INTERNATIONAL STANDARD

ISO 16684-2

First edition
2014-12-01

Graphic technology — Extensible metadata platform (XMP) —

Part 2: Description of XMP schemas using RELAX NG

Technologie graphique — Plate-forme de métadonnées extensibles (XMP) —
Partie 2: Description des schémas XMP utilisant RELAX NG

Reference number
ISO 16684-2:2014(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 (see www.iso.org/directives).

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

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

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/130 *Graphic technology*.

ISO 16684 consists of the following parts, under the general title *Graphic technology — Extensible metadata platform (XMP)*:

— *Part 1: Data model, serialization, and core properties*

— *Part 2: Description of XMP schemas using RELAX NG*

Introduction

ISO 16684 (all parts) defines aspects of the Extensible Metadata Platform (XMP) that are generic, neutral to the domain of usage. Refer to the Introduction in ISO 16684-1 for general information. This part of ISO 16684 is about description of XMP schemas for formal or mechanical validation of XMP. RELAX NG has been chosen as the schema language. It is an ISO standard, ISO/IEC 19757-2, and is both powerful and easy to use.

There are two major components of formal validation, schemas and validation engines. A schema, or schema file, is a formal description of constraints regarding the structure and contents of properties in an XMP packet, on top of the requirements for conforming XMP packets as mandated by ISO 16684-1. A validation engine is a software tool that compares an input XMP packet to one or more schemas, and produces a report on whether the XMP packet conforms to the schemas.

This part of ISO 16684 defines policies for validation engines to follow so that schemas can be shared, so that the schemas do not require customization for each validation engine. It also defines policies for schemas to follow in order to operate with a conforming validation engine, and to make the schemas robust and modular.

This part of ISO 16684 does not address how a validation engine reports success or failure. Reporting success is easy, reporting failure can be complicated by a number of factors.

— It can be difficult to relate a specific RDF usage error to a human-understood XMP data model.

— Recovery from one error can be difficult, masking other errors after the first.

— As an open model, creation of new data items in XMP is expected. Allowing for this in schemas and/or clear reporting of unexpected input when validating can be difficult.

Graphic technology — Extensible metadata platform (XMP) —

Part 2: Description of XMP schemas using RELAX NG

1 Scope

This part of ISO 16684 specifies the use of RELAX NG to describe serialized XMP metadata. This applies to how conforming schemas can use the features of RELAX NG.

2 Normative references

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

ISO 16684-1:2012, *Graphic technology — Extensible metadata platform (XMP) specification — Part 1: Data model, serialization and core properties*

ISO/IEC 19757-2, *Information technology — Document Schema Definition Language (DSDL) — Part 2: Regular-grammar-based validation — RELAX NG*

3 Terms and definitions

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

3.1
canonical serialization
serialization providing a one-to-one mapping between the XMP data model and the serialized XML

3.2
general qualifier
XMP qualifier other than xml:lang

3.3
schema
schema file
formal description of serialized XMP

3.4
validation
process of verifying whether serialized XMP follows one or more schemas

3.5
validation engine
software tool that performs validation

3.6
XMP
extensible metadata platform, as defined by ISO 16684-1

3.7
XMP entity
XMP property, array item, structure field, or qualifier

4 Conformance

This part of ISO 16684 describes methodology to create interoperable software and schema files in order to validate XMP metadata using regular-grammar-based validation schemas defined in ISO/IEC 19757-2 and referred to in this part of ISO 16684 as RELAX NG. Conformance on the part of a software validation engine enables the creation of schema files with lower complexity. Conforming validation engines shall adhere to all requirements of ISO 16684-1 and this part of ISO 16684. Conforming validation engines can provide additional features that are not explicitly forbidden by this part of ISO 16684. Conformance on the part of schema files enables their interchange among conforming validation engines. Conforming schema files shall adhere to all requirements of this part of ISO 16684. Conforming schema files can provide additional features that are not explicitly forbidden by this part of ISO 16684.

5 Canonical serialization of XMP

5.1 General

A major difficulty in validating XMP is that the RDF metadata format used by XMP allows multiple XML representations for the same metadata content. For reference, see ISO 16684-1:2012, 7.9. The RELAX NG schema language is used to validate serialized XML, not the RDF or XMP data models. Writing a RELAX NG schema to cover all possible XML forms for XMP is unacceptably complex. A canonical serialization of XMP is defined to limit this complexity. The canonical serialization requires specific forms of the XMP serialization defined in ISO 16684-1:2012, Clause 7, banning other forms. This provides a one-to-one mapping between the XMP data model and the canonical XML.

An XMP validation engine shall produce a canonical serialization of the XMP as part of the validation process. The validation engine shall accept any XMP as input that is allowed under ISO 16684-1. This XMP shall be parsed then serialized using the canonical forms described in this Clause. Next, the RELAX NG schema or schemas shall be applied to that canonical serialization. The XMP validation engine shall not modify original files containing the input XMP as part of the validation process.

5.2 XMP packet serialization

The canonical serialization of XMP shall have an rdf:RDF outermost XML element, which shall contain a single rdf:Description element to contain all XMP properties. That is, properties for all top-level namespaces are within this one rdf:Description element. The XMP properties shall be grouped by namespace. The namespace groups can have any order.

EXAMPLE An XMP metadata packet is shown containing XMP properties of different namespaces in a single rdf:Description element.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
              xmlns:dc="http://purl.org/dc/elements/1.1/"
              xmlns:xmp="http://ns.adobe.com/xap/1.0/"
              xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/">
       <rdf:Description rdf:about="">
              <dc:format>image/jpeg</dc:format>
              <xmp:Rating>3</xmp:Rating>
              <xmp:CreateDate>2012-02-29T12:33:44</xmp:CreateDate>
              <xmpMM:DocumentID>uuid:example1234</xmpMM:DocumentID>
       </rdf:Description>
</rdf:RDF>
```

5.3 Property serialization

The canonical serialization of XMP shall use the XML element form for XMP properties. The property attribute notation defined in ISO 16684-1:2012, 7.9.2.2 shall not be used.

5.4 Structure value serialization

The canonical serialization of XMP shall use a nested rdf:Description element for structure values. The structure fields shall be serialized as XML elements within that rdf:Description element. Values in a structure can be of type simple, structure, or array. They shall be serialized as described in 6.3, 6.4, and 6.5. The rdf:parseType=”resource” attribute notation defined in ISO 16684-1:2012, 7.9.2.3 shall not be used. The structure field attribute notation defined in ISO 16684-1:2012, 7.9.2.4 shall not be used.

EXAMPLE A structure value from an XMP metadata stream is shown in its canonically serialized form.

```
<xmpMM:DerivedFrom>
       <rdf:Description>
              <stRef:documentID>id:document</stRef:documentID>
              <stRef:instanceID>id:instance</stRef:instanceID>
       </rdf:Description>
</xmpMM:DerivedFrom>
```

5.5 Array value serialization

The canonical serialization of XMP shall use a nested rdf:Bag, rdf:Seq, or rdf:Alt element for array values. The array items shall be rdf:li elements within the rdf:Bag, rdf:Seq, or rdf:Alt element.

EXAMPLE An array value from an XMP metadata stream is shown in its canonically serialized form.

```
<dc:subject>
       <rdf:Bag>
              <rdf:li>subject 1</rdf:li>
       <rdf:li>subject 2</rdf:li>
       </rdf:Bag>
</dc:subject>
```


5.6 Qualifier serialization

The canonical serialization of XMP shall use the XML syntax defined in ISO 16684-1:2012, 7.8 for qualifiers. An xml:lang qualifier shall be serialized as an xml:lang attribute in the start tag of the XML element whose name is that of the XMP entity having the xml:lang qualifier. All general qualifiers shall be serialized as XML elements within an rdf:Description element that is within the XML element whose name is that of the XMP entity having the qualifier. The value of that XMP entity shall be within an rdf:value element within that rdf:Description element. The rdf:value element should be the first XML element within that rdf:Description element.

The rdf:parseType=”resource” notation defined in ISO 16684-1:2012, 7.9.2.3 shall not be used for an XMP entity with general qualifiers. The field attribute notation defined in ISO 16684-1:2012, 7.9.2.4 shall not be used for an XMP entity with general qualifiers. The RDF TypedNode notation defined in ISO 16684-1:2012, 7.9.2.5 shall not be used for an rdf:type qualifier.

EXAMPLE A part from an XMP metadata stream using a qualifier is shown in its canonically serialized form.

```
<xmp:Identifier>
       <rdf:Bag>
              <rdf:li>
                     <rdf:Description>
                            <rdf:value>0-13-110941-3</rdf:value>
                     <xmpidq:Scheme>ISBN</xmpidq:Scheme>
                     </rdf:Description>
       </rdf:li>
   </rdf:Bag>
</xmp:Identifier>
```

6 RELAX NG idioms for XMP

6.1 General

Having a canonical serialization for XMP is necessary to simplify the creation of RELAX NG schemas. As a consequence, considerably more freedom can be granted in the creation of the RELAX NG schemas.

This clause defines idioms for RELAX NG schemas that can improve reuse and interchange. They can also enable improved operation of a validation engine, for example making it easier to provide helpful error messages. Use of these idioms is not required.

6.2 Modularization

RELAX NG provides three main forms of modularization, the rng:define element, the rng:grammar element, and the rng:include element. The RELAX NG idioms for XMP provide recommendations for using these elements to simplify the creation of full RELAX NG schemas for XMP and to improve reuse and interchange of schema modules.

The rng:define element is perhaps the most important. It provides a means to create a named pattern, typically used to define a specific easily understood aspect of the XML to be validated. Lower level rng:define patterns can be referenced by name in higher level RELAX NG patterns.

The rng:grammar element provides a means to package rng:define patterns, and to control the scope of their names if desired.

The rng:include element provides a means to textually include one RELAX NG schema file within another.

While no specific naming conventions are required, the following approach can be chosen:

— names for individual properties can be composed as “prefix.localName”, e.g.”dc.title”;

— names for structure fields can be composed as “prefix.localName”, e.g. “stRef:documentID”;

— names for types can be composed as “Standard/Origin.Types.QValue.Typename”, e.g.”ISO 16684-1.Types.QValue.Boolean”, and “Standard/Origin.Types.Base.Typename”, e.g.”ISO 16684-1.Types.Base.Boolean”;

— names for array types can additionally denote the array type in their name; for example an unordered array of simple type Boolean can be expressed as e.g ISO 16684-1.Types.QValue.UnorderedArray.Boolean or ISO 16684-1.Types.Base.UnorderedArray.Boolean

NOTE The QValue idiom ensures that properties with or without qualifiers can be validated, the Base idiom contains the actual RELAX NG grammar for the type.

6.3 Use of data types

6.3.1 General

The XMP data model is defined in ISO 16684-1:2012, 6.3 using 3 forms of value: simple, structure, and array. Specific value types used are defined in ISO 16684-1:2012, 8.2. These can be thought of as data types, and formally defined that way for validation.

RELAX NG provides a primitive data type for unconstrained text. It also has features to describe various kinds of constrained text. It allows use of external data type libraries, most importantly the W3C XML Schema Datatypes. The rng:define element can be used to define arbitrary custom data types.

The rng:text type shall be used where an XMP value is unconstrained text. In all other cases, an rng:define element should be used to create a custom data type. This provides modularization of the data type semantics, makes the RELAX NG schema easier to understand, and fosters reuse of data types.

The pattern within an rng:define for a simple XMP value should describe the XML character data of the serialized value with appropriate semantic restrictions.

EXAMPLE Two rng:define instances describe constraints on text values to create custom data types.

```
<rng:define name=”ISO16684-1.Types.Boolean” combine=”choice”>
       <rng:data type=”string”>
              <rng:param name=”pattern”>True|False</rng:param>
       </rng:data>
</rng:define>
```



```
<rng:define name="xmp.type.Rating" combine="choice">
   <rng:choice>
      <rng:value type="float">-1</rng:value>
      <rng:data type="float">
         <rng:param name="minInclusive">0</rng:param>
         <rng:param name="maxInclusive">5</rng:param>
      </rng:data>
   </rng:choice>
</rng:define>
```

NOTE The combine="choice" attribute in the rng:define element is a workaround for the way rng:define and rng:include interact. Suppose the Boolean type is defined in one schema file, then included and used in two other schema files, which are then themselves included and used in another schema file. The final schema file will have two definitions for Boolean, which are, of course, identical. RELAX NG does not detect that they are identical and requires the combine attribute to be used telling how to select among them. Using combine="choice" says to pick any, which is appropriate since they are identical.

6.3.2 Structure value data types

The pattern within an rng:define for a structure value type shall describe the nested rdf:Description element and the structure fields within that. Each field should have a separate rng:define that describes its XML element and value. The value should be given as an rng:ref element referring to the appropriate data type.

EXAMPLE Several rng:define instances describe a structure value type using rng:ref elements.

```
<rng:define name="ISO16684-1.Types.ResourceRef" combine="choice">
       <rng:element name="rdf:Description">
              <rng:interleave>
                     <rng:optional>
                            <rng:ref name="stRef.documentID"/>
                     </rng:optional>
                     <rng:optional>
                            <rng:ref name="stRef.filePath"/>
                     </rng:optional>
              </rng:interleave>
       </rng:element>
</rng:define>

<rng:define name="stRef.documentID" combine="choice">
       <rng:element name="stRef:documentID">
              <rng:ref name="ISO16684-1.Types.GUID"/>
       </rng:element>
</rng:define>

<rng:define name="stRef.filePath" combine="choice">
       <rng:element name="stRef:filePath">
              <rng:ref name="ISO16684-1.Types.URI"/>
       </rng:element>
</rng:define>
```


NOTE Only two of the RenditionRef fields, defined in ISO 16684-1:2012, 8.2.2.9, are shown above in order to reduce the size of the example.

The rng:interleave element shall be used to denote that the structure fields can appear in any order. The rng:optional element shall be used if a structure field is optional, it shall be omitted if a structure field is required.

6.3.3 Array value data types

The pattern within an rng:define for an array value type shall describe the nested rdf:Bag, rdf:Seq, or rdf:Alt element, and the rdf:li array item elements within that.

EXAMPLE 1 An rng:define instance describes an array data type.

```
<rng:define name="ISO16684-1.Types.OrderedArray.Date" combine="choice">
       <rng:element name="rdf:Seq">
              <rng:zeroOrMore>
                     <rng:element name="rdf:li">
```

```
                        <rng:ref name="ISO16684-1.Types.Date"/>
                  </rng:element>
            </rng:zeroOrMore>
      </rng:element>
</rng:define>
```

The rng:zeroOrMore element should be used if the array can be empty and has no upper bound. The rng:oneOrMore element should be used if the array has at least one item and has no upper bound. For more than one required item, that number of explicit rdf:li patterns should be used. For a closed upper bound, the appropriate number of rdf:li patterns enclosed in rng:optional should be used.

EXAMPLE 2 An rng:define instance describes minimum and maximum number of entries in an array data type.

```
<rng:define name="ThreeToFiveOrderedDates" combine="choice">
      <rng:element name="rdf:Seq">
            <rng:ref name="DateItem"/>
            <rng:ref name="DateItem"/>
            <rng:ref name="DateItem"/>
            <rng:optional>
                  <rng:ref name="DateItem"/>
            </rng:optional>
            <rng:optional>
                  <rng:ref name="DateItem"/>
            </rng:optional>
      </rng:element>
</rng:define>

<rng:define name="DateItem" combine="choice">
      <rng:element name="rdf:li">
            <rng:ref name="ISO16684-1.Types.Date"/>
      </rng:element>
</rng:define>
```


6.3.4 Collections of data types

Collections of useful data types should be gathered in a RELAX NG schema file. This schema file should have an outer rng:grammar element that contains the rng:defines for the data types.

EXAMPLE An rng:grammar element in a separate schema file describes a collection of often used data types.

```
<rng:grammar xmlns:rng="http://relaxng.org/ns/structure/1.0"
                  datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

      <rng:define name="ISO16684-1.Types.Boolean" combine="choice">
            <rng:data type="string">
                  <rng:param name="pattern">True|False</rng:param>
            </rng:data>
      </rng:define>

      <rng:define name="ISO16684-1.Types.Integer" combine="choice">
            <rng:data type="integer"/>
      </rng:define>

      <rng:define name="ISO16684-1.Types.Rational" combine="choice">
            <rng:data type="string">
                  <rng:param name="pattern">\d+/[1-9]\d*</rng:param>
            </rng:data>
      </rng:define>

</rng:grammar>
```

NOTE Only three of the types defined in ISO 16684-1:2012, 8.2 are shown above in order to reduce the size of the example.

6.4 RELAX NG for properties

A top-level namespace shall have a RELAX NG schema file with an outer rng:grammar element. Within that shall be an rng:define element listing all of the properties it contains. Each property should have a

separate rng:define that describes its XML element and value. The value should be given as an rng:ref element referring to the appropriate data type.

EXAMPLE An outer rng:grammar element in its own file for a top-level namespace lists the rng:define elements for all of the properties it contains.

```
<rng:grammar xmlns:rng="http://relaxng.org/ns/structure/1.0"
                    xmlns:xmp="http://ns.adobe.com/xap/1.0/"
                    datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

   <rng:include href="ISO16684-1.Types.rng"/>

   <rng:define name="ISO16684-1.Properties-xmp" combine="choice">
      <rng:interleave>
            <rng:optional>
                  <rng:ref name="xmp.CreateDate"/>
            </rng:optional>
            <rng:optional>
                  <rng:ref name="xmp.Rating"/>
            </rng:optional>
      </rng:interleave>
   </rng:define>

   <rng:define name="xmp.CreateDate" combine="choice">
      <rng:element name="xmp:CreateDate">
            <rng:ref name="ISO16684-1.Types.Date"/>
      </rng:element>
   </rng:define>

   <rng:define name="xmp.Rating" combine="choice">
      <rng:element name="xmp:Rating">
            <rng:ref name="xmp.Rating"/>
      </rng:element>
   </rng:define>

   <rng:define name="xmp.type.Rating" combine="choice">
      <rng:choice>
            <rng:value type="float">-1</rng:value>
            <rng:data type="float">
                  <rng:param name="minInclusive">0</rng:param>
                  <rng:param name="maxInclusive">5</rng:param>
            </rng:data>
      </rng:choice>
   </rng:define>

</rng:grammar>
```


NOTE Only two of the XMP namespace properties defined in ISO 16684-1:2012, 8.4 are shown above in order to reduce the size of the example.

The rng:interleave element shall be used to denote that the properties can appear in any order. The rng:optional element shall be used if a property is optional, it shall be omitted if a property is required.

6.5 RELAX NG for XMP packets

A RELAX NG schema for a complete XMP packet shall have an outer rng:grammar element. This should contain rng:include elements for the allowed top-level namespaces and a single rng:start element. The rng:start element shall contain patterns for the outer rdf:RDF and rdf:Description elements of the canonical XMP serialization. Within the rdf:Description pattern shall be an rng:interleave element that contains references to the properties in the allowed namespaces.

EXAMPLE An outer rng:grammar element in its own file for a complete XMP packet contains rng:include elements for two top-level namespaces and an rng:start element with patterns for the outer rdf:RDF and rdf:Description elements.

```
<rng:grammar xmlns:rng="http://relaxng.org/ns/structure/1.0"
             xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

      <rng:include href="ISO16684-1_Properties-dc.rng"/>
```