs:annotation>
          <xs:documentation>
             <doc:dataConceptType>Message</doc:dataConceptType>
             <doc:name>Message():message</doc:name>
             <doc:definition>A generic message container; used as the
                    generalized class for all messages.
                </doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:sequence>
          <xs:element ref="ext:extensions" minOccurs="0">
             <xs:annotation>
                 <xs:documentation>
                    <doc:dataConceptType>Association</doc:dataConceptType>
                    <doc:name>Message.extensions:Extensions
                        </doc:name>
                    <doc:definition>The extensions included in the
                            message.
                        </doc:definition>
                    <doc:multiplicity>0..1</doc:multiplicity>
                 </xs:documentation>
             </xs:annotation>
          </xs:element>
          <xs:element ref="cbc:versionID">
             <xs:annotation>
                 <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Message.versionID:id</doc:name>
                    <doc:definition>The earliest version of the message that
                            defines all of the elements that might be encountered
                            in the current instance.
                        </doc:definition>
                 </xs:documentation>
             </xs:annotation>
          </xs:element>
          <xs:element ref="cbc:profileID" minOccurs="0">
             <xs:annotation>
                 <xs:documentation>
                    <doc:dataConceptType>Data Element</doc:dataConceptType>
                    <doc:name>Message.profileID:id</doc:name>
                    <doc:definition>Identifies a user-defined customized
                            profile of the message being used.
                        </doc:definition>
                 </xs:documentation>
             </xs:annotation>
          </xs:element>
       </xs:sequence>
   </xs:complexType>
</xs:schema>
```

Annex J
(informative)

Example message exchange: request message schema

J.1 General

This annex provides the message schema for the request message identified in the example message exchange defined in Annex C.

J.2 Request message schema

```
 <?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:IncidentListRequest-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:cac="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   xmlns:msg="urn:iso:std:iso:24531:tech:CommonMessageComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:IncidentListRequest-2"
   elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0a">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Incident List Request</doc:schemaName>
             <doc:description>
                    Sample request for obtaining a list of incidents in a region.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                    Using XML in ITS standards, data registries, and data
                    dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                    http://www.trevilon.com/iso/24531/index.html
                </doc:releasePackage>
          </doc:general>
          <doc:changeHistory>
             <doc:change>
                <doc:version>2.0a</doc:version>
                <doc:date>2011-08-07</doc:date>
                <doc:description>Original version</doc:description>
             </doc:change>
             <doc:change>
                 <doc:version>2.0b</doc:version>
                 <doc:date>2011-11-01</doc:date>
                 <doc:description>Updated imports</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
             <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
       schemaLocation="../common/14817-Documentation-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
```

```
        schemaLocation="../common/24531-CommonBasicComponents-2.0b.xsd"/>
    <xs:import namespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
        schemaLocation="../common/24531-CommonAggregateComponents-2.0b.xsd"/>
    <xs:import namespace="urn:iso:std:iso:24531:tech:CommonMessageComponents-2"
        schemaLocation="../common/24531-CommonMessageComponents-2.0b.xsd"/>
    <!-- ===== Aggregate Element Definitions ===== -->
    <xs:element name="incidentListRequest" type="IncidentListRequestStructure"/>
    <!-- ===== Aggregate Type Definitions ===== -->
    <xs:complexType name="IncidentListRequestStructure">
        <xs:annotation>
           <xs:documentation>
              <doc:dataConceptType>Message</doc:dataConceptType>
              <doc:name>IncidentListRequestStructure():message
                 </doc:name>
              <doc:definition>A request for a listing of all known active
                     incidents within the interestArea.
                 </doc:definition>
           </xs:documentation>
        </xs:annotation>
        <xs:complexContent>
           <xs:extension base="msg:MessageStructure">
              <xs:sequence>
                  <xs:element ref="cac:interestAreaLocation">
                     <xs:annotation>
                        <xs:documentation>
                           <doc:dataConceptType>Association</doc:dataConceptType>
                           <doc:name>Message.interestAreaLocation
                                 </doc:name>
                           <doc:definition>The area in which to search for incidents
                                 </doc:definition>
                           <doc:multiplicity>1</doc:multiplicity>
                        </xs:documentation>
                     </xs:annotation>
                  </xs:element>
              </xs:sequence>
           </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

Annex K
(informative)

Example message exchange: response message schema

K.1 General

This annex provides the message schema for the response message identified in the example message exchange defined in Annex C.

K.2 Response message schema

```
 <?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns="urn:iso:std:iso:24531:tech:IncidentList-2"
   xmlns:doc="urn:iso:std:iso:14817:tech:Documentation-2"
   xmlns:cbc="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
   xmlns:cac="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
   xmlns:msg="urn:iso:std:iso:24531:tech:CommonMessageComponents-2"
   targetNamespace="urn:iso:std:iso:24531:tech:IncidentList-2"
elementFormDefault="qualified"
   attributeFormDefault="unqualified" version="2.0b">
   <!-- ===== Header ===== -->
   <xs:annotation>
       <xs:documentation>
          <doc:general>
             <doc:schemaName>Incident List</doc:schemaName>
                         <doc:description>
                   A list of incidents in a region.
                </doc:description>
             <doc:status>Draft</doc:status>
             <doc:standardTitle>
                   Using XML in ITS standards, data registries, and data
                   dictionaries
                </doc:standardTitle>
             <doc:standardNumber>ISO 24531</doc:standardNumber>
             <doc:validatedBy>Oxygen</doc:validatedBy>
             <doc:releasePackage>
                   http://www.trevilon.com/iso/24531/index.html
                </doc:releasePackage>
          </doc:general>
                   <doc:changeHistory>
             <doc:change>
                <doc:version>2.0a</doc:version>
                <doc:date>2011-08-07</doc:date>
                <doc:description>Original version</doc:description>
             </doc:change>
             <doc:change>
                 <doc:version>2.0b</doc:version>
                 <doc:date>2011-11-01</doc:date>
                 <doc:description>Updated imports</doc:description>
             </doc:change>
          </doc:changeHistory>
          <doc:contact>
             <doc:name>Ken Vaughn</doc:name>
                <doc:email>kvaughn@trevilon.com</doc:email>
          </doc:contact>
       </xs:documentation>
   </xs:annotation>
   <!-- ===== Imports ===== -->
   <xs:import namespace="urn:iso:std:iso:14817:tech:Documentation-2"
       schemaLocation="../common/14817-Documentation-2.0b.xsd"/>
```

```
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonBasicComponents-2"
       schemaLocation="../common/24531-CommonBasicComponents-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonAggregateComponents-2"
       schemaLocation="../common/24531-CommonAggregateComponents-2.0b.xsd"/>
   <xs:import namespace="urn:iso:std:iso:24531:tech:CommonMessageComponents-2"
       schemaLocation="../common/24531-CommonMessageComponents-2.0b.xsd"/>
   <!-- ===== Aggregate Element Definitions ===== -->
   <xs:element name="incidentList" type="IncidentListStructure"/>
   <!-- ===== Aggregate Type Definitions ===== -->
   <xs:complexType name="IncidentListStructure">
       <xs:annotation>
          <xs:documentation>
             <doc:dataConceptType>Message</doc:dataConceptType>
             <doc:name>IncidentListStructure():message
                </doc:name>
             <doc:definition>A listing of all known active incidents within
                    the requested interestArea.
                </doc:definition>
          </xs:documentation>
       </xs:annotation>
       <xs:complexContent>
          <xs:extension base="msg:MessageStructure">
             <xs:sequence>
                 <xs:element ref="cac:incident" minOccurs="0" maxOccurs="unbounded">
                    <xs:annotation>
                       <xs:documentation>
                          <doc:dataConceptType>Association</doc:dataConceptType>
                          <doc:name>IncidentList.incidents:Incident
                                </doc:name>
                          <doc:definition>An list of incidents within the requested
                                    interestArea.
                                </doc:definition>
                          <doc:multiplicity>0..*</doc:multiplicity>
                       </xs:documentation>
                    </xs:annotation>
                 </xs:element>
             </xs:sequence>
          </xs:extension>
       </xs:complexContent>
   </xs:complexType>
</xs:schema>
```

Annex L
(informative)

Example message exchange: default genericode files

L.1 General

This annex provides the default genericode files for the example message exchange defined in Annex C.

L.2 Incident type code list

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<gc:CodeList xmlns:gc="http://docs.oasis-open.org/codelist/ns/genericode/1.0/">
   <!--
       Description: Code list for identifying the type of incident.
       Status: Draft
       Standard: ISO 24531
       Using XML in ITS standards, data registries, and data dictionaries
       Change History:
          Version: 1.0a
             Date: 2011-08-07
             Description: Original Version
       Contact:
          Name: Ken Vaughn
          E-mail: kvaughn@trevilon.com
   -->
   <Identification>
       <ShortName>IncidentTypeCode</ShortName>
       <LongName>Incident Type Code</LongName>
       <Version>1.0</Version>
       <Agency>
          <LongName xml:lang="en">ISO TC on Intelligent Transportation Systems
          </LongName>
          <Identifier>ISO TC 204</Identifier>
       </Agency>
   </Identification>
   <ColumnSet>
       <Column Id="code" Use="required">
          <ShortName>Code</ShortName>
          <Data Type="normalizedString"/>
       </Column>
       <Column Id="name" Use="optional">
          <ShortName>Name</ShortName>
          <Data Type="string"/>
       </Column>
       <Column Id="support" Use="required">
          <ShortName>Support</ShortName>
          <Data Type="string"/>
       </Column>
       <Key Id="codeKey">
          <ShortName>CodeKey</ShortName>
          <ColumnRef Ref="code"/>
       </Key>
   </ColumnSet>

   <!-- ===== Code List ===== -->
   <SimpleCodeList>
       <Row>
          <Value ColumnRef="code">
             <SimpleValue>CR</SimpleValue>
          </Value>
          <Value ColumnRef="name">
```

```
      <SimpleValue>Crash</SimpleValue>
   </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
</Row>
<Row>
   <Value ColumnRef="code">
      <SimpleValue>RW</SimpleValue>
   </Value>
   <Value ColumnRef="name">
      <SimpleValue>Roadwrok</SimpleValue>
   </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
</Row>
<Row>
   <Value ColumnRef="code">
      <SimpleValue>CO</SimpleValue>
   </Value>
   <Value ColumnRef="name">
      <SimpleValue>Congestion</SimpleValue>
   </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
</Row>
<Row>
   <Value ColumnRef="code">
      <SimpleValue>DI</SimpleValue>
   </Value>
   <Value ColumnRef="name">
      <SimpleValue>Disaster</SimpleValue>
   </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
</Row>
<Row>
   <Value ColumnRef="code">
      <SimpleValue>SE</SimpleValue>
   </Value>
   <Value ColumnRef="name">
      <SimpleValue>Special Event</SimpleValue>
   </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
</Row>
<Row>
   <Value ColumnRef="code">
      <SimpleValue>WE</SimpleValue>
   </Value>
   <Value ColumnRef="name">
      <SimpleValue>Weather</SimpleValue>
            </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
</Row>
<Row>
   <Value ColumnRef="code">
      <SimpleValue>AQ</SimpleValue>
   </Value>
   <Value ColumnRef="name">
      <SimpleValue>Air Quality</SimpleValue>
   </Value>
   <Value ColumnRef="support">
      <SimpleValue>Optional</SimpleValue>
   </Value>
```

```
        </Row>
        <Row>
           <Value ColumnRef="code">
              <SimpleValue>VI</SimpleValue>
           </Value>
           <Value ColumnRef="name">
              <SimpleValue>Visibility</SimpleValue>
           </Value>
           <Value ColumnRef="support">
              <SimpleValue>Optional</SimpleValue>
           </Value>
        </Row>
    </SimpleCodeList>
</gc:CodeList>
```

L.3 Traffic impact code list

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<gc:CodeList xmlns:gc="http://docs.oasis-open.org/codelist/ns/genericode/1.0/">
    <!--
        Description: Code list for identifying the impact that an incident is
        having or is expected to have on travel conditions.
        Status: Draft
        Standard: ISO 24531
        Using XML in ITS standards, data registries, and data dictionaries
        Change History:
           Version: 1.0a
              Date: 2011-08-07
              Description: Original Version
        Contact:
           Name: Ken Vaughn
           E-mail: kvaughn@trevilon.com
    -->

    <Identification>
        <ShortName>TrafficImpactCode</ShortName>
        <LongName>Traffic Impact Code</LongName>
        <Version>1.0</Version>
        <CanonicalUri>urn:iso:std:iso:24531:tech:rafficImpactCode-1</CanonicalUri>
        <CanonicalVersionUri>urn:iso:std:iso:24531:tech:TrafficImpactCode-1
        </CanonicalVersionUri>
        <Agency>
           <LongName xml:lang="en">ISO</LongName>
           <Identifier>ISO</Identifier>
        </Agency>
    </Identification>
    <ColumnSet>
        <Column Id="code" Use="required">
           <ShortName>Code</ShortName>
           <Data Type="normalizedString"/>
        </Column>
        <Column Id="name" Use="optional">
           <ShortName>Name</ShortName>
           <Data Type="string"/>
        </Column>
        <Column Id="support" Use="required">
           <ShortName>Support</ShortName>
           <Data Type="string"/>
        </Column>
        <Key Id="codeKey">
           <ShortName>CodeKey</ShortName>
           <ColumnRef Ref="code"/>
        </Key>
    </ColumnSet>
    <SimpleCodeList>
        <Row>
           <Value ColumnRef="code">
              <SimpleValue>0</SimpleValue>
           </Value>
           <Value ColumnRef="name">
              <SimpleValue>No noticeable impact</SimpleValue>
```

```
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef="code">
               <SimpleValue>1</SimpleValue>
            </Value>
            <Value ColumnRef="name">
               <SimpleValue>Minor slowing without significant delay</SimpleValue>
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef="code">
               <SimpleValue>2</SimpleValue>
            </Value>
            <Value ColumnRef="name">
               <SimpleValue>Delays of 1-5 minutes</SimpleValue>
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef="code">
               <SimpleValue>3</SimpleValue>
            </Value>
            <Value ColumnRef="name">
               <SimpleValue>Delays of 5-20 minutes</SimpleValue>
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef="code">
               <SimpleValue>4</SimpleValue>
            </Value>
            <Value ColumnRef="name">
               <SimpleValue>Delays of 5-20 minutes and delays affecting nearby routes</Sim-
pleValue>
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef="code">
               <SimpleValue>5</SimpleValue>
            </Value>
            <Value ColumnRef="name">
               <SimpleValue>Delays of 20-60 minutes</SimpleValue>
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef="code">
               <SimpleValue>6</SimpleValue>
            </Value>
            <Value ColumnRef="name">
               <SimpleValue>Delays of 20-60 minutes and delays affecting nearby routes of
more than 5 minutes</SimpleValue>
            </Value>
            <Value ColumnRef="support">
               <SimpleValue>Mandatory</SimpleValue>
```

```
            </Value>
         </Row>
         <Row>
            <Value ColumnRef=”code”>
               <SimpleValue>7</SimpleValue>
            </Value>
            <Value ColumnRef=”name”>
               <SimpleValue>Regional conditions slowing traffic speeds by 25-50%</SimpleV-
alue>
            </Value>
            <Value ColumnRef=”support”>
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef=”code”>
               <SimpleValue>8</SimpleValue>
                     </Value>
            <Value ColumnRef=”name”>
               <SimpleValue>Delays of more than 60 minutes</SimpleValue>
                     </Value>
            <Value ColumnRef=”support”>
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef=”code”>
               <SimpleValue>9</SimpleValue>
            </Value>
            <Value ColumnRef=”name”>
               <SimpleValue>Delays of more than 60 minutes and delays affecting nearby
routes of more than 15 minutes</SimpleValue>
            </Value>
            <Value ColumnRef=”support”>
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
         <Row>
            <Value ColumnRef=”code”>
               <SimpleValue>10</SimpleValue>
            </Value>
            <Value ColumnRef=”name”>
               <SimpleValue>Regional conditions slowing traffic speeds by more than 50%</Sim-
pleValue>
            </Value>
            <Value ColumnRef=”support”>
               <SimpleValue>Mandatory</SimpleValue>
            </Value>
         </Row>
      </SimpleCodeList>
</gc:CodeList>
```

Annex M
(informative)

Example message exchange: default context value association file

M.1 General

This annex provides the context-value association (CVA) file for the example message exchange defined in Annex C.

M.2 Default context value association file

```
<?xml version="1.0" encoding="UTF-8"?>
<cva:ContextValueAssociation xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://docs.oasis-open.org/codelist/ns/ContextValueAssociation/1.0/
 http://docs.oasis-open.org/codelist/cs01-ContextValueAssociation-1.0/xsd/ContextValueA-
ssociation.xsd"
 xmlns:cva="http://docs.oasis-open.org/codelist/ns/ContextValueAssociation/1.0/"
 xmlns:sch="http://purl.oclc.org/dsdl/schematron"
 xmlns:cc="urn:iso:std:iso:24531:tech:CommonComponents-1"
 xmlns:msc="urn:iso:std:iso:24531:tech:MessageSetComponents-1"
 name="DefaultQualifiedDataTypes"
 version="2011-08-07 20:08:23(UTC)"
 id="urn:iso:std:iso:24531:tech:DefaultQualifiedDataTypes-1">
   <Annotation>
       <Description>
        This describes all of the qualified supplementary components and
        business information entities for the ISO 24531 Example.
       </Description>
   </Annotation>
   <Title>ISO 24531 Example qualified information items</Title>
     <ValueTests>
         <!-- Value test allow you to define valid ranges -->
         <ValueTest xml:id="latTest1" test=". &lt;= 90"/>
        <ValueTest xml:id="latTest2" test=". &gt;= -90"/>
        <ValueTest xml:id="longTest1" test=". &lt;= 180"/>
        <ValueTest xml:id="longTest2" test=". &gt;= -180"/>
     </ValueTests>
   <ValueLists>
       <!-- Value lists allow you to define valid code lists -->
       <ValueList xml:id="TrafficImpactCode-1.0"
          uri="../../cl/24531-sample/TrafficImpactCode-1.0a.gc"/>
       <ValueList xml:id="IncidentTypeCode-1.0"
          uri="../../cl/24531-sample/IncidentTypeCode-1.0a.gc"/>
   </ValueLists>
   <Contexts>
       <!-- Contexts alow you to specify the locations at which certain rules
       apply -->
       <!-- Traffic Impact Codes should always be in the defined list -->
       <Context address="//msc:trafficImpactCode"
          values="TrafficImpactCode-1.0"/>
       <!-- Latitude should always be between -90 and +90 -->
       <Context address="//cc:latitudeQuantity" values="latTest1 latTest2"/>
       <!-- Longitude should always be between -180 and +180 -->
       <Context address="//cc:longitudeQuantity" values="longTest1 longTest2"/>
       <!-- The type code should be contained in the incident type code list
       when the field appears within an incident structure; however, the
       "type" property may be used in different contexts where other code lists
       apply, so we need to ensure that we call out the specific usage where
       this test will be applied -->
       <Context address="//msc:incident/msc:typeCode"
          values="IncidentTypeCode-1.0"/>
```

```
   </Contexts>
</cva:ContextValueAssociation>
```

Annex N
(informative)

Example CVA transformation file

N.1 General

This annex provides an example transformation file that can be used to transform a CVA file (e.g. as provided in Annex K), and its referenced genericode files, into an eXtensible Stylesheet Language Transformation (XSLT) file (e.g. such as provided in Annex M). The resultant file can then be used to validate the values in an XML message document that corresponds to a defined message set, such as the example message exchange defined in Annex C.

N.2 CVA transformation file

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:cva="http://docs.oasis-open.org/codelist/ns/ContextValueAssociation/1.0/"
    version="2.0">

    <xsl:template match="/">
        <xsl:text disable-output-escaping="yes">
&lt;xsl:stylesheet
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"</xsl:text>
        <xsl:apply-templates select="/cva:ContextValueAssociation" />
        <xsl:text disable-output-escaping="yes">
    version="1.0"&gt;
    &lt;xsl:template match="/"&gt;
        &lt;html&gt;
        &lt;p&gt;Start&lt;/p&gt;
</xsl:text>
        <xsl:for-each select="/Context">
                <xsl:text disable-output-escaping="yes">
        &lt;xsl:for-each select="</xsl:text>
                <xsl:value-of select="@address"/>
                <xsl:text disable-output-escaping="yes">"&gt;
            &lt;xsl:choose&gt;
                &lt;xsl:when test="</xsl:text><xsl:apply-templates select="."/>
                <xsl:text disable-output-escaping="yes">"&gt;&lt;/xsl:when&gt;
                &lt;xsl:otherwise&gt;
                    &lt;p&gt;&lt;xsl:for-each select="ancestor-or-self::*"&gt;
                        &lt;xsl:value-of select="name()"/&gt;/
                    &lt;/xsl:for-each&gt;
                    = &lt;xsl:value-of select="."/&gt; is out-of-range.&lt;/p&gt;
                &lt;/xsl:otherwise&gt;</xsl:text>
                <xsl:text disable-output-escaping="yes">
            &lt;/xsl:choose&gt;
      &lt;/xsl:for-each&gt;
                </xsl:text>
        </xsl:for-each>
        <xsl:text disable-output-escaping="yes">
        &lt;p&gt;Finished&lt;/p&gt;
        &lt;/html&gt;
   &lt;/xsl:template&gt;
&lt;/xsl:stylesheet&gt;</xsl:text>
    </xsl:template>

    <xsl:template match="cva:ContextValueAssociation">
        <xsl:for-each select="namespace::node()">
            <xsl:if test="(name(.) != 'xml') and (name(.) != 'cva') and (name(.) !=
'sch')">
```

```
    xmlns:<xsl:value-of select="name(.)" />="<xsl:value-of select="." />"</xsl:if></
xsl:for-each>
    </xsl:template>

    <xsl:template match="//Context">
        <xsl:variable name="values" select="tokenize(@values, '\s+')" />
        <xsl:variable name="loop" select="0" />
        <xsl:for-each select="//ValueTest">
            <xsl:variable name="id" select="@xml:id" />
            <xsl:variable name="test" select="@test" />
            <xsl:for-each select="$values">
                <xsl:variable name="pos" select="position()" />
                <xsl:variable name="value" select="." />
                <xsl:if test="$id = $value"><xsl:if test="$pos > 1"> and </
xsl:if>(<xsl:value-of select="$test" />)</xsl:if>
                <xsl:variable name="loop" select="1" />
            </xsl:for-each>
        </xsl:for-each>

        <xsl:for-each select="//ValueList">
            <xsl:variable name="id" select="@xml:id" />
            <xsl:variable name="uri" select="@uri" />
            <xsl:for-each select="$values">
                <xsl:variable name="pos" select="position()" />
                <xsl:variable name="value" select="." />
                <xsl:if test="$id = $value"><xsl:if test="$pos > 1"> or </
xsl:if><xsl:call-template name="ValueList"><xsl:with-param name="uri" select="$uri"></
xsl:with-param></xsl:call-template></xsl:if>
                <xsl:variable name="loop" select="1" />
            </xsl:for-each>
        </xsl:for-each>

    </xsl:template>


    <xsl:template name="ValueList">
        <xsl:param name="uri" />
        <xsl:text>contains('</xsl:text>
        <xsl:for-each select="document($uri)//Value"><xsl:if test="@ColumnRef = 'code'">
            <xsl:text> </xsl:text><xsl:value-of select="SimpleValue" /><xsl:text> </
xsl:text>
        </xsl:if></xsl:for-each>
        <xsl:text>',concat(' ',../@id))</xsl:text>
    </xsl:template>

</xsl:stylesheet>
```