INTERNATIONAL STANDARD

ISO/IEC 21000-2

Second edition
2005-10-01

Information technology — Multimedia framework (MPEG-21) —

Part 2:
Digital Item Declaration

Technologies de l'information — Cadre multimédia (MPEG-21) —

Partie 2: Déclaration d'article numérique

Reference number
ISO/IEC 21000-2:2005(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

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) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

ISO/IEC 21000-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This second edition cancels and replaces the first edition (ISO/IEC 21000-2:2003), which has been technically revised.

ISO/IEC 21000 consists of the following parts, under the general title *Information technology — Multimedia framework (MPEG-21)*:

— *Part 1: Vision, Technologies and Strategy [Technical Report]*

— *Part 2: Digital Item Declaration*

— *Part 3: Digital Item Identification*

— *Part 5: Rights Expression Language*

— *Part 6: Rights Data Dictionary*

— *Part 7: Digital Item Adaptation*

— *Part 8: Reference Software*

— *Part 9: File Format*

— *Part 10: Digital Item Processing*

— *Part 11: Evaluation Tools for Persistent Association Technologies [Technical Report]*

— *Part 12: Test Bed for MPEG-21 Resource Delivery [Technical Report]*

— *Part 15: Event Reporting*

— *Part 16: Binary Format*

The following parts are under preparation:

— *Part 4: Intellectual Property Management and Protection Components*

— *Part 14: Conformance Testing*

Introduction

Today, many elements exist to build an infrastructure for the delivery and consumption of multimedia content. There is, however, no “big picture” to describe how these elements, either in existence or under development, relate to each other. The aim for MPEG-21 is to describe how these various elements fit together. Where gaps exist, MPEG-21 will recommend which new standards are required. ISO/IEC JTC 1/SC 29/WG 11 (MPEG) will then develop new standards as appropriate while other relevant standards may be developed by other bodies. These specifications will be integrated into the multimedia framework through collaboration between MPEG and these bodies.

The result is an open framework for multimedia delivery and consumption, with both the content creator and content consumer as focal points. This open framework provides content creators and service providers with equal opportunities in the MPEG-21 enabled open market. This will also be to the benefit of the content consumer providing them access to a large variety of content in an interoperable manner.

The vision for MPEG-21 is to define a multimedia framework *to enable transparent and augmented use of multimedia resources across a wide range of networks and devices* used by different communities.

This second part of MPEG-21 (ISO/IEC 21000-2) specifies the mechanism for declaring the structure and makeup of Digital Items.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

The ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured the ISO and IEC that he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with the ISO and IEC. Information may be obtained from the companies listed in Annex A.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified in Annex A. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Information technology — Multimedia framework (MPEG-21) —

Part 2: Digital Item Declaration

1 Scope

This document describes the ISO/IEC 21000 Digital Item Declaration technology, which is Part 2 of the ISO/IEC 21000 series of International Standards. It specifies:

— the Digital Item Declaration Model (see 6),

— the Digital Item Declaration Representation in XML (see 7), and

— XML schemas comprising grammars for the Digital Item Declaration representation in XML (see 8).

2 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/IEC 21000 (all parts), *Information Technology — Multimedia Framework (MPEG-21)*

IETF RFC 2045, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*, IETF Request for Comments: 2045, November 1996

IETF RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1*, IETF Request for Comments: 2616, June 1999

IETF RFC 3548, *The Base16, Base32, and Base64 Data Encodings*, IETF Request for Comments: 3548, July 2003

IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*, IETF Request For Comments: 3986, January 2005

W3C XINCLUDE, *XML Inclusions (XInclude) Version 1.0*, W3C Recommendation, 20 December 2004

W3C XML, *Extensible Markup Language 1.0 (Second Edition)*, W3C Recommendation, 6 October 2000

W3C XMLC14N, *Canonical XML Version 1.0*, W3C Recommendation, 15 March 2001

W3C XMLNAMES, *Namespaces in XML*, W3C Recommendation, 14 January 1999

W3C XMLSCHEMA, *XML Schema Part 1: Structures Second Edition* and *XML Schema Part 2: Datatypes Second Edition*, W3C Recommendations, 28 October 2004

3 Terms and definitions

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

3.1
Abstraction
distinct intellectual or artistic creation or concept

[ISO/IEC TR 21000-1:2004]

3.2
Asset
Manifestation, i.e. physical or digital embodiment of an Expression

[ISO/IEC TR 21000-1:2004]

3.3
Digital Item
a structured digital object with a standard representation, identification and metadata within the MPEG-21 framework

NOTE This entity is the fundamental unit of distribution and transaction within the MPEG-21 framework as a whole; it has, however, no further technical meaning. Within this document (part 2 of MPEG-21: Digital Item Declaration), an *item* is a grouping of sub-*items* and/or *components* that are bound to relevant *descriptors*, as defined within the Digital Item Declaration Model (see 6). The term *item* is a technical term, and is, as such, a narrower term than Digital Item. In conclusion, the use of the two different terms Digital Item and *item* within MPEG-21 is consistent and intended.

3.4
Expression
intellectual or artistic realisation of an Abstraction

[ISO/IEC TR 21000-1:2004]

3.5
Manifestation
the physical or digital embodiment of an Expression

[ISO/IEC TR 21000-1:2004]

4 Symbols and abbreviated terms

For the purposes of this document, the following abbreviations apply.

DID: Digital Item Declaration

DIDL: Digital Item Declaration Language

EBNF: Extended Backus-Naur Form

IANA: Internet Assigned Numbers Authority

IETF: Internet Engineering Task Force

IPMP: Intellectual Property Management and Protection

JPEG: Joint Photographic Experts Group

MIME: Multipurpose Internet Mail Extensions (IETF RFC 2045)

MPEG: Moving Picture Experts Group

MPEG-21: ISO/IEC 21000 (all parts)

MPEG-7: ISO/IEC 15938

MP3: MPEG-1/2 layer III (audio coding)

RFC: Request for Comments

SVG: Scalable Vector Graphics

URI: Uniform Resource Identifier (IETF RFC 3986)

URL: Uniform Resource Locator (IETF RFC 3986)

URN: Uniform Resource Name (IETF RFC 3986)

W3C: World Wide Web Consortium

XML: Extensible Markup Language (W3C XML)

5 Conventions

5.1 Naming convention

It should be noted that the Digital Item Declaration Model (clause 6) contains the concept names that are used throughout the MPEG-21 standard. As such, this model should be considered to be the “ultimate arbiter” of these MPEG-21 concept names.

5.2 Documentation convention

The semantics of each entity in the Digital Item Declaration Model is specified using the constructs provided by EBNF [4], and is shown in this document using a specific font and background:

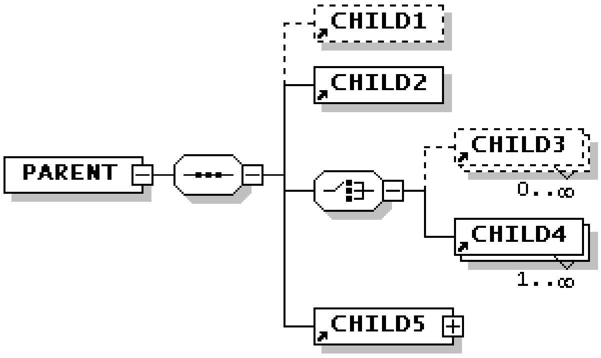
```
entity ::= (part1 | part2)+ part3*
```

The syntax of each element in the Digital Item Declaration Representation is specified using the constructs provided by XML Schema [2].

Element names and attribute names in the representation are in SMALL CAPS. Throughout the document, *italics* are used when referring to entities defined in the Digital Item Declaration Model (see clause 4), hereafter known as the Model.

The syntax of each element in the Digital Item Declaration representation is specified using the following format.

Table 1 — Example element specification

<table>
<tr><td>Diagram</td><td colspan="3">
</td></tr>
<tr><td>Children</td><td colspan="3"><CHILD1> <CHILD2> <CHILD3> <CHILD4> <CHILD5></td></tr>
<tr><td>Used by</td><td colspan="3"><GRANDPARENT1> <GRANDPARENT2></td></tr>
<tr><td rowspan="2">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>ID</td><td>ID</td><td>A unique ID value, which can be referenced by another element.</td></tr>
<tr><td>Source</td><td colspan="3"><xsd:element name="PARENT">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="CHILD1" minOccurs="0"/>
            <xsd:element ref="CHILD2"/>
            <xsd:choice>
                <xsd:element ref="CHILD3" minOccurs="0" maxOccurs="unbounded"/>
                <xsd:element ref="CHILD4" minOccurs="1" maxOccurs="unbounded"/>
            </xsd:choice>
            <xsd:element ref="CHILD5"/>
        </xsd:sequence>
        <xsd:attribute name="ID" type="xsd:id"/>
    </xsd:complexType>
</xsd:element></td></tr>
</table>

The Language Definition clause contains syntax diagrams for each element. Here is an example syntax diagram with annotations:

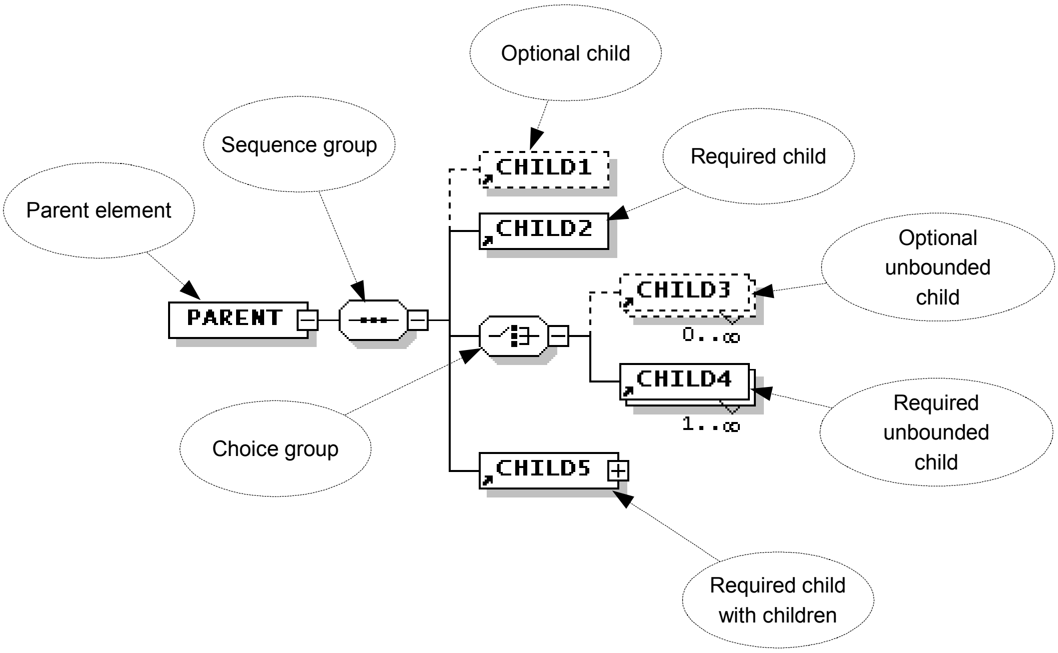


Figure 1 — Example element syntax diagram

Non-normative examples are included in separate clauses, and are shown in this document using a separate font and background:

```
<Example attribute1="example attribute value">
  <Element1>example element content</Element1>
</Example>
```

5.3 Namespace prefix conventions

This document makes use of vocabularies from several XML namespaces (where the definition of an XML namespace is as specified in W3C XMLNAMES [7]). Qualified Names are written with a namespace prefix followed by a colon followed by the local part of the Qualified Name as shown in the following example.

EXAMPLE `didl:DIDL`

For the purposes of this document the Table below gives the namespace prefixes associated with the identified namespaces.

Table 2 — Namespace prefixes

Namespace prefix	Namespace
didmodel	urn:mpeg:mpeg21:2002:02-DIDMODEL-NS
didl	urn:mpeg:mpeg21:2002:02-DIDL-NS
dii	urn:mpeg:mpeg21:2002:01-DII-NS
mx	urn:mpeg:mpeg21:2003:01-REL-MX-NS
sx	urn:mpeg:mpeg21:2003:01-REL-SX-NS
dia	urn:mpeg:mpeg21:2003:01-DIA-NS
diac	urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS

mpeg7	urn:mpeg:mpeg7:schema:2001
dsig	http://www.w3.org/2000/09/xmldsig#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
xml	http://www.w3.org/XML/1998/namespace
xi	http://www.w3.org/2001/XInclude
xsd	http://www.w3.org/2001/XMLSchema
xsi	http://www.w3.org/2001/XMLSchema-instance
dc	http://purl.org/dc/elements/1.1/
uaprof	http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405
foo	This namespace prefix is used for demonstration only.

6 Digital Item Declaration Model

6.1 Purpose and Overview

The purpose of this clause is to describe a set of abstract terms and concepts to form a useful model for defining Digital Items. Within this model, a Digital Item is the digital representation of an Asset, and as such, it is the unit that is acted upon (managed, described, exchanged, collected, etc.) within the model. The goal of this model is to be as flexible and general as possible, while providing for the "hooks" that enable higher level functionality. This, in turn, allows the model to serve as a key foundation in the building of higher level models in other MPEG-21 elements (such as Identification or IPMP). This model specifically does not define a language in and of itself. Instead, the model helps to provide a common set of abstract concepts and terms that can be used to define such a scheme, or to perform mappings between existing schemes capable of Digital Item Declaration, for comparison purposes.

6.2 Abstract Model

6.2.1 Entity Descriptions

In the following descriptions, the defined entities in *italics* are intended to be unambiguous terms within this model. The prose descriptions define the semantic "meaning" of the terms, and the EBNF representations define the precise intended relationship or structure between terms within the model.

6.2.2 *container*

A *container* is a structure that allows *items* and/or *containers* to be grouped. These groupings of *items* and/or *containers* can be used to form logical packages (for transport or exchange) or logical shelves (for organization). *Descriptors* allow for the "labelling" of *containers* with information that is appropriate for the purpose of the grouping (e.g. delivery instructions for a package, or category information for a shelf).

It should be noted that a *container* itself is not an *item*; *containers* are groupings of *items* and/or *containers*.

```
container ::= descriptor* container* item*
```

6.2.3 *item*

An *item* is a grouping of sub-*items* and/or *components* that are bound to relevant *descriptors*. *Descriptors* contain information about the *item*, as a representation of an Asset. *Items* may contain *choices*, which allow them to be customized or configured. *Items* may be conditional (on *predicates* asserted by *selections* defined in the *choices*). An *item* that contains no sub-*items* can be considered a whole -- a logically indivisible Asset. An *item* that does contain sub-*items* can be considered a compilation -- an Asset composed of potentially independent sub-parts. *tems* may also contain *annotations* to their sub-parts.

The relationship between *items* and Digital Items (as defined in ISO/IEC 21000-1:2001, MPEG-21 Vision, Technologies and Strategy) could be stated as follows: *items* are declarative representations of Digital Items.

```
item ::= condition* descriptor* choice* (item | component)* annotation*
```

6.2.4 *component*

A *component* is the binding of a *resource* to a set of *descriptors*. These *descriptors* are information concerning all or part of the specific *resource* instance. Such *descriptors* will typically contain control or structural information about the *resource* (such as bit rate, character set, start points or encryption information) but not information describing the “content” within.

It should be noted that a *component* itself is not an *item*; *components* are <u>building blocks</u> of *items*.

```
component ::= condition* descriptor* resource anchor*
```

6.2.5 *anchor*

An *anchor* binds *descriptors* to a *fragment,* which corresponds to a specific location or part of a *resource*. These *descriptors* are information concerning all or part of the *fragment*.

```
anchor ::= condition* descriptor* fragment
```

6.2.6 *descriptor*

A *descriptor* associates information with the enclosing entity. This information may be a *component* (such as a thumbnail of an image, or a text *component*), or a textual *statement*.

NOTE Though a *descriptor* associates information with the enclosing entity, that information does not always directly concern the enclosing entity. The enclosing entity specifies what the associated information concerns. (For example, a *descriptor* in a *component*, associates information with a *component* that concerns the *resource* in that *component*.)

```
descriptor ::= condition* descriptor* (component | statement)
```

6.2.7 *condition*

A *condition* describes the enclosing entity as being optional, and links it to the *selection*(s) that affect its inclusion. Multiple *predicates* within a *condition* are combined as a conjunction (an AND relationship). Any *predicate* may be negated within a *condition*. Multiple *conditions* associated with a given entity are combined as a disjunction (an OR relationship) when determining whether to include the entity.

```
condition ::= predicate+
```

6.2.8 *choice*

A *choice* describes a set of related *selections* that can affect the configuration of an *item*. The *selections* within a *choice* are either exclusive (choose exactly one) or inclusive (choose any number, including all or none).

```
choice ::= condition* descriptor* selection+
```

6.2.9 *selection*

A *selection* describes a specific decision that will affect one or more *conditions* somewhere within an *item*. If the *selection* is chosen, its *predicate* becomes true; if it is not chosen, its *predicate* becomes false; if it is left unresolved, its *predicate* is undecided.

```
selection ::= condition* descriptor* predicate
```

6.2.10 *annotation*

An *annotation* describes a set of information about another identified entity of the model without altering or adding to that entity. The information can take the form of *assertions*, *descriptors*, and *anchors*.

```
annotation ::= assertion* descriptor* anchor*
```

6.2.11 *assertion*

An *assertion* defines a full or partially configured state of a *choice* by asserting true, false or undecided values for some number of *predicates* associated with the *selections* for that *choice*.

```
assertion ::= predicate*
```

6.2.12 *resource*

A *resource* is an individually identifiable Asset such as a video or audio clip, an image, or a textual Asset. A *resource* may also potentially be a physical object. All *resources* shall be locatable via an unambiguous address.

6.2.13 *fragment*

A *fragment* unambiguously designates a specific point or range within a *resource*. *Fragment* may be *resource* type specific.

6.2.14 *statement*

A *statement* is a literal textual value that contains information, but not an Asset. Examples of likely *statements* include descriptive, control, revision tracking or identifying information (such as an identifier as described in ISO/IEC 21000-3).

6.2.15 *predicate*

A *predicate* is an unambiguously identifiable declaration that can be true, false or undecided.

The following diagram is an example showing the most important entities within this model, how they are related, and as such, the hierarchical structure of the Digital Item Declaration Model.

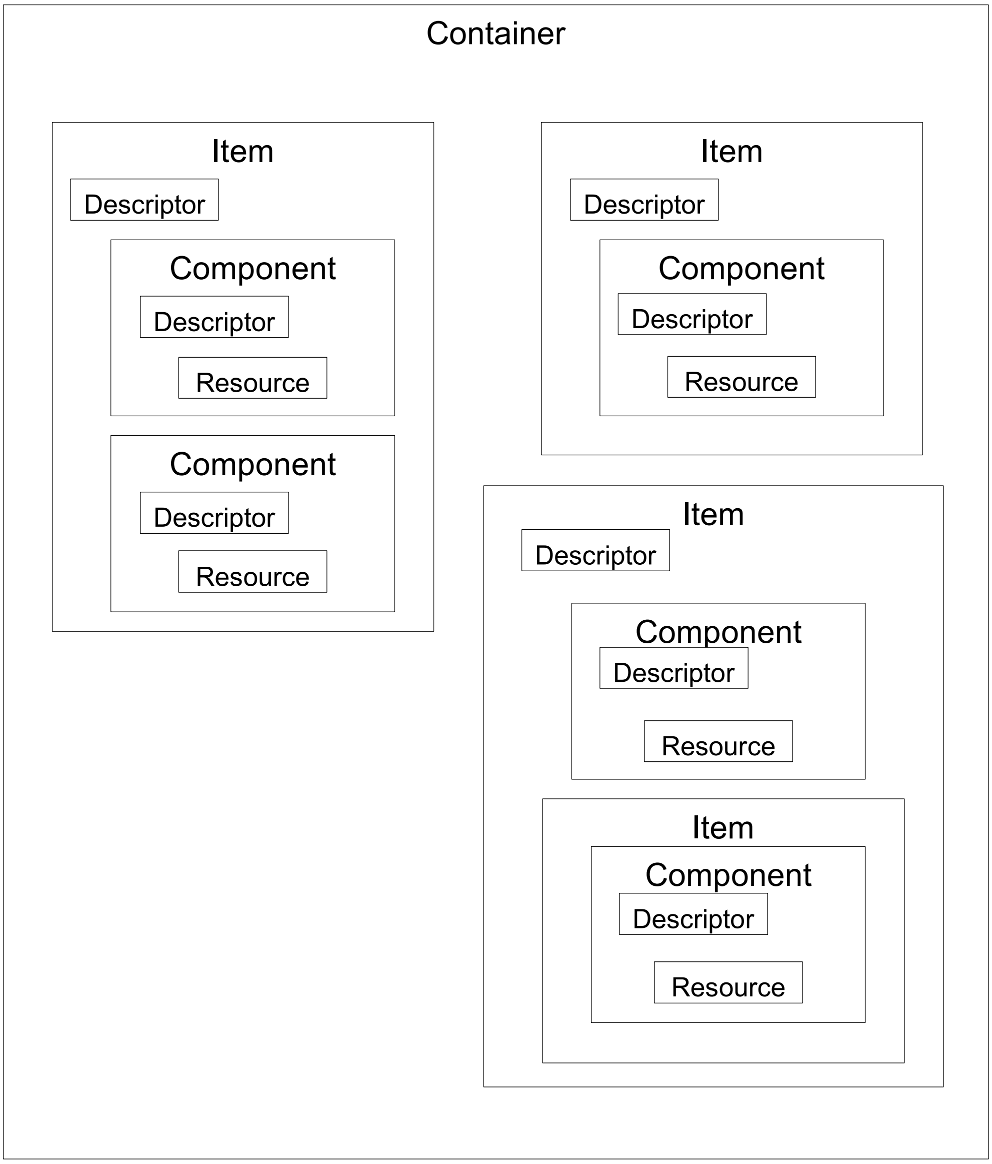


Figure 2 — Example Digital Item Declaration model

7 Digital Item Declaration Representation

7.1 Purpose and Overview

7.1.1 Purpose

The purpose of this clause is to describe the syntax and semantics of the W3C XML representation for declaring Digital Items. The syntax is defined using XML schema (as specified in W3C XMLSCHEMA). The goal is to be as flexible and general as possible, while providing the "hooks" for higher level functionality that will allow it to serve as a key foundation in the building of higher level schema in other MPEG-21 domains (such as Identification or Rights Management). For the purposes of this document, the XML schema syntax descriptions are collectively referred to as DIDL schema.

7.1.2 DIDL Overview

DIDL documents are W3C XML 1.0 [1] documents. The reader is assumed to be familiar with the terms and concepts of XML 1.0.

In addition, DIDL syntax is based on an abstract structure defined in the Digital Item Declaration Model (see clause 6 above). The following abstract entities defined in the Model are each represented in DIDL by a like-named DIDL element:

— *container*

— *item*

— *component*

— *anchor*

— *fragment*

— *descriptor*

— *choice*

— *selection*

— *condition*

— *annotation*

— *assertion*

— *resource*

— *statement*

For example, the abstract *descriptor* entity in the Model is represented in DIDL by the DESCRIPTOR element. Therefore, the reader is likewise assumed to be familiar with the terms and concepts defined in the Model.

A DIDL document consists of a DIDL root element with a single ITEM child element or CONTAINER child element. Thus, a DIDL document can represent either an *item* or a *container*.

In addition, DIDL defines the following special element type that does not correspond to any of the Model entities: DECLARATIONS. This special element is used for specific purposes within DIDL.

The DECLARATIONS element is used to define a set of DIDL elements in a document without actually instantiating them. A declared element (i.e. a child element of a DECLARATIONS element) is not considered to be instantiated unless it is referenced (by an XInclude `include` element).

DIDL makes broad use of XML's ID attribute type. Generally, attributes of this type are used to make an internal association between one DIDL element and another. For example, many DIDL elements have an ID attribute, which makes them available as targets of internal references (see 7.2.3) by XInclude `include` elements, and, in limited cases, available for annotation by ANNOTATION elements. In addition, other attributes of type ID that are not named 'id' are used to make specific kinds of associations between specific elements. For example, the SELECT_ID attribute of the SELECTION element allows CONDITION elements to be associated with specific SELECTIONS. It is strongly recommended that attributes of type ID be assigned globally unique values, in order to avoid collisions when DIDL documents are merged. This is especially important when there are external dependencies on the ID values, such as a signature on the DIDL document, or an external reference from some other document or resource. Finally, it is noted that the use of the ID attribute should not be confused with normative identification mechanisms defined in any other part of ISO/IEC 21000.

7.2 DIDL Definition

7.2.1 Validation

Validating a document against the DIDL schema (as specified in W3C XMLSCHEMA) is necessary, but not sufficient, to determine its validity with respect to DIDL. After a document is validated against the DIDL schema, it shall also be subjected to additional validation rules. These additional rules are given below in the descriptions of the elements to which they pertain.

7.2.2 Canonicalization

Like any XML document, a single logical DIDL document can be manifested in a wide variety of syntactic representations. Although the various syntactic representations each contain a different sequence of characters, they are all logically equivalent. In certain applications, such as generating a digest value for a digital signature on all or part of a DIDL document, it is necessary to define a method for generating a single predictable (deterministic) syntactic representation. Achieving this result for DIDL documents containing internal references requires special consideration. The following canonicalization method takes this consideration into account:

Canonical XML 1.0 [3] (as specified in W3C XMLC14N) with the following additional constraint:

> All internal references are syntactically resolved; that is, the logical replacement is reflected in the Canonical syntactic representation. The XInclude `include` tags corresponding to internal references are removed in this process.

This canonicalization method is identified by the following URN: "urn:mpeg:mpeg21:2002:02:did-canonicalization". This URN identifier is for use by any part of ISO/IEC 21000 that requires canonicalization, or by any other application with similar requirements.

7.2.3 Document modularity

NOTE 1 Modularity of DIDL documents can be achieved by utilising XML Inclusions (XInclude) as specified in W3C XINCLUDE [7]. XInclude can be used to reference elements within a document, or to elements in a different document. The former type of reference is known as an internal reference; the latter is known as an external reference. An internal reference allows a single source to be maintained for an element that occurs in more than one place in a DIDL document. An external reference allows a DIDL document to be split up into multiple linked discrete documents.

NOTE 2 XInclude addresses the same requirements previously addressed by the REFERENCE element in the first edition of this part of ISO/IEC 21000. For this reason no effort is made to enable use of xml:base and xml:lang on DIDL elements other than those for which use of REFERENCE was allowed in the first edition of this part of ISO/IEC 21000.

NOTE 3 There are some differences in the provision of this functionality through the use of XInclude instead of the REFERENCE element. See Annex B for a list of the key differences.

7.2.4 Element Descriptions

The following basic principles apply to all element types:

— Wherever DESCRIPTOR children are allowed, they are always the first children (except where preceded by CONDITION).

7.2.5 <DIDL>

The DIDL element is the root element of a DIDL instance document. The DIDL root element may contain an optional DECLARATIONS element, followed by exactly one CONTAINER or ITEM.

The DIDL element shall include a namespace declaration that declares the DIDL namespace for the root DIDL element and its contents. This is required so that applications will recognize the document as a DIDL document, that is covered by this specification. The DIDL namespace URI is "urn:mpeg:mpeg21:2002:02-DIDL-NS". The "02" represents a serial number that is expected to change as the DIDL schema evolves along with this part of ISO/IEC 21000.

The namespace declaration may take the form of a default namespace declaration, or a prefix-specific namespace declaration, as shown respectively in the following two examples:

EXAMPLE 1 Default namespace declaration

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
...
</DIDL>
```

EXAMPLE 2 Prefix-specific namespace declaration

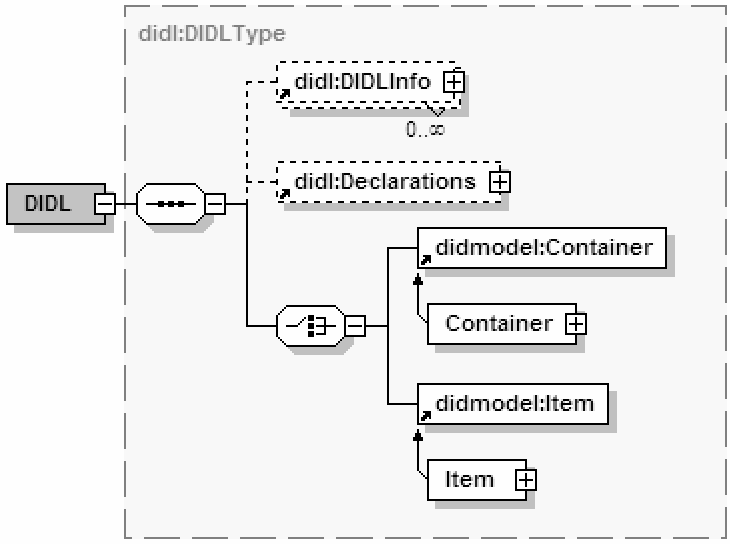
```
<didl:DIDL xmlns:didl="urn:mpeg:mpeg21:2002:02-DIDL-NS">
...
</didl:DIDL>
```

NOTE In this part of ISO/IEC 21000 the namespace prefix `didl` will be used to map to the namespace `urn:mpeg:mpeg21:2002:02-DIDL-NS`, however this mapping is not normative.

The DIDL element may have an optional DIDLDOCUMENTID attribute. This attribute can be used to convey the URI of the DIDL document. Identifiers of the Digital Items declared within the DIDL document shall not be conveyed in this attribute.

The DIDL element may have any attributes from other namespaces. Applications that do not understand any such attributes may ignore those attributes. Since the DIDL element is not part of the DID abstract model, it is only valid for information regarding the DIDL document to be contained in any such attributes. Information about the Digital Item represented by the DID shall not be conveyed in such attributes.

Table 3 — DIDL element syntax

Diagram			
Children	<DIDLInfo><Declarations><Container><Item>		
Attributes	**Name**	**Type**	**Description**
	DIDLDocumentId	anyURI	A URI of the DIDL document.

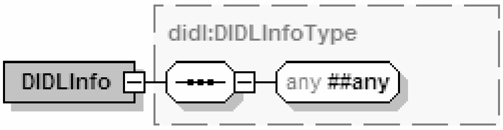
Source

```
<element name="DIDL" type="didl:DIDLType"/>
<complexType name="DIDLType">
 <sequence>
  <element ref="didl:DIDLInfo" minOccurs="0" maxOccurs="unbounded"/>
  <element ref="didl:Declarations" minOccurs="0"/>
  <choice>
   <element ref="didmodel:Container"/>
   <element ref="didmodel:Item"/>
  </choice>
 </sequence>
 <attribute name="DIDLDocumentId" type="anyURI"/>
 <anyAttribute namespace="##other" processContents="lax"/>
</complexType>
```

7.2.6 <DIDLInfo>

The DIDLINFO element allows information applicable only to the DIDL document to be included in the DIDL document. Since it is not part of the DID abstract model, it shall not contain information about the Digital Item represented by the DID. An example of such information might be the digital signature for the DIDL document. Applications that do not understand any information included in a DIDLINFO element may ignore such information.

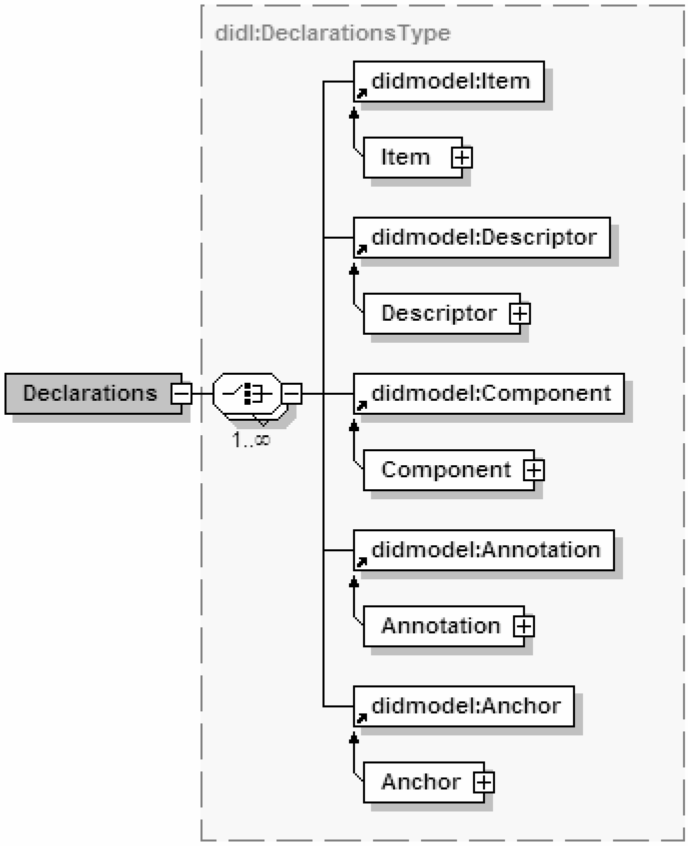
Table 4 — DIDLINFO element syntax

Diagram	
Used by	<DIDL>
Source	<element name="DIDLInfo" type="didl:DIDLInfoType"/> <complexType name="DIDLInfoType"> <sequence> <any namespace="##any" processContents="lax"/> </sequence> </complexType>

7.2.7 <Declarations>

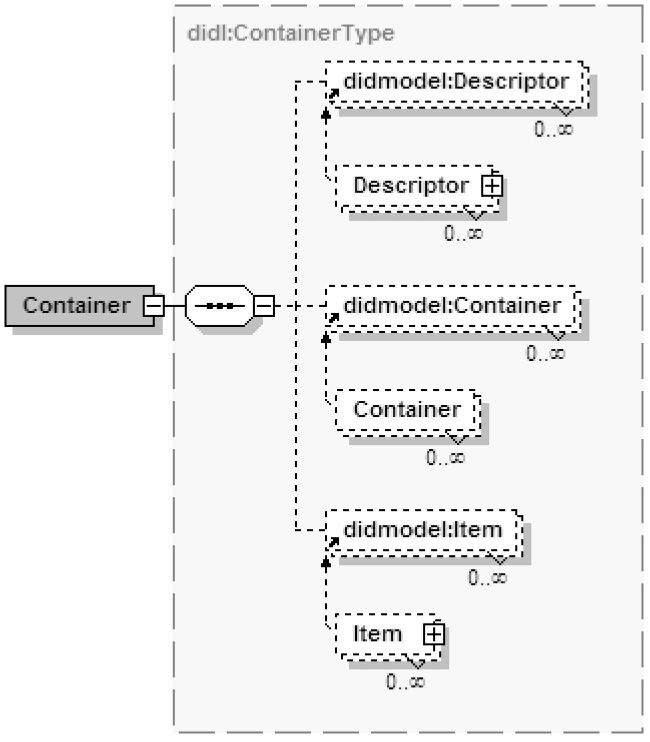
The DECLARATIONS element is used to define a set of DIDL elements - without instantiating them - for later use in a document via an internal reference (see 7.2.3).

Table 5 — DECLARATIONS element syntax

Diagram	
Children	<Item> <Descriptor> <Component> <Annotation> <Anchor>
Used by	<DIDL>

Source	`<element name="Declarations" type="didl:DeclarationsType"/>`
	`<complexType name="DeclarationsType">`
	` <choice maxOccurs="unbounded">`
	`  <element ref="didmodel:Item"/>`
	`  <element ref="didmodel:Descriptor"/>`
	`  <element ref="didmodel:Component"/>`
	`  <element ref="didmodel:Annotation"/>`
	`  <element ref="didmodel:Anchor"/>`
	` </choice>`
	`</complexType>`

Table 6 — CONTAINER element syntax

<table>
<tr><td>Diagram</td><td colspan="3">
</td></tr>
<tr><td>Children</td><td colspan="3"><Descriptor> <Container> <Item></td></tr>
<tr><td>Used by</td><td colspan="3"><DIDL></td></tr>
<tr><td rowspan="2">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>id</td><td>ID</td><td>A unique ID value.</td></tr>
<tr><td>Source</td><td colspan="3"><pre>
 <element name="Container" type="didl:ContainerType"
substitutionGroup="didmodel:Container"/>
 <complexType name="ContainerType">
  <complexContent>
   <extension base="didmodel:ContainerType">
    <sequence>
     <element ref="didmodel:Descriptor" minOccurs="0"
      maxOccurs="unbounded"/>
     <element ref="didmodel:Container" minOccurs="0"
      maxOccurs="unbounded"/>
     <element ref="didmodel:Item" minOccurs="0"
      maxOccurs="unbounded"/>
    </sequence>
    <attributeGroup ref="didl:ID_ATTRS"/>
    <anyAttribute namespace="##other" processContents="lax"/>
   </extension>
  </complexContent>
 </complexType>
</pre></td></tr>
</table>

7.2.8 <Container>

The CONTAINER element represents a *container*. As such, it is a grouping of ITEMS and/or possibly other CONTAINERS, bound with a set of DESCRIPTORS that contain descriptive information about the *container*.

A CONTAINER may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information about the *container* by means of an attribute.

NOTE *descriptor* information can be associated with the *container* by such attributes, as well as by child DESCRIPTOR elements.

7.2.9 <Item>

An ITEM element represents an *item*. As such, it is a grouping of possible sub-ITEMS and/or COMPONENTS, bound to a set of relevant DESCRIPTORS containing descriptive information about the *item*. In addition, an ITEM can be made conditional via a set of CONDITION child elements, made configurable via a set of CHOICE elements, and annotated via a set of ANNOTATION elements.

An ITEM may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information about the *item* by means of an attribute.

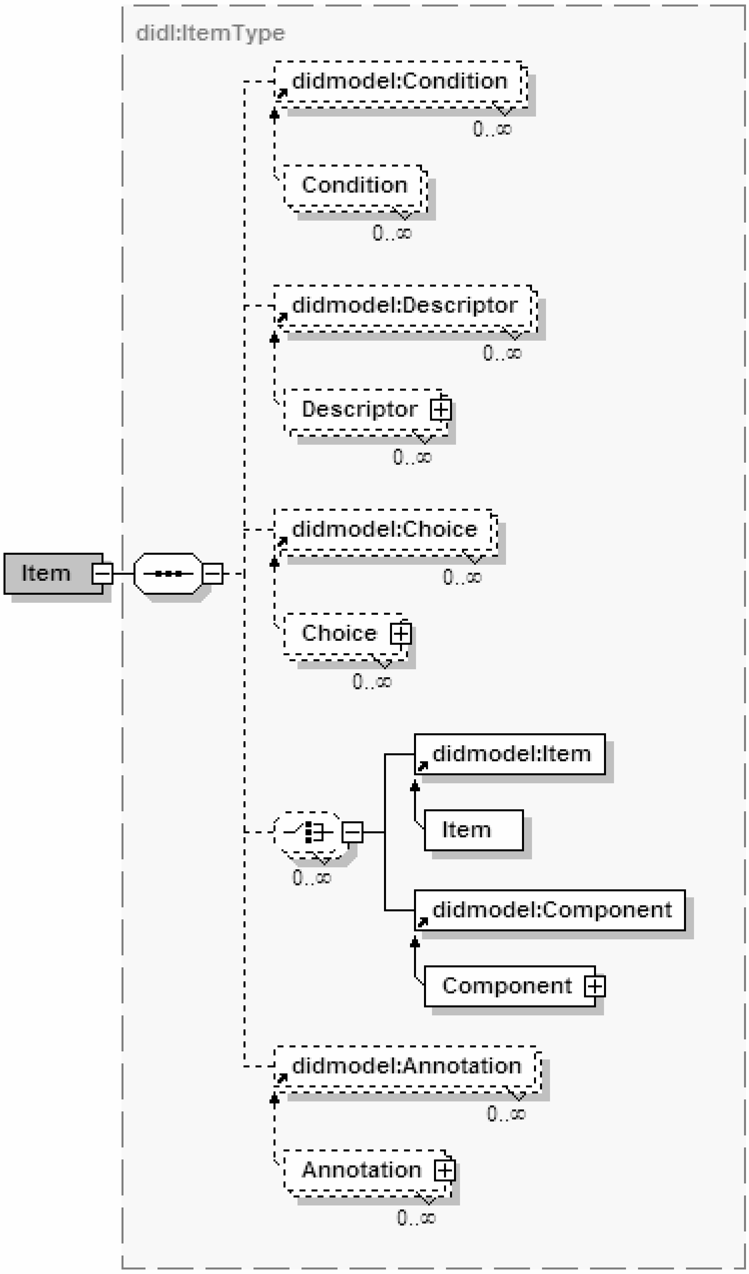
NOTE *descriptor* information can be associated with the *item* by such attributes, as well as by child DESCRIPTOR elements.

Items are intended to be the lowest level of granularity transacted by Users within the MPEG-21 framework.

Validation Rule:

— An ITEM element shall not be conditional on any of its descendant SELECTION elements. In other words, an ITEM shall not contain a CONDITION element specifying a SELECT_ID value that identifies any descendant SELECTION element within the ITEM.

Table 7 — ITEM element syntax

Diagram	
Children	<Condition> <Descriptor> <Choice> <Item> <Component> <Annotation>
Used by	<Declarations> <Item> <DIDL>

<table>
<tr><td rowspan="2">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>id</td><td>ID</td><td>A unique ID value.</td></tr>
</table>

Source

```
<element name="Item" type="didl:ItemType"
 substitutionGroup="didmodel:Item"/>
<complexType name="ItemType">
 <complexContent>
  <extension base="didmodel:ItemType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Choice" minOccurs="0"
     maxOccurs="unbounded"/>
    <choice minOccurs="0" maxOccurs="unbounded">
     <element ref="didmodel:Item"/>
     <element ref="didmodel:Component"/>
    </choice>
    <element ref="didmodel:Annotation" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
```

EXAMPLE The following example illustrates how Containers and Items can be used to represent a *container* of some kind that contains a single composite *item* – a “compilation.” The Container represents the shelf or package, the outermost Item represents the composite *item* as a whole, and the inner Items represent the individual *items* that make up the compilation.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS ">
 <Container>
  <Item>
   <Item>
    .
    .
    .
   </Item>
   <Item>
    .
    .
    .
   </Item>
  </Item>
 </Container>
</DIDL>
```

7.2.10 <Component>

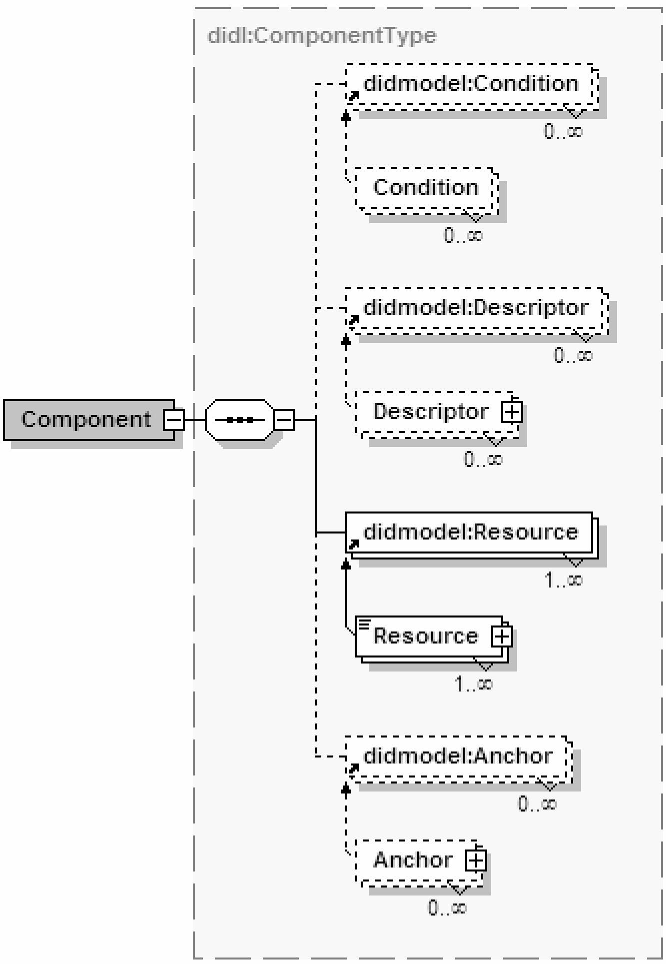
A COMPONENT element represents a *component*. As such, it groups a RESOURCE element with a set of DESCRIPTORS containing descriptive information about the *resource*, plus a set of ANCHORS specifying points or regions of interest in the *resource*. The COMPONENT, being a logical union of a *resource* with relevant descriptive data and *anchors*, is intended to be the basic building block of digital content within a DIDL document.

A COMPONENT may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information about the *resource* by means of an attribute of the COMPONENT.

NOTE *descriptors* containing descriptive information about the *resource* can be associated with the *component* by such attributes, as well as by child DESCRIPTOR elements of the COMPONENT.

If multiple RESOURCE children are present, they are considered bit equivalent and any one of them may be used. An agent may discriminate between them using specific information it has about retrieval from these sources, or using such information present in a DESCRIPTOR.

Table 8 — COMPONENT element syntax

Diagram	

Children	<Condition> <Descriptor> <Resource> <Anchor>		
Used by	<Declarations> <Descriptor> <Item>		
Attributes	**Name**	**Type**	**Description**
	id	ID	A unique ID value.

Source

```
<element name="Component" type="didl:ComponentType"
 substitutionGroup="didmodel:Component"/>
<complexType name="ComponentType">
 <complexContent>
  <extension base="didmodel:ComponentType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Resource" maxOccurs="unbounded"/>
    <element ref="didmodel:Anchor" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
```

7.2.11 <Resource>

A RESOURCE element represents a *resource*. As such, it defines an individually identifiable Asset such as a video or audio clip, an image, an electronic ticket, or a textual Asset.

Normally, a *resource* is defined in a RESOURCE element by reference, by specifying the *resource*'s URI in the REF attribute. The URI identifies the *resource* for the purpose of allowing an application to retrieve the *resource*'s contents. This URI can be a traditional URL, which gives the explicit physical location from which to retrieve the contents, or a more abstract identifier, such as a URN, which identifies the *resource* contents independent from their location.

The data type of the *resource* is identified by the MIMETYPE attribute, which is a concatenation of MIME media-type, sub-type, and parameters, as defined in IETF RFC 2045 (e.g. 'video/mpeg'). The MIMETYPE attribute identifies the data type of the *resource* before any content-encodings specified in the CONTENTENCODING attribute were applied to the *resource*.

The MIME media-type may be modified by the presence of the CONTENTENCODING attribute which specifies the content-encoding as defined in IETF RFC 2616. When present, the value of the CONTENTENCODING attribute indicates what additional content-encodings have been applied to the *resource*, and thus what decoding mechanisms to apply in order to obtain the MIME media-type identified by the MIMETYPE attribute. Content-encoding is primarily used to allow a document to be compressed without losing the identity of its underlying MIME media-type.

If multiple content-encodings have been applied to the *resource*, each content-encoding shall be listed in the value of the CONTENTENCODING attribute in a space-delimited list in the order in which they were applied.

NOTE The token identifying the first content-encoding applied to the unencoded *resource* is provided as the first (left-most) token in the list.

Accepted tokens for use in the value of the CONTENTENCODING attribute are the content-encoding tokens registered by the IANA.

It is also possible to define a *resource* by value, by including the *resource*'s data either as character data or as well-formed XML within the content of the RESOURCE element.

The ENCODING attribute is used to specify the encoding format used to include the *resource* by value.

If the *resource* is included by value and no such encoding is applied, then the ENCODING attribute shall be omitted.

If the ENCODING attribute is present, the value shall be set to 'base64' and the *resource* shall be provided by value using a base64 encoding as specified in IETF RFC 3548.

A RESOURCE may have any attribute from other namespaces. Such attributes may provide additional information about the RESOURCE.

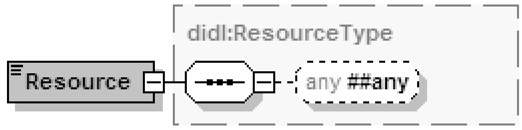
Validation Rule:

— If the REF attribute is specified, then the RESOURCE shall not be defined by value, and vice versa.

— If the *resource* is not included by value the ENCODING attribute shall be omitted.

— If the ENCODING attribute is present, the value shall be set to 'base64'.

Table 9 — RESOURCE element syntax

Diagram			
Used by	<Component>		
Attributes	**Name**	**Type**	**Description**
	mimeType	string	Specifies the data type of the *resource*, before any of the content-encodings specified in the CONTENTENCODING attribute were applied to the *resource*, as a concatenation of MIME media-type, sub-type, and parameters, as defined in IETF RFC 2045.
	ref	anyURI	The URI value that identifies the *resource*. If the REF attribute is omitted, then the element shall contain the *resource* inline as character data or well-formed XML.
	encoding	string	Specifies the encoding format used to include the *resource* by value in the DIDL XML document.
	contentEncoding	NMTOKENS	Specifies the content-encoding(s) as defined in IETF RFC 2616. A content-encoding value is used as a modifier to the MIME media-type. When present, its value indicates what additional content-encodings have been applied to the *resource*.

Source

```
<element name="Resource" type="didl:ResourceType"
 substitutionGroup="didmodel:Resource"/>
<complexType name="ResourceType" mixed="true">
 <complexContent mixed="true">
  <extension base="didmodel:ResourceType">
   <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0"/>
   </sequence>
   <attribute name="mimeType" type="string" use="required"/>
   <attribute name="ref" type="anyURI"/>
   <attribute name="encoding" type="string"/>
   <attribute name="contentEncoding" type="NMTOKENS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
```

EXAMPLE 1 COMPONENTS and RESOURCES are added to the example document from 0 to illustrate how they might be used. Each individual *item*, represented by each inner ITEM, contains a *component* that comprises a single photographic *resource*. Thus, this document could be used to represent a shelf (the CONTAINER) containing a photo album (the outermost ITEM) made up of two individual photographs (the two inner ITEMS).

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Container>
  <Item>
   <Item>
    <Component>
     <Resource ref="myFirstPicture.jpg" mimeType="image/jpeg" />
    </Component>
   </Item>
   <Item>
    <Component>
     <Resource ref="mySecondPic.bmp" mimeType="image/x-ms-bmp" />
    </Component>
   </Item>
  </Item>
 </Container>
</DIDL>
```

EXAMPLE 2 This second example shows the use of the ENCODING and CONTENTENCODING attributes. This document could be used to a represent an audio recording of a seminar (the outermost ITEM) split up as several individual recordings (the inner ITEMS). The *component* of the first sub-ITEM contains a *resource* defined by value and the *resource*'s data is base64 encoded, therefore the ENCODING attribute is present and the value is set to 'base64'. The *component* of the second sub-ITEM contains a *resource* defined by reference, therefore the ENCODING attribute is not present. However the resource data is "gzip" compressed (as per IETF RFC 1952), therefore the CONTENTENCODING attribute is present with a value of 'gzip' (see IETF RFC 2616). The *component* of the third sub-ITEM contains a *resource* defined by value and whose data is base64 encoded, therefore the ENCODING attribute is present and the value is set to 'base64'. This *resource* is also "gzip" compressed, therefore the CONTENTENCODING attribute is also present with a value of 'gzip'. In this case the *resource* is a "gzip" compressed MP3 audio recording. To include the *resource*'s data within the content of the RESOURCE element, the data (i.e. the "gzip" compressed MP3 data) has been base64 encoded. Thus to access the MP3 data, first a base64 decoding is applied to the content of the RESOURCE element, then a "gzip" decoding is applied to the resource data. If more than one content encoding is specified (that is if the CONTENTENCODING attribute value contains more than one content-encoding token), then the indicated decodings would be applied in reverse order (i.e. from right to left) of their ordering in the CONTENTENCODING attribute value.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item>
  <Item>
   <Component>
    <Resource mimeType="audio/mpeg" encoding="base64">
UNJVYfkr0HWjNKtYi0kUZZda0vN55WVo3ZsP6SJ6V3mCrpiXG47Ypae3WYIvUPvfcGT0zyFm
FGCojJiQSrn1S652JCkryMULsDZjDiIvLtBwachRUsFkSAk//qSwIuU6wAa2h12mMYAAtPDb
…
5MYsADSwPJJrrCMywQ6mi2Qhk9cmSq0FRE8Cs6gv=
    </Resource>
   </Component>
  </Item>
  <Item>
   <Component>
    <Resource ref="session1.gz" mimeType="audio/mpeg"
         contentEncoding="gzip"/>
   </Component>
  </Item>
  <Item>
   <Component>
    <Resource mimeType="audio/mpeg" encoding="base64"
         contentEncoding="gzip">
FaCLWxjYQElqZt7yLPZnQvsXUOc5ghUOWUCVCdXQ/GyBo0xLJYy2bkaYnBoPjpzBiPHAR6bd
MJITucuHPODgpM22Jtohdov/OYEVJmoDxEkrqZQKV6vJlQVetXVBEgIQE6wwhSygjFq7JBcY
…
/wTBNzYWURYkBaAnjOSz82qBB/XkA4j9OU95fsJthcREFWpm2Kz65euqlsJuI0EJD/2rjx=
    </Resource>
   </Component>
  </Item>
 </Item>
</DIDL>
```

7.2.12 <Descriptor>

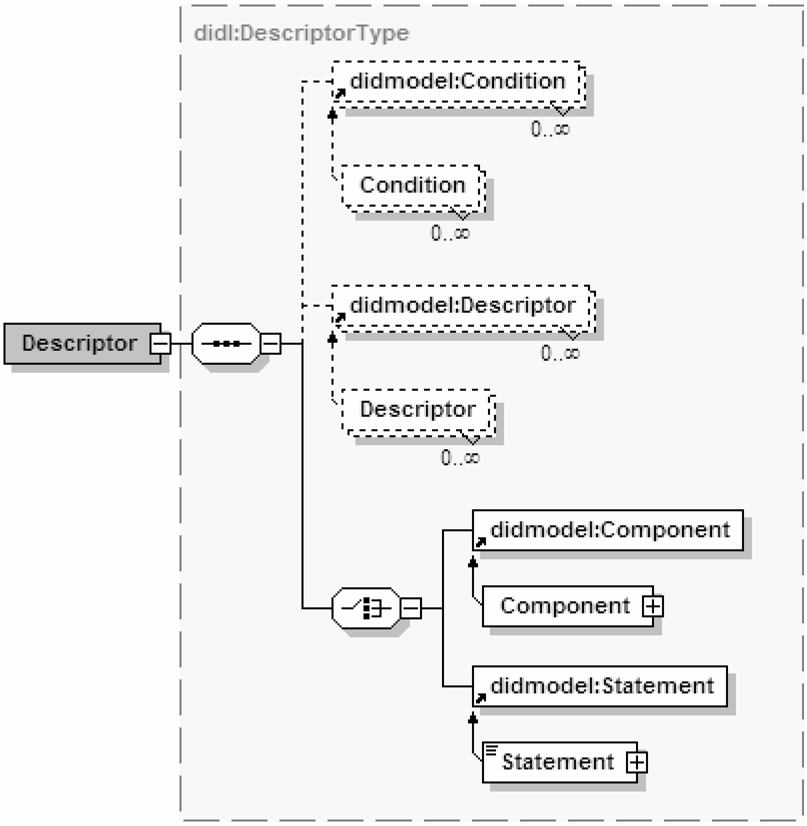
A DESCRIPTOR represents a *descriptor*. As such, it associates information with its parent element. This information may be contained in a COMPONENT element, or a STATEMENT element.

Typically, a DESCRIPTOR is used to associate descriptive data with a parent element. Descriptive data may take the form of a *component* or a *statement*. An example of a *component* containing descriptive data is that of a thumbnail version of a photographic image. An example of a *statement* containing descriptive data is that of a simple textual description, or meta-data, such as the title and author of an Asset.

A DESCRIPTOR may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information about the *descriptor* represented by the DESCRIPTOR with such an attribute.

NOTE *descriptor* information can be associated with another *descriptor* by such attributes, as well as by child DESCRIPTOR elements of the parent DESCRIPTOR.

Table 10 — DESCRIPTOR element syntax

<table>
<tr><td rowspan="1">Diagram</td><td colspan="3">
</td></tr>
<tr><td>Children</td><td colspan="3"><Condition> <Descriptor> <Component> <Statement></td></tr>
<tr><td>Used by</td><td colspan="3"><Anchor> <Annotation> <Choice> <Component> <Declarations> <Descriptor> <Item> <Selection></td></tr>
<tr><td rowspan="2">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>id</td><td>ID</td><td>A unique ID value.</td></tr>
</table>

Source	<pre><element name="Descriptor" type="didl:DescriptorType"
	 substitutionGroup="didmodel:Descriptor"/>
	<complexType name="DescriptorType">
	 <complexContent>
	  <extension base="didmodel:DescriptorType">
	   <sequence>
	    <element ref="didmodel:Condition" minOccurs="0"
	     maxOccurs="unbounded"/>
	    <element ref="didmodel:Descriptor" minOccurs="0"
	     maxOccurs="unbounded"/>
	    <choice>
	     <element ref="didmodel:Component"/>
	     <element ref="didmodel:Statement"/>
	    </choice>
	   </sequence>
	   <attributeGroup ref="didl:ID_ATTRS"/>
	   <anyAttribute namespace="##other" processContents="lax"/>
	  </extension>
	 </complexContent>
	</complexType></pre>

7.2.13 <Statement>

A STATEMENT represents a *statement*. As such, it defines a textual value that contains information, but not an Asset. Examples of likely STATEMENTS include descriptive, control, revision tracking or identifying information.

A STATEMENT may contain any data format, including plain text and various machine-readable formats such as well-formed XML. The data type of the STATEMENT is identified by the MIMETYPE attribute, which is a concatenation of MIME media-type, sub-type, and parameters, as defined in IETF RFC 2045 (e.g. 'text/xml'). The MIMETYPE attribute identifies the data type of the *statement* before any content-encodings specified in the CONTENTENCODING attribute were applied to the *statement*.

The MIME media-type may be modified by the presence of the CONTENTENCODING attribute which specifies the content-encoding as defined in IETF RFC 2616. When present, the value of the CONTENTENCODING attribute indicates what additional content-encodings have been applied to the *statement*, and thus what decoding mechanisms to apply in order to obtain the MIME media-type identified by the MIMETYPE attribute. Content-encoding is primarily used to allow a document to be compressed without losing the identity of its underlying MIME media-type.

If multiple content-encodings have been applied to the *statement*, each content-encoding shall be listed in the value of the CONTENTENCODING attribute in a space-delimited list in the order in which they were applied.

NOTE 1 The token identifying the first content-encoding applied to the unencoded *statement* is provided as the first (left-most) token in the list.

Accepted tokens for use in the value of the CONTENTENCODING attribute are the content-coding tokens registered by the IANA.

The ENCODING attribute is used to specify the encoding format used to include the *statement* by value within the content of the STATEMENT element.

If the *statement* is included by value and no such encoding is applied, then the ENCODING attribute shall be omitted.

If the ENCODING attribute is present, the value shall be set to 'base64' and the *statement* shall be provided by value using a base64 encoding as specified in IETF RFC 3548.

A *statement* may also be defined in a STATEMENT element by reference, by specifying the *statement's* URI in the REF attribute. The URI identifies the *statement* for the purpose of allowing an application to retrieve the *statement's* contents. This URI can be a traditional URL, which gives the explicit physical location from which to retrieve the contents, or a more abstract identifier, such as a URN, which identifies the *statement* contents independent from their location.

A STATEMENT may have any attribute from other namespaces. Such attributes may provide additional information about the STATEMENT.

A STATEMENT containing plain text can be used to associate a human-readable description with the parent element. A STATEMENT containing data in some machine-readable format (such as well-formed XML) can be used to express application-specific metadata in an application-specific form. The machine-readable type of STATEMENT is intended to preserve associations of application-specific meta-data with various elements, without requiring that all applications handling a document be able to process the meta-data. For example, a STATEMENT that binds a proprietary type of meta-data with an Item may be understandable only by a particular product of a particular company, but still any product that handles DIDL documents will preserve the association.

NOTE 2 To ensure interoperability among different applications, further specification beyond the scope of this part of ISO/IEC 21000 may be required, in particular for the interoperable use of plain text STATEMENTS.

NOTE 3 Although many of the examples given in this document use plain text for Statement contents, it should be understood that the Statement is primarily intended to carry machine-interpretable formats (such as XML-based formats).

NOTE 4 For STATEMENTS containing well-formed XML, the grammar (i.e. the schema) of the XML fragment contained within the STATEMENT is identified by the namespace of the fragment.

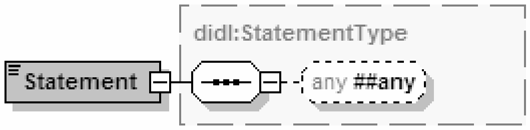
Validation Rule:

— If the REF attribute is specified, then the STATEMENT shall not be defined by value, and vice-versa.

— If the statement is not included by value the encoding attribute shall be omitted.

— If the encoding attribute is present, the value shall be set to 'base64'.

Table 11 — STATEMENT element syntax

Diagram			
Used by	<Descriptor>		
Attributes	**Name**	**Type**	**Description**
	mimeType	string	Specifies the data type of the STATEMENT as a concatenation of MIME media-type, sub-type, and parameters, as defined in IETF RFC 2045.
	ref	anyURI	The URI value that identifies the *statement*. If the REF attribute is omitted, then the element shall contain the *statement* inline as character data or well-formed XML.
	encoding	string	Specifies the encoding format used to include the *statement* by value in the DIDL XML document.
	contentEncoding	NMTOKENS	Specifies the content-encoding(s) as defined in IETF RFC 2616. A content-encoding value is used as a modifier to the MIME media-type. When present, its value indicates what additional content-encodings have been applied to the *statement*.

Source

```
<element name="Statement" type="didl:StatementType"
 substitutionGroup="didmodel:Statement"/>
<complexType name="StatementType" mixed="true">
 <complexContent mixed="true">
  <extension base="didmodel:StatementType">
   <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0"/>
   </sequence>
   <attribute name="mimeType" type="string" use="required"/>
   <attribute name="ref" type="anyURI"/>
   <attribute name="encoding" type="string"/>
   <attribute name="contentEncoding" type="NMTOKENS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
```

EXAMPLE DESCRIPTORS with STATEMENTS are now added to the example from 7.2.9 that express metadata. Here, metadata is expressed in plain (human readable) text, with a custom XML schema, and in an XML-based standardized machine-interpretable format (MPEG-7 in this case). This shows how the STATEMENT can express metadata in any format, and how differently-formatted metadata can coexist. The first pair of DESCRIPTORS gives a description for the CONTAINER. Note that the first DESCRIPTOR has a child DESCRIPTOR that explicitly indicates its relationship with the parent CONTAINER. The second DESCRIPTOR pair gives a title for the photo album. The third and fourth DESCRIPTOR pairs give captions for each of the individual photos in the album using three different mechanisms.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS"
   xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
   xmlns:foo="http://www.bar.org/title-schema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <Container id="C1">
  <Descriptor id="D1">
    <Descriptor>
      <Statement mimeType="text/xml">
        <Relation xsi:type="mpeg7:RelationType" name="titleOf">
          <mpeg7:Argument idref="D1"/>
          <mpeg7:Argument idref="C1"/>
        </Relation>
      </Statement>
    </Descriptor>
   <Statement mimeType="text/plain">My Photo Albums</Statement>
  </Descriptor>
  <Descriptor>
   <Statement mimeType="text/xml">
    <mpeg7:Mpeg7>
     <mpeg7:DescriptionUnit xsi:type="mpeg7:CreationInformationType">
      <mpeg7:Creation>
       <mpeg7:Title>
        My Photo Albums
       </mpeg7:Title>
      </mpeg7:Creation>
     </mpeg7:DescriptionUnit>
    </mpeg7:Mpeg7>
   </Statement>
  </Descriptor>
  <Item>
   <Descriptor>
    <Statement mimeType="text/xml">
     <foo:TITLE>Photo Album #1</foo:TITLE>
    </Statement>
   </Descriptor>
   <Descriptor>
    <Statement mimeType="text/xml">
     <mpeg7:Mpeg7>
      <mpeg7:DescriptionUnit xsi:type="mpeg7:CreationInformationType">
       <mpeg7:Creation>
        <mpeg7:Title>
         Photo Album #1
        </mpeg7:Title>
       </mpeg7:Creation>
      </mpeg7:DescriptionUnit>
     </mpeg7:Mpeg7>
    </Statement>
   </Descriptor>
   <Item>
    <Descriptor>
     <Statement mimeType="text/plain">
      Johnny’s first day at school
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <foo:CAPTION>Johnny’s first day at school</foo:CAPTION>
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
```



```
      <mpeg7:Mpeg7>
       <mpeg7:DescriptionUnit xsi:type="mpeg7:CreationInformationType">
        <mpeg7:Creation>
         <mpeg7:Abstract>
          <mpeg7:FreeTextAnnotation>
           Johnny’s first day at school
          </mpeg7:FreeTextAnnotation>
         </mpeg7:Abstract>
        </mpeg7:Creation>
       </mpeg7:DescriptionUnit>
      </mpeg7:Mpeg7>
     </Statement>
    </Descriptor>
    <Component>
     <Resource ref="myFirstPicture.jpg" mimeType="image/jpeg"/>
    </Component>
   </Item>
   <Item>
    <Descriptor>
     <Statement mimeType="text/plain">
      Jane’s first day at school
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <foo:CAPTION>Jane’s first day at school</foo:CAPTION>
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <mpeg7:Mpeg7>
       <mpeg7:DescriptionUnit xsi:type="mpeg7:CreationInformationType">
        <mpeg7:Creation>
         <mpeg7:Abstract>
          <mpeg7:FreeTextAnnotation>
           Janes’s first day at school
          </mpeg7:FreeTextAnnotation>
         </mpeg7:Abstract>
        </mpeg7:Creation>
       </mpeg7:DescriptionUnit>
      </mpeg7:Mpeg7>
     </Statement>
    </Descriptor>
    <Component>
     <Resource ref="mySecondPic.bmp" mimeType="image/x-ms-bmp"/>
    </Component>
   </Item>
  </Item>
 </Container>
</DIDL>
```

7.2.14 <Anchor>

An ANCHOR element represents an *anchor*. As such, it binds a set of DESCRIPTORS to a specific location or range within the *resource* identified by the RESOURCE element within the parent COMPONENT element. The *fragment* part of the *anchor* entity in the Model is represented in an ANCHOR element by the FRAGMENT child element. If the *resource* is defined by value the FRAGMENT child element specifies the desired location or part of interest within the associated *resource.*

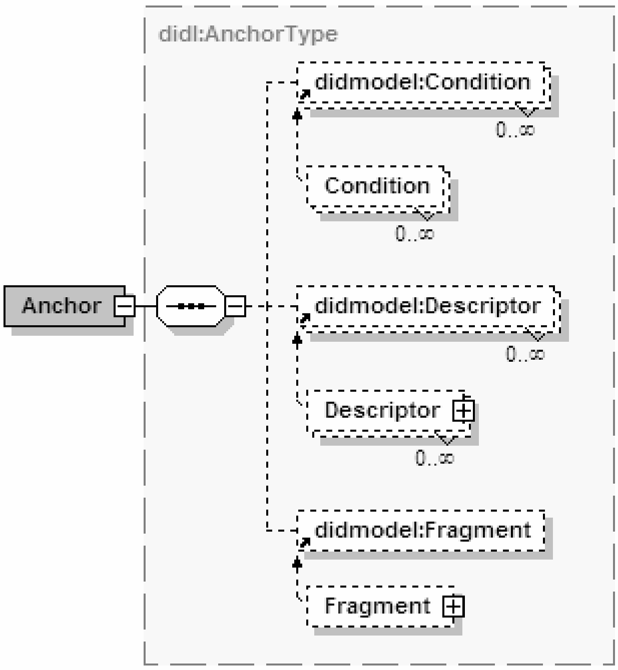
An ANCHOR may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information bound by the *anchor* to the *fragment* within the *resource* by means of an attribute of the ANCHOR.

NOTE *descriptor* information can be bound to the *fragment* within the *resource* by such attributes, as well as by child DESCRIPTOR elements of the ANCHOR.

The ID attribute, in addition to its normal usage as the target of an XInclude `include`, can be used as a reference target by an external *resource*, such as the *resource* with which the ANCHOR is associated. This allows for a two-way linkage between an *anchor* and a *resource*. In practice, as with all attributes of type ID, it is recommended that the value of this ID attribute be made globally unique, so that it never needs to be changed. Since DIDL agents will not necessarily be able to edit any *resource*, this is the only way to guarantee that the two-way linkage is preserved.

Multiple ANCHORS within an ITEM may have the same PRECEDENCE attribute values. In such cases, the answer to the question which ANCHOR has the higher precedence (in other words, which ANCHOR is default for the ITEM) will be determined at the application-level.

Table 12 — ANCHOR element syntax

Diagram	
Children	<Condition> <Descriptor>.<Fragment>
Used by	<Annotation> <Component> <Declarations>

	Name	Type	Description
Attributes	precedence	unsignedInt	An unsigned integer value indicating the relative ranking of this ANCHOR among the other *anchors* in an ITEM. The ANCHOR with the highest precedence value is the default *anchor* for the ITEM. If this attribute is omitted, the precedence of the *anchor* is zero.
	id	ID	A unique ID value.

Source

```
<element name="Anchor" type="didl:AnchorType"
 substitutionGroup="didmodel:Anchor"/>
<complexType name="AnchorType">
 <complexContent>
  <extension base="didmodel:AnchorType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Fragment" minOccurs="0"/>
   </sequence>
   <attribute name="precedence" type="unsignedInt"/>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
```

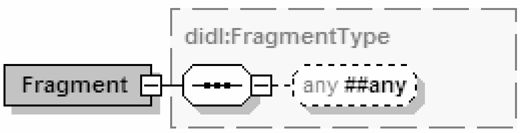
7.2.15 <Fragment>

A FRAGMENT element represents a *fragment*. As such, it unambiguously designates a specific location or part within a *resource*. The FRAGMENTID attribute contains a fragment identifier as defined in IETF RFC 3986, and that specifies the desired location or part of interest within the *resource*. The FRAGMENT element may also contain metadata that locates the desired location or part of interest within the *resource*.

The FRAGMENT element may specify both a FRAGMENTID and metadata. In such cases, the metadata further locates a sub-location or sub-part of interest within the location or part identified by the FRAGMENTID. If the metadata is not consistent with the fragment ID, then the location or part of interest designated by the FRAGMENT element is undefined.

If the *resource* is defined by value the FRAGMENT element specifies the desired location or part of interest within the *resource.*

Table 13 — FRAGMENT element syntax

Diagram			
Used by	<Anchor>		
Attributes	**Name**	**Type**	**Description**
	fragmentId	string	A string specifying a fragment identifier as defined in IETF RFC 3986 that locates the part of interest within the associated *resource*.
Source	(see below)		

```
<element name="Fragment" type="didl:FragmentType"
 substitutionGroup="didmodel:Fragment"/>
<complexType name="FragmentType">
 <complexContent>
  <extension base="didmodel:FragmentType">
   <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0"/>
   </sequence>
   <attribute name="fragmentId" type="string"/>
  </extension>
 </complexContent>
</complexType>
```

EXAMPLE 1 This example is a single ITEM showing how multiple ANCHORS could be used on a single RESOURCE within a COMPONENT. The first *anchor* attaches a description “The whole session” to the beginning of the audio clip. The second *anchor* attaches a different description “Jim’s killer drum solo” to a time-point of 17 minutes and 30 seconds past the beginning of the clip using a fragment identifier for MPEG audio as specified by ISO/IEC 21000-17. The assigned precedence values tell a user interface to display the first *anchor* before the second one when displaying the list of *anchor*s, and to make the first *anchor* the default.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item>
  <Component>
    <Resource ref="JimsGarageBand.mp3" mimeType="audio/mpeg" />
    <Anchor precedence="200">
     <Descriptor>
      <Statement mimeType="text/plain">The whole session</Statement>
     </Descriptor>
    </Anchor>
    <Anchor precedence="100">
     <Descriptor>
      <Statement mimeType="text/plain">
       Jim’s killer drum solo
      </Statement>
     </Descriptor>
     <Fragment fragmentId="mp(/~time('npt','1050')"/>
    </Anchor>
  </Component>
 </Item>
</DIDL>
```

EXAMPLE 2 This example is a single ITEM showing how media locator metadata from ISO/IEC 15938 (MPEG-7) can be used to locate a fragment and to further locate a sub-fragment of an identified fragment. The first *anchor* attaches a semantic description “Garden area” (also using ISO/IEC 15938) to the polygonal area of the building schematic SVG image. The second *anchor* attaches a semantic description “AV Equipment storage” to a rectangular box which is itself a sub-fragment of the graphic element identified by "ConferenceRoom", which is an SVG URI fragment identifier using the bare name form.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS"
   xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001">
 <Item>
  <Component>
    <Resource ref="schematics.svg" mimeType="image/svg+xml" />
    <Anchor>
     <Descriptor>
      <Statement mimeType="text/xml">
       <mpeg7:Description xsi:type="mpeg7:SemanticDescriptionType">
        <mpeg7:Semantics>
         <mpeg7:Label>
          <mpeg7:Name> Garden area </mpeg7:Name>
          <mpeg7:Definition>
           <mpeg7:FreeTextAnnotation>
            Garden area for employee relaxation
           </mpeg7:FreeTextAnnotation>
          </mpeg7:Definition>
         </mpeg7:Label>
        </mpeg7:Semantics>
       </mpeg7:Description>
      </Statement>
     </Descriptor>
     <Fragment>
      <mpeg7:Mpeg7>
       <mpeg7:Description xsi:type="mpeg7:ContentEntityType">
        <mpeg7:MultimediaContent xsi:type="mpeg7:ImageType">
         <mpeg7:Image>
          <mpeg7:SpatialLocator>
           <mpeg7:Polygon>
            <mpeg7:Coords mpeg7:dim="2 5"> 5 25 10 20 15 15 10 10 5
15</mpeg7:Coords>
           </mpeg7:Polygon>
```



```
          </mpeg7:SpatialLocator>
         </mpeg7:Image>
        </mpeg7:MultimediaContent>
       </mpeg7:Description>
      </mpeg7:Mpeg7>
     </Fragment>
    </Anchor>
    <Anchor>
     <Descriptor>
      <Statement mimeType="text/xml">
       <mpeg7:Description xsi:type="mpeg7:SemanticDescriptionType">
        <mpeg7:Semantics>
         <mpeg7:Label>
          <mpeg7:Name> AV Equipment Storage </mpeg7:Name>
          <mpeg7:Definition>
           <mpeg7:FreeTextAnnotation>
            Cupboard for audio visual equipment storage
           </mpeg7:FreeTextAnnotation>
          </mpeg7:Definition>
         </mpeg7:Label>
        </mpeg7:Semantics>
       </mpeg7:Description>
      </Statement>
     </Descriptor>
     <Fragment fragmentId="ConferenceRoom">
      <mpeg7:Mpeg7>
       <mpeg7:Description xsi:type="mpeg7:ContentEntityType">
        <mpeg7:MultimediaContent xsi:type="mpeg7:ImageType">
         <mpeg7:Image>
          <mpeg7:SpatialLocator>
           <mpeg7:Box mpeg7:dim="1 4"> 12 16 30 40 </mpeg7:Box>
          </mpeg7:SpatialLocator>
         </mpeg7:Image>
        </mpeg7:MultimediaContent>
       </mpeg7:Description>
      </mpeg7:Mpeg7>
     </Fragment>
    </Anchor>
  </Component>
 </Item>
</DIDL>
```

7.2.16 <Choice>

A CHOICE element represents a *choice*. As such, it encapsulates a set of related SELECTIONS that can affect the configuration of an ITEM. The optional MINSELECTIONS and MAXSELECTIONS attributes specify the number of SELECTIONS that are required to be made for a CHOICE to be validly resolved. For example, if MINSELECTIONS and MAXSELECTIONS are omitted, then the CHOICE is multiple, meaning that any number of SELECTIONS may be made, including zero. If the MINSELECTIONS and MAXSELECTIONS attributes are both set to ‘1’, then the CHOICE is single, meaning that exactly one SELECTION is required to be chosen.

A CHOICE may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information about the *choice* by means of an attribute.

NOTE *descriptor* information can be associated with the *choice* by such attributes, as well as by child DESCRIPTOR elements.

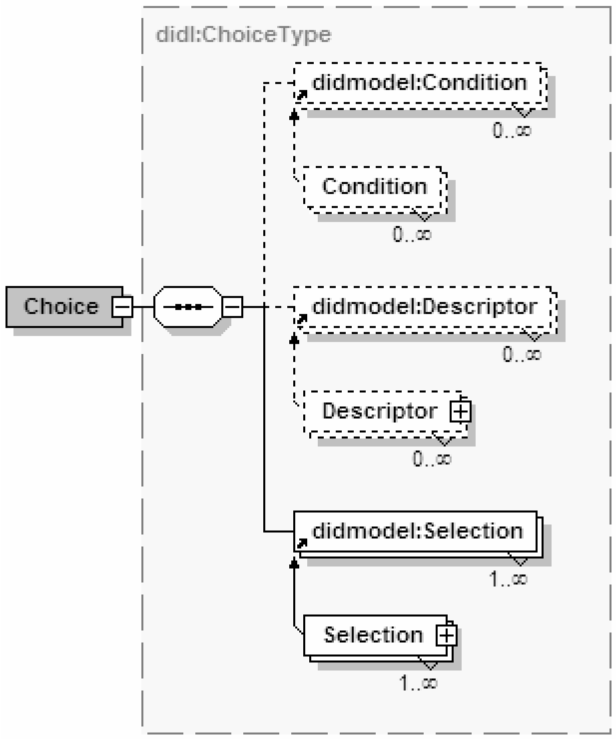
Validation Rules:

— The value of the MAXSELECTIONS attribute shall be no less than the value of the MINSELECTIONS attribute.

— The value of the MINSELECTIONS attribute shall be no larger than the number of SELECTION children.

— The values specified in the DEFAULT attribute shall each match the SELECT_ID value of one of the SELECTIONS within this CHOICE. If the DEFAULT attribute is specified, the number of individual values in the DEFAULT attribute shall not be less than the value of the MINSELECTIONS attribute, nor more than the value of the MAXSELECTIONS attribute.

Table 14 — CHOICE element syntax

Diagram	

Children	<Condition> <Descriptor> <Selection>		
Used by	<Item>		
Attributes	**Name**	**Type**	**Description**
	minSelections	nonNegativeInteger	Minimum number of *selections* that are required to be made. If not present, there is no minimum number.
	maxSelections	positiveInteger	Maximum number of *selections* that are allowed to be made. If not present, there is no maximum number.
	default	IDREFS	A list of ID values defined in SELECT_ID attributes of SELECTION elements, indicating the set of default *selections* for this *choice*.
	choice_id	ID	Serves as the target for an ASSERTION element.
Source			

```
<element name="Choice" type="didl:ChoiceType"
 substitutionGroup="didmodel:Choice"/>
<complexType name="ChoiceType">
 <complexContent>
  <extension base="didmodel:ChoiceType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Selection" maxOccurs="unbounded"/>
   </sequence>
   <attribute name="minSelections" type="nonNegativeInteger"/>
   <attribute name="maxSelections" type="positiveInteger"/>
   <attribute name="default" type="IDREFS"/>
   <attribute name="choice_id" type="ID"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
```

7.2.17 <Selection>

A SELECTION element represents a *selection*. As such, it defines a specific decision about a particular CHOICE. The SELECT_ID attribute value identifies the *predicate* embodied by the SELECTION, and relates it to one or more CONDITIONS somewhere within an ITEM. At configuration time[1)], if the SELECTION is chosen, its *predicate* becomes true; if it is rejected, its *predicate* becomes false; if it is left unresolved, its *predicate* is left undecided.

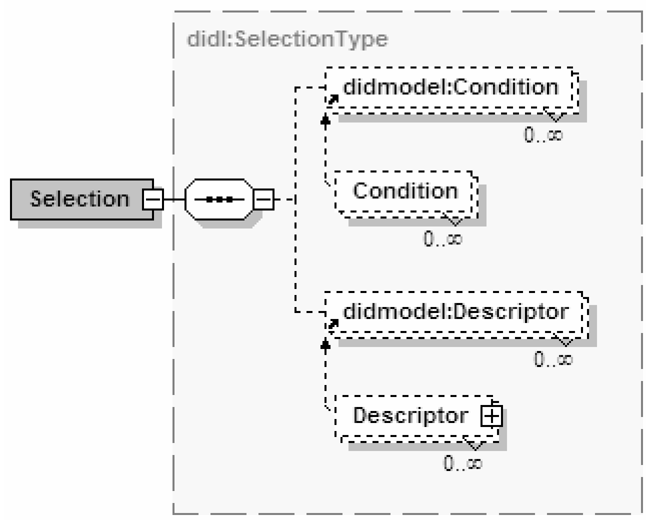
A SELECTION may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information about the *selection* by means of an attribute.

NOTE 1 *descriptor* information can be associated with the *selection* by such attributes, as well as by child DESCRIPTOR elements.

1) The time when some software agent decides it is the appropriate time to make decisions on the SELECTIONS defined within a CHOICE

NOTE 2 SELECTIONS and entire CHOICES can be made conditional (i.e. can have one or more CONDITION child elements). This makes it possible to implement complex decision trees in which certain selections make certain subsequent CHOICES or SELECTIONS redundant.

Table 15 — SELECTION element syntax

<table>
<tr><td>Diagram</td><td>
</td></tr>
<tr><td>Children</td><td><Condition> <Descriptor></td></tr>
<tr><td>Used by</td><td><Choice></td></tr>
<tr><td>Attributes</td><td><table><tr><td>Name</td><td>Type</td><td>Description</td></tr><tr><td>select_id</td><td>ID</td><td>An ID value that can be referenced in a CONDITION element.</td></tr></table></td></tr>
<tr><td>Source</td><td><pre>
<element name="Selection" type="didl:SelectionType"
 substitutionGroup="didmodel:Selection"/>
<complexType name="SelectionType">
 <complexContent>
  <extension base="didmodel:SelectionType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attribute name="select_id" type="ID" use="required"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
</pre></td></tr>
</table>

EXAMPLE This example, shows a DIDL document containing a CHOICE on whether to include a supplemental video clip, followed by a CHOICE on the encoding preference of the video clip. If, during configuration time, the video clip SELECTION is rejected, then the encoding preference CHOICE should be skipped.


```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item>
  <Choice minSelections="1" maxSelections="1">
   <Descriptor>
    <Statement mimeType="text/plain">
     Would you like to include the supplemental video?
    </Statement>
   </Descriptor>
   <Selection select_id="SHOW_VIDEO">
    <Descriptor>
     <Statement mimeType="text/plain">Yes please!</Statement>
    </Descriptor>
   </Selection>
   <Selection select_id="NO_VIDEO">
    <Descriptor>
     <Statement mimeType="text/plain">No thanks.</Statement>
    </Descriptor>
   </Selection>
  </Choice>
  <Choice minSelections="1" maxSelections="1">
   <Descriptor>
    <Statement mimeType="text/plain">
     What video format do you prefer?
    </Statement>
   </Descriptor>
   <Selection select_id="VIDEO_MPEG">
    <Descriptor>
     <Statement mimeType="text/plain">I want MPEG-1/2</Statement>
    </Descriptor>
   </Selection>
   <Selection select_id="VIDEO_QUICKTIME">
    <Descriptor>
     <Statement mimeType="text/plain">I want Quicktime</Statement>
    </Descriptor>
   </Selection>
  </Choice>
  <Component>
   <Resource ref="audio.mp3" mimeType="audio/mpeg"/>
  </Component>
  <Component>
   <Condition require="SHOW_VIDEO VIDEO_MPEG"/>
   <Resource ref="video.mpg" mimeType="video/mpeg"/>
  </Component>
  <Component>
   <Condition require="SHOW_VIDEO VIDEO_QUICKTIME"/>
   <Resource ref="video.mov" mimeType="video/quicktime"/>
  </Component>
 </Item>
</DIDL>
```

7.2.18 <Condition>

A CONDITION element represents a *condition*. As such, it denotes the parent element as being conditional on a set of predicate tests. The REQUIRE attribute lists the set of *predicate*s that are required to become true, and the EXCEPT attribute lists the set of *predicate*s that are required to become false, in order for the CONDITION to be satisfied. Each *predicate* is identified by the value of the SELECT_ID attribute in a SELECTION element.

A set of CONDITION elements defines a Boolean combination of predicate tests. Multiple tests within a CONDITION are combined as a conjunction (an AND relationship). Multiple CONDITION elements within a given parent are combined as a disjunction (an OR relationship).

Validation Rules:

— Each ID value specified in the REQUIRE and EXCEPT attributes shall match a SELECT_ID attribute value defined in a SELECTION element in a CHOICE that is a child of an ITEM element that is an ancestor of the CONDITION.

— Empty CONDITIONS are not permitted. Therefore, it is not valid for a CONDITION element to have neither a REQUIRE attribute nor an EXCEPT attribute.

Table 16 — CONDITION element syntax

<table>
<tr><td>Diagram</td><td colspan="3">Condition</td></tr>
<tr><td>Used by</td><td colspan="3"><Anchor> <Choice> <Component> <Descriptor> <Item> <Selection></td></tr>
<tr><td rowspan="3">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>require</td><td>IDREFS</td><td>A list of ID values matching SELECT_ID attribute of SELECTION element(s), indicating the SELECTION(s) that are required to be asserted for this CONDITION to evaluate to true.</td></tr>
<tr><td>except</td><td>IDREFS</td><td>A list of ID values matching SELECT_ID attribute of SELECTION element(s), indicating the SELECTION(s) that are required to be asserted for this CONDITION to evaluate to false.</td></tr>
<tr><td>Source</td><td colspan="3"><element name="Condition" type="didl:ConditionType"
 substitutionGroup="didmodel:Condition"/>
<complexType name="ConditionType">
 <complexContent>
  <extension base="didmodel:ConditionType">
   <attribute name="require" type="IDREFS"/>
   <attribute name="except" type="IDREFS"/>
  </extension>
 </complexContent>
</complexType></td></tr>
</table>

EXAMPLE This example shows a simple CHOICE. The CHOICE specifies two SELECTIONS and specifies that exactly one SELECTION is required to be chosen. The DESCRIPTOR for the CHOICE describes the *choice* in a human-readable format, and the DESCRIPTORS for the SELECTIONS do likewise. If the user chooses the MP3 selection, then the MP3_FORMAT *predicate* becomes true and the WMA_FORMAT *predicate* becomes false, so the first COMPONENT is retained and the second one is discarded. Likewise, if the user chooses the WMA selection, then the WMA_FORMAT *predicate* becomes true and the MP3_FORMAT *predicate* becomes false, so the first is discarded and the second one is retained.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item>
  <Choice minSelections="1" maxSelections="1">
   <Descriptor>
    <Statement mimeType="text/plain">
     What format would you prefer?
    </Statement>
   </Descriptor>
   <Selection select_id="MP3_FORMAT">
    <Descriptor>
     <Statement mimeType="text/plain">I want MP3</Statement>
    </Descriptor>
   </Selection>
   <Selection select_id="WMA_FORMAT">
    <Descriptor>
     <Statement mimeType="text/plain">I want WMA</Statement>
    </Descriptor>
   </Selection>
  </Choice>
  <Component>
   <Condition require="MP3_FORMAT"/>
   <Resource ref="clip.mp3" mimeType="audio/mpeg"/>
  </Component>
  <Component>
   <Condition require="WMA_FORMAT"/>
   <Resource ref="clip.wma" mimeType="audio/x-ms-wma"/>
  </Component>
 </Item>
</DIDL>
```

7.2.19 <Annotation>

An ANNOTATION element represents an *annotation*. As such, it allows additional DESCRIPTORS and/or ANCHORS to be logically added to an element in one or more DIDL elements without affecting the original contents of the element. This allows, for example, an end-user to associate bookmarks and commentary to a created Asset, without altering or adding to that Asset. It also allows ASSERTIONS to be made on the *predicates* associated with the SELECTIONS of a given CHOICE.

An ANNOTATION may target an element located somewhere within the parent document ("internal annotation"), or an element in an external DIDL document ("external annotation").

Within ANNOTATIONS, an XInclude `include` performs the same functionality as it does in other contexts of DIDL (such as ITEMS or DESCRIPTORS): the included items targeted by an `include` within an ANNOTATION are included (at runtime) in the ANNOTATION itself <u>before</u> it is logically attached to the ANNOTATION TARGET.

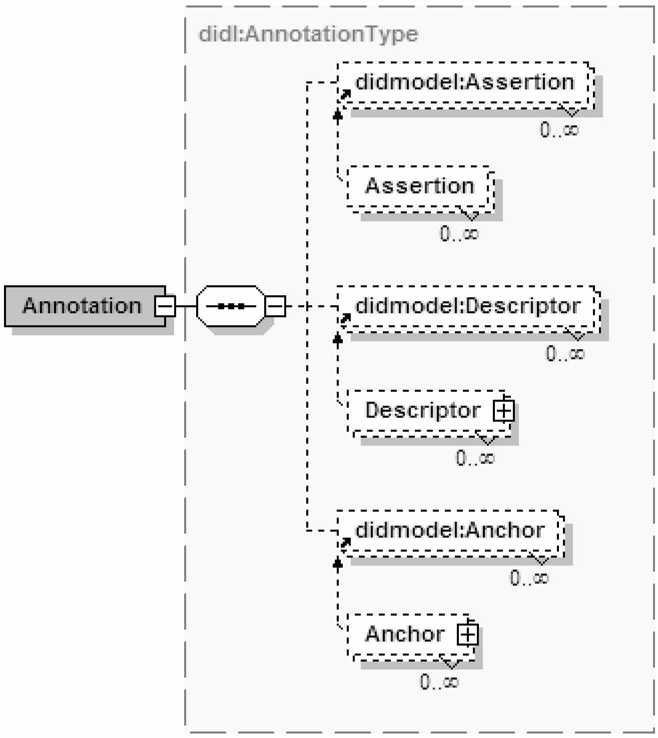
An ANNOTATION may have any attribute from other namespaces. Such attributes provide an additional representation of a *descriptor* and as such contain descriptive information logically added to an element in one or more DIDL elements without affecting the original contents of the element by means of an attribute of the ANNOTATION.

NOTE *descriptor* information can be logically added by such attributes, as well as by child DESCRIPTOR elements of the ANNOTATION.

Validation Rules:

- For internal annotation, the values given in the TARGET attribute shall each correspond to some descendant element of the parent element, or that of the parent element itself.
- The contents of an ANNOTATION shall conform to the content model of the targeted element(s). For example, ANCHORS may be included only if the TARGET attribute values each match the ID value of a COMPONENT.
- If an ANNOTATION contains an ASSERTION, then its TARGET attribute values shall each match the ID attribute value of an ITEM.

Table 17 — ANNOTATION element syntax

<table>
<tr><td>Diagram</td><td colspan="3">
</td></tr>
<tr><td>Children</td><td colspan="3"><Assertion> <Descriptor> <Anchor></td></tr>
<tr><td>Used by</td><td colspan="3"><Item> <Declarations></td></tr>
<tr><td rowspan="3">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>target</td><td>anyURI</td><td>Identifies the elements being annotated.</td></tr>
<tr><td>id</td><td>ID</td><td>A unique ID value.</td></tr>
</table>

Source	<pre><element name="Annotation" type="didl:AnnotationType"
	 substitutionGroup="didmodel:Annotation"/>
	<complexType name="AnnotationType">
	 <complexContent>
	  <extension base="didmodel:AnnotationType">
	   <sequence>
	    <element ref="didmodel:Assertion" minOccurs="0"
	     maxOccurs="unbounded"/>
	    <element ref="didmodel:Descriptor" minOccurs="0"
	     maxOccurs="unbounded"/>
	    <element ref="didmodel:Anchor" minOccurs="0"
	     maxOccurs="unbounded"/>
	   </sequence>
	   <attribute name="target" type="anyURI" use="required"/>
	   <attributeGroup ref="didl:ID_ATTRS"/>
	   <anyAttribute namespace="##other" processContents="lax"/>
	  </extension>
	 </complexContent>
	</complexType></pre>

EXAMPLE This example shows how an ANNOTATION element can be used to logically add elements to an Item without actually modifying its contents. In this case, the element being annotated is the first child ITEM of the outermost ITEM. At rendering time, the application software would logically add the DESCRIPTOR to the display of the ITEM.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item>
  <Item id="PHOTO_1">
   <Component>
    <Resource ref="myFirstPicture.jpg" mimeType="image/jpeg" />
   </Component>
  </Item>
  <Item>
   <Component>
    <Resource ref="mySecondPic.bmp" mimeType="image/x-ms-bmp" />
   </Component>
  </Item>
  <Annotation target="#PHOTO_1">
   <Descriptor>
    <Statement mimeType="text/plain">This photo is really cool!</Statement>
   </Descriptor>
  </Annotation>
 </Item>
</DIDL>
```

7.2.20 <Assertion>

An ASSERTION element represents an *assertion*. As such, it allows a set of *predicates* in a particular CHOICE to be asserted as true or false. This captures a particular state of a set of SELECTIONS within a CHOICE to be instantiated without modifying the original document. ASSERTIONS are always part of an ANNOTATION to an ITEM. The TARGET attribute of the ASSERTION identifies the affected CHOICE.

ASSERTIONS may either fully or partially resolve the given CHOICE. The *predicates* represented by the missing SELECT_ID values are left to the same value as before the resolving of the ASSERTION. It is possible to continue resolving a partially resolved CHOICE by simply assigning true or false values to some or all of the undecided *predicates* to arrive at a new ASSERTION. ASSERTIONS shall be processed in document order starting at the root DIDL element.

Validation Rules:

— The associated CHOICE element shall be a descendant of the ITEM whose ID attribute value matches the parent ANNOTATION'S TARGET attribute value.

— The number of true *predicates* (i.e. the number of SELECT_ID values listed in the TRUE attribute) shall be less than or equal to the MAXSELECTIONS attribute value in the associated CHOICE.

— The total number of SELECT_ID values defined in the associated CHOICE, minus the number of SELECT_ID values listed in the FALSE attribute, shall be greater than or equal to the MINSELECTIONS attribute value in the associated CHOICE.

Table 18 — ASSERTION element syntax

<table>
<tr><td>Diagram</td><td colspan="3">
</td></tr>
<tr><td>Used by</td><td colspan="3"><Annotation></td></tr>
<tr><td rowspan="4">Attributes</td><td>Name</td><td>Type</td><td>Description</td></tr>
<tr><td>target</td><td>anyURI</td><td>Identifies the CHOICE that this ASSERTION affects. Shall contain an ID value that matches a CHOICE_ID attribute within the descendants of the parent of the ANNOTATION that contains this ASSERTION.</td></tr>
<tr><td>true</td><td>NMTOKENS</td><td>The set of ID values corresponding to the SELECT_ID attributes of SELECTIONS that are to be asserted as true.</td></tr>
<tr><td>false</td><td>NMTOKENS</td><td>The set of ID values corresponding to the SELECT_ID attributes of SELECTIONS that are to be asserted as false.</td></tr>
<tr><td>Source</td><td colspan="3"><pre>
<element name="Assertion" type="didl:AssertionType"
 substitutionGroup="didmodel:Assertion"/>
<complexType name="AssertionType">
 <complexContent>
  <extension base="didmodel:AssertionType">
   <attribute name="target" type="anyURI" use="required"/>
   <attribute name="true" type="NMTOKENS" use="optional"/>
   <attribute name="false" type="NMTOKENS" use="optional"/>
  </extension>
 </complexContent>
</complexType>
</pre></td></tr>
</table>

EXAMPLE This example shows an ASSERTION can be used to "save" a configuration within a document. In this case, the ASSERTION is targeting the format CHOICE, and is asserting the MP3_FORMAT SELECTION. Since the CHOICE specifies a MAXSELECTIONS value of 1, the application software can completely resolve the CHOICE with the given ASSERTION.

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item id="THE_Item">
  <Choice choice_id="FORMAT_Choice" minSelections="1" maxSelections="1">
   <Descriptor>
    <Statement mimeType="text/plain">What format would you prefer?</Statement>
   </Descriptor>
   <Selection select_id="MP3_FORMAT">
    <Descriptor>
     <Statement mimeType="text/plain">I want MP3</Statement>
    </Descriptor>
   </Selection>
   <Selection select_id="WMA_FORMAT">
    <Descriptor>
     <Statement mimeType="text/plain">I want WMA</Statement>
    </Descriptor>
   </Selection>
  </Choice>
  . . .
  <Annotation target="#THE_Item">
   <Assertion target="#FORMAT_Choice" true="MP3_FORMAT"/>
  </Annotation>
 </Item>
</DIDL>
```

8 The Digital Item Declaration XML Schema Definitions (informative)

8.1 Purpose and Overview

The purpose of this clause is to provide the informative XML schemas specified according to W3C XMLSCHEMA comprising complete grammars for the representation of the Digital Item Declaration in XML.

In subclause 8.2 a DID Model abstract schema is given. The purpose of the DID Model abstract schema is to define abstract XML schema datatypes corresponding to entities of the DID Model that are represented in XML as XML elements.

In subclause 8.3 a DIDL schema is given. The DIDL schema defines concrete DIDL datatypes based on the DID model abstract datatypes, in addition to some special DIDL datatypes that do not correspond to DID Model entities. These concrete DIDL datatypes also define the structural relationship of the DIDL elements (via the content model of the DIDL elements) based on the defined relationships of corresponding DID Model entities, thus enabling the XML representation of the Digital Item Declaration. This DIDL schema can be used to schema validate a DIDL document that contains only DIDL elements as defined in clause 7.

The DID Model abstract schema also allows for representations of the DID defined by other parts of MPEG-21 that can be based on the same DID Model abstract datatypes.

NOTE The content model of the DIDL schema elements is defined in terms of corresponding elements of the DID Model abstract schema. This allows elements from representations of DID entities defined by other parts of MPEG-21 based on the same DID Model abstract datatypes to be substituted in a DID represented using DIDL. For example, part 4 of MPEG-21 defines protected representations of DID entities that can be substituted for corresponding DIDL elements within a DIDL representation of the DID.

8.2 DID Model Abstract Schema

```
<?xml version="1.0"?>
<!--=========================================
####################################################################
# ISO/IEC 21000-2:2005                         #
#  Information technology                        #
#  - Multimedia framework (MPEG-21)                   #
#   - Part 2: Digital Item Declaration                 #
#                                  #
# Schema for DID Model abstract Types                  #
#                                  #
####################################################################
 =========================================-->
<schema targetNamespace="urn:mpeg:mpeg21:2002:02-DIDMODEL-NS"
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:didmodel="urn:mpeg:mpeg21:2002:02-DIDMODEL-NS"
 elementFormDefault="qualified" attributeFormDefault="unqualified"
 version="0.01">
  <complexType name="DIDBaseType" abstract="true"/>
  <element name="Container" type="didmodel:ContainerType" abstract="true"/>
  <complexType name="ContainerType" abstract="true">
     <complexContent>
        <extension base="didmodel:DIDBaseType"/>
     </complexContent>
  </complexType>
  <element name="Item" type="didmodel:ItemType" abstract="true"/>
  <complexType name="ItemType" abstract="true">
     <complexContent>
        <extension base="didmodel:DIDBaseType"/>
     </complexContent>
  </complexType>
  <element name="Descriptor" type="didmodel:DescriptorType" abstract="true"/>
  <complexType name="DescriptorType" abstract="true">
     <complexContent>
        <extension base="didmodel:DIDBaseType"/>
     </complexContent>
  </complexType>
  <element name="Statement" type="didmodel:StatementType" abstract="true"/>
  <complexType name="StatementType" abstract="true">
     <complexContent>
        <extension base="didmodel:DIDBaseType"/>
     </complexContent>
  </complexType>
  <element name="Component" type="didmodel:ComponentType" abstract="true"/>
  <complexType name="ComponentType" abstract="true">
     <complexContent>
        <extension base="didmodel:DIDBaseType"/>
     </complexContent>
  </complexType>
  <element name="Anchor" type="didmodel:AnchorType" abstract="true"/>
  <complexType name="AnchorType" abstract="true">
     <complexContent>
        <extension base="didmodel:DIDBaseType"/>
     </complexContent>
  </complexType>
  <element name="Fragment" type="didmodel:FragmentType" abstract="true"/>
  <complexType name="FragmentType" abstract="true">
     <complexContent>
```

```
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
      <element name="Condition" type="didmodel:ConditionType" abstract="true"/>
      <complexType name="ConditionType" abstract="true">
         <complexContent>
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
      <element name="Choice" type="didmodel:ChoiceType" abstract="true"/>
      <complexType name="ChoiceType" abstract="true">
         <complexContent>
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
      <element name="Selection" type="didmodel:SelectionType" abstract="true"/>
      <complexType name="SelectionType" abstract="true">
         <complexContent>
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
      <element name="Resource" type="didmodel:ResourceType" abstract="true"/>
      <complexType name="ResourceType" abstract="true">
         <complexContent>
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
      <element name="Annotation" type="didmodel:AnnotationType" abstract="true"/>
      <complexType name="AnnotationType" abstract="true">
         <complexContent>
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
      <element name="Assertion" type="didmodel:AssertionType" abstract="true"/>
      <complexType name="AssertionType" abstract="true">
         <complexContent>
            <extension base="didmodel:DIDBaseType"/>
         </complexContent>
      </complexType>
   </schema>
```

8.3 DIDL Schema

```
<?xml version="1.0"?>
<!--=========================================
##################################################################
# ISO/IEC 21000-2:2005                      #
#  Information technology                   #
#  - Multimedia framework (MPEG-21)              #
#   - Part 2: Digital Item Declaration              #
#                                 #
# Schema for Derived DIDL Types                  #
#                                 #
##################################################################
 =========================================-->
<schema targetNamespace="urn:mpeg:mpeg21:2002:02-DIDL-NS"
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:didl="urn:mpeg:mpeg21:2002:02-DIDL-NS"
 xmlns:didmodel="urn:mpeg:mpeg21:2002:02-DIDMODEL-NS"
 elementFormDefault="qualified" attributeFormDefault="unqualified"
 version="0.01">
 <!--============================================================

    Import abstract types representing DID model entities:

   ============================================================-->
 <import namespace="urn:mpeg:mpeg21:2002:02-DIDMODEL-NS"
  schemaLocation="didmodel.xsd"/>
 <!--============================================================-->
 <attributeGroup name="ID_ATTRS">
  <attribute name="id" type="ID" use="optional"/>
 </attributeGroup>
 <!--============================================================

    Container element may contain any number of Container elements
    followed by any number of Items.

   ============================================================-->
 <element name="Container" type="didl:ContainerType"
substitutionGroup="didmodel:Container"/>
 <complexType name="ContainerType">
  <complexContent>
   <extension base="didmodel:ContainerType">
    <sequence>
     <element ref="didmodel:Descriptor" minOccurs="0"
      maxOccurs="unbounded"/>
     <element ref="didmodel:Container" minOccurs="0"
      maxOccurs="unbounded"/>
     <element ref="didmodel:Item" minOccurs="0"
      maxOccurs="unbounded"/>
    </sequence>
    <attributeGroup ref="didl:ID_ATTRS"/>
    <anyAttribute namespace="##other" processContents="lax"/>
   </extension>
  </complexContent>
 </complexType>
 <!--============================================================

    Item element contains any number Choice elements,
    followed by at least one Item or Component element.
    An Item may be conditional.

   ============================================================-->
```



```
<element name="Item" type="didl:ItemType"
 substitutionGroup="didmodel:Item"/>
<complexType name="ItemType">
 <complexContent>
  <extension base="didmodel:ItemType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Choice" minOccurs="0"
     maxOccurs="unbounded"/>
    <choice minOccurs="0" maxOccurs="unbounded">
     <element ref="didmodel:Item"/>
     <element ref="didmodel:Component"/>
    </choice>
    <element ref="didmodel:Annotation" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--================================================================

   A Descriptor contains descriptive data about its parent
   element.

   The Descriptor may be resource-based (comprised of a single
   Component), or text-based (comprised of a single Statement).
   A Descriptor may be conditional.

  ================================================================-->
<element name="Descriptor" type="didl:DescriptorType"
 substitutionGroup="didmodel:Descriptor"/>
<complexType name="DescriptorType">
 <complexContent>
  <extension base="didmodel:DescriptorType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <choice>
     <element ref="didmodel:Component"/>
     <element ref="didmodel:Statement"/>
    </choice>
   </sequence>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--================================================================

   A Statement contains textual descriptive data within a
   Descriptor.

   Attribs:
```

```
   MimeType - A string identifying the type of metadata

  ============================================================-->
<element name="Statement" type="didl:StatementType"
 substitutionGroup="didmodel:Statement"/>
<complexType name="StatementType" mixed="true">
 <complexContent mixed="true">
  <extension base="didmodel:StatementType">
   <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0"/>
   </sequence>
   <attribute name="mimeType" type="string" use="required"/>
   <attribute name="ref" type="anyURI"/>
   <attribute name="encoding" type="string"/>
   <attribute name="contentEncoding" type="NMTOKENS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--============================================================

   Component element contains one or more Resource elements, followed by any
   number of Anchor elements.
   A Component may be conditional.

  ============================================================-->
<element name="Component" type="didl:ComponentType"
 substitutionGroup="didmodel:Component"/>
<complexType name="ComponentType">
 <complexContent>
  <extension base="didmodel:ComponentType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Resource" maxOccurs="unbounded"/>
    <element ref="didmodel:Anchor" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--============================================================

   An Anchor element indicates a point of interest in the
   resource of the parent Component.
   An Anchor may be conditional.

   Attribs:
   precedence - An unsigned integer value indicating the position
       that this start point should occupy relative to the
       other start points in this title. The highest
       precedence anchor is the default entry
       point.

  ============================================================-->
<element name="Anchor" type="didl:AnchorType"
```

```
  substitutionGroup="didmodel:Anchor"/>
 <complexType name="AnchorType">
  <complexContent>
   <extension base="didmodel:AnchorType">
    <sequence>
     <element ref="didmodel:Condition" minOccurs="0"
      maxOccurs="unbounded"/>
     <element ref="didmodel:Descriptor" minOccurs="0"
      maxOccurs="unbounded"/>
     <element ref="didmodel:Fragment" minOccurs="0"/>
    </sequence>
    <attribute name="precedence" type="unsignedInt"/>
    <attributeGroup ref="didl:ID_ATTRS"/>
    <anyAttribute namespace="##other" processContents="lax"/>
   </extension>
  </complexContent>
 </complexType>
 <!--=============================================================

   A Fragment element indicates a part of interest in a
   resource.

   Attribs:
   fragmentId - A fragment identifier that may locate the part
       of interest within the resource. This string,
       when appended to the URI of the resource, plus
       a '#', becomes the full URI for the part of
  .    interest.

  ==============================================================-->
 <element name="Fragment" type="didl:FragmentType"
  substitutionGroup="didmodel:Fragment"/>
 <complexType name="FragmentType">
  <complexContent>
   <extension base="didmodel:FragmentType">
    <sequence>
     <any namespace="##any" processContents="lax" minOccurs="0"/>
    </sequence>
    <attribute name="fragmentId" type="string"/>
   </extension>
  </complexContent>
 </complexType>
 <!--=============================================================

   Condition element contains no children. It indicates a
   selection condition for the parent file. Multiple Condition
   tags indicate an 'OR' relationship, in that only one Condition
   needs to be satisfied for the parent element to be retrieved,
   included, etc. The individual IDs in the require attribute of
   a Condition tag have an 'AND' relationship in that selection
   of all of the IDs referenced are required to satisfy that
   Condition.
```

```
   Attribs:
   require  - Indicates the Selection(s) required to be affirmed
      for this Condition to be "satisfied".

   except - Indicates the Selection(s) required to be denied
      for this Condition to be "satisfied".

  ================================================================-->
<element name="Condition" type="didl:ConditionType"
 substitutionGroup="didmodel:Condition"/>
<complexType name="ConditionType">
 <complexContent>
  <extension base="didmodel:ConditionType">
   <attribute name="require" type="IDREFS"/>
   <attribute name="except" type="IDREFS"/>
  </extension>
 </complexContent>
</complexType>
<!--================================================================

   Choice element contains one or more Selections.
   A Choice may be conditional.

   Attribs:

   MinSelections  - Minimum number of Selections that are required be made.
        If not present, there is no minimum.

   MaxSelections  - Maximum number of Selections that are allowed to be made.
        If not present, there is no maximum.

   default  - Indicates one or more default selections to use in the absence
      of info to make a more specific decision. Shall conform to the
      requirements of the minSelections and/or maxSelections
      attributes if present.

   choice_id  - Serves as a target for Assertion elements.

  ================================================================-->
<element name="Choice" type="didl:ChoiceType"
 substitutionGroup="didmodel:Choice"/>
<complexType name="ChoiceType">
 <complexContent>
  <extension base="didmodel:ChoiceType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Selection" maxOccurs="unbounded"/>
   </sequence>
   <attribute name="minSelections" type="nonNegativeInteger"/>
   <attribute name="maxSelections" type="positiveInteger"/>
   <attribute name="default" type="IDREFS"/>
   <attribute name="choice_id" type="ID"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--================================================================
```



```
   Selection element contains no children.
   A Selection may be conditional.

   Attrib: select_id - Uniquely identifies the Selection

  ===============================================================-->
<element name="Selection" type="didl:SelectionType"
 substitutionGroup="didmodel:Selection"/>
<complexType name="SelectionType">
 <complexContent>
  <extension base="didmodel:SelectionType">
   <sequence>
    <element ref="didmodel:Condition" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attribute name="select_id" type="ID" use="required"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--===============================================================

   Resource element contains or points to binary data. The
   contained data may be binary or any valid XML element.

   Attribs:
   mimeType - An identifier of a recognized scheme indicating the type
      of the resource.
   ref - A URI from which the resource data can be obtained

  ===============================================================-->
<element name="Resource" type="didl:ResourceType"
 substitutionGroup="didmodel:Resource"/>
<complexType name="ResourceType" mixed="true">
 <complexContent mixed="true">
  <extension base="didmodel:ResourceType">
   <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0"/>
   </sequence>
   <attribute name="mimeType" type="string" use="required"/>
   <attribute name="ref" type="anyURI"/>
   <attribute name="encoding" type="string"/>
   <attribute name="contentEncoding" type="NMTOKENS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--===============================================================

   Annotation contains any number of Assertions followed by
   any number of Descriptors followed by any number of Anchors

   Attrib: target - Points to the element being annotated
```

```
   Restrictions:
     1. The target shall reference an element within the
       parent Item, or may reference the parent Item itself.
     2. The contents of an Annotation shall conform with the
       content model of the targeted element. For example,
       Anchors may be included only if the target references
       a Component.

  =============================================================-->
<element name="Annotation" type="didl:AnnotationType"
 substitutionGroup="didmodel:Annotation"/>
<complexType name="AnnotationType">
 <complexContent>
  <extension base="didmodel:AnnotationType">
   <sequence>
    <element ref="didmodel:Assertion" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Descriptor" minOccurs="0"
     maxOccurs="unbounded"/>
    <element ref="didmodel:Anchor" minOccurs="0"
     maxOccurs="unbounded"/>
   </sequence>
   <attribute name="target" type="anyURI" use="required"/>
   <attributeGroup ref="didl:ID_ATTRS"/>
   <anyAttribute namespace="##other" processContents="lax"/>
  </extension>
 </complexContent>
</complexType>
<!--===============================================================

   Assertion contains no child elements

   Attribs:

     true  - The set of selection IDs which are asserted as true
     false - The set of selection IDs which are asserted as false

  =============================================================-->
<element name="Assertion" type="didl:AssertionType"
 substitutionGroup="didmodel:Assertion"/>
<complexType name="AssertionType">
 <complexContent>
  <extension base="didmodel:AssertionType">
   <attribute name="target" type="anyURI" use="required"/>
   <attribute name="true" type="NMTOKENS" use="optional"/>
   <attribute name="false" type="NMTOKENS" use="optional"/>
  </extension>
 </complexContent>
</complexType>
<!-- elements from here onward are unique to the DIDL representation-->
<!--===============================================================

   DIDL element may contain exactly one Container or Item.

  =============================================================-->
```



```
 <element name="DIDL" type="didl:DIDLType"/>
 <complexType name="DIDLType">
  <sequence>
   <element ref="didl:DIDLInfo" minOccurs="0" maxOccurs="unbounded"/>
   <element ref="didl:Declarations" minOccurs="0"/>
   <choice>
    <element ref="didmodel:Container"/>
    <element ref="didmodel:Item"/>
   </choice>
  </sequence>
  <attribute name="DIDLDocumentId" type="anyURI"/>
  <anyAttribute namespace="##other" processContents="lax"/>
 </complexType>
 <!--==============================================================

  A DIDLInfo element may contain information only applicable to
  the DIDL document and not the Digital Item represented.

 ==============================================================-->
 <element name="DIDLInfo" type="didl:DIDLInfoType"/>
 <complexType name="DIDLInfoType">
  <sequence>
   <any namespace="##any" processContents="lax"/>
  </sequence>
 </complexType>
 <!--==============================================================

   A Declarations element contains any number of Items, Descriptors,
   Components, and Anchors, in any order.

  ==============================================================-->
 <element name="Declarations" type="didl:DeclarationsType"/>
 <complexType name="DeclarationsType">
  <choice maxOccurs="unbounded">
   <element ref="didmodel:Item"/>
   <element ref="didmodel:Descriptor"/>
   <element ref="didmodel:Component"/>
   <element ref="didmodel:Annotation"/>
   <element ref="didmodel:Anchor"/>
  </choice>
 </complexType>
</schema>
```


9 Example Digital Items expressed in DIDL

9.1 Example 1: Using MPEG-7 descriptors in conjunction with a Choice

EXAMPLE This example shows how MPEG-7 descriptors can be used to express the information required for a user agent to resolve a CHOICE.

```
<?xml version="1.0" encoding="UTF-8"?>
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS"
 xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xi="http://www.w3.org/2001/XInclude">
   <Declarations>
       <!-- Define a Descriptor that will be reused in multiple places in this Item.
            In this case, it is the invariant parts of the media profile of the content.
-->
       <Descriptor id="COMMON_PROFILE_DESC">
           <Statement mimeType="text/xml">
               <mpeg7:Mpeg7>
                   <mpeg7:DescriptionUnit xsi:type="mpeg7:MediaProfileType">
                       <mpeg7:MediaFormat>
                           <mpeg7:Content>audiovisual</mpeg7:Content>
                           <mpeg7:Medium href="urn:mpeg:mpeg7:cs:MPEG7MediumCS:1.1"/>
                           <mpeg7:FileFormat
href="urn:mpeg:mpeg7:cs:MPEG7FileFormatCS:3"/>
                           <mpeg7:AudioCodingFormat
                            href="urn:mpeg:mpeg7:cs:MPEG7AudioCodingFormatCS:4.3.1"/>
                       </mpeg7:MediaFormat>
                   </mpeg7:DescriptionUnit>
               </mpeg7:Mpeg7>
           </Statement>
       </Descriptor>
   </Declarations>
   <Item>
       <Choice minSelections="1" maxSelections="1">
           <!-- Each Selection contains the parts of the MediaFormat that vary between
the
               three alternative resources for the content. In this example, the client
               machine will have the opportunity to balance the allocation of bandwidth
on
               its network connection with the quality level desired by the end user. --
>
           <Selection select_id="HI_QUALITY">
               <Descriptor id="HI_QUALITY_DESC">
                   <Statement mimeType="text/xml">
                       <mpeg7:Mpeg7>
                           <mpeg7:DescriptionUnit xsi:type="mpeg7:MediaProfileType">
                               <mpeg7:MediaFormat>
                                   <mpeg7:VisualCodingFormat

href="urn:mpeg:mpeg7:cs:MPEG7VisualCodingFormatCS:3.1.4"/>
                                   <mpeg7:BitRate>384000</mpeg7:BitRate>
                               </mpeg7:MediaFormat>
                               <mpeg7:MediaQuality>
                                   <mpeg7:QualityRating ratingType="objective">
                                       <mpeg7:RatingValue>89.4</mpeg7:RatingValue>
                                       <mpeg7:RatingMetric>
                                           <mpeg7:QualityRatingScheme
                                   href="urn:mpeg:mpeg7:cs:MPEG-
7QualityRatingSchemeCS:2.3"/>

   <mpeg7:RatingStyle>higherBetter</mpeg7:RatingStyle>
                                       </mpeg7:RatingMetric>
                                   </mpeg7:QualityRating>
                               </mpeg7:MediaQuality>
                           </mpeg7:DescriptionUnit>
```

```
                    </mpeg7:Mpeg7>
                </Statement>
            </Descriptor>
        </Selection>
        <Selection select_id="MED_QUALITY">
            <Descriptor id="MED_QUALITY_DESC">
                <Statement mimeType="text/xml">
                    <mpeg7:Mpeg7>
                        <mpeg7:DescriptionUnit xsi:type="mpeg7:MediaProfileType">
                            <mpeg7:MediaFormat>
                                <mpeg7:VisualCodingFormat

href="urn:mpeg:mpeg7:cs:MPEG7VisualCodingFormatCS:3.1.3"/>
                                <mpeg7:BitRate>128000</mpeg7:BitRate>
                            </mpeg7:MediaFormat>
                            <mpeg7:MediaQuality>
                                <mpeg7:QualityRating ratingType="objective">
                                    <mpeg7:RatingValue>65.3</mpeg7:RatingValue>
                                    <mpeg7:RatingMetric>
                                        <mpeg7:QualityRatingScheme
                                href="urn:mpeg:mpeg7:cs:MPEG-
7QualityRatingSchemeCS:2.3"/>

   <mpeg7:RatingStyle>higherBetter</mpeg7:RatingStyle>
                                    </mpeg7:RatingMetric>
                                </mpeg7:QualityRating>
                            </mpeg7:MediaQuality>
                        </mpeg7:DescriptionUnit>
                    </mpeg7:Mpeg7>
                </Statement>
            </Descriptor>
        </Selection>
        <Selection select_id="LO_QUALITY">
            <Descriptor id="LO_QUALITY_DESC">
                <Statement mimeType="text/xml">
                    <mpeg7:Mpeg7>
                        <mpeg7:DescriptionUnit xsi:type="mpeg7:MediaProfileType">
                            <mpeg7:MediaFormat>
                                <mpeg7:VisualCodingFormat

href="urn:mpeg:mpeg7:cs:MPEG7VisualCodingFormatCS:3.1.2"/>
                                <mpeg7:BitRate>64000</mpeg7:BitRate>
                            </mpeg7:MediaFormat>
                            <mpeg7:MediaQuality>
                                <mpeg7:QualityRating ratingType="objective">
                                    <mpeg7:RatingValue>35.6</mpeg7:RatingValue>
                                    <mpeg7:RatingMetric>
                                        <mpeg7:QualityRatingScheme
                                href="urn:mpeg:mpeg7:cs:MPEG-
7QualityRatingSchemeCS:2.3"/>

   <mpeg7:RatingStyle>higherBetter</mpeg7:RatingStyle>
                                    </mpeg7:RatingMetric>
                                </mpeg7:QualityRating>
                            </mpeg7:MediaQuality>
                        </mpeg7:DescriptionUnit>
                    </mpeg7:Mpeg7>
                </Statement>
            </Descriptor>
        </Selection>
    </Choice>
    <!-- In each of the following Components, the relevant Descriptors are included
by
         reference using XInclude. -->
    <Component>
        <Condition require="HI_QUALITY"/>
        <Descriptor>
            <xi:include xpointer="element(HI_QUALITY_DESC/1)"/>
```

```
            </Descriptor>
            <Descriptor>
                <xi:include xpointer="element(COMMON_PROFILE_DESC/1)"/>
            </Descriptor>
            <Resource ref="rtsp://telemedia1:/v11.mp4" mimeType="video/mp4v-es"/>
        </Component>
        <Component>
            <Condition require="MED_QUALITY"/>
            <Descriptor>
                <xi:include xpointer="element(MED_QUALITY_DESC/1)"/>
            </Descriptor>
            <Descriptor>
                <xi:include xpointer="element(COMMON_PROFILE_DESC/1)"/>
            </Descriptor>
            <Resource ref="rtsp://telemedia1/v12.mp4" mimeType="video/mp4v-es"/>
        </Component>
        <Component>
            <Condition require="LO_QUALITY"/>
            <Descriptor>
                <xi:include xpointer="element(LO_QUALITY_DESC/1)"/>
            </Descriptor>
            <Descriptor>
                <xi:include xpointer="element(COMMON_PROFILE_DESC/1)"/>
            </Descriptor>
            <Resource ref="rtsp://telemedia1/v13.mp4" mimeType="video/mp4v-es"/>
        </Component>
    </Item>
</DIDL>
```

9.2 Example 2: Expressing the same set of metadata in different descriptor formats

EXAMPLE This example shows how different descriptor formats can be used to express overlapping metadata sets. This allows a wider range of applications to access the metadata. If a particular application understands one format but not the other, it can simply ignore the other descriptor and still make full use of the Digital Item. This example shows MPEG-7 and RDF-based Dublin Core descriptors. Also note that the COMPONENT in this Digital Item has more than one RESOURCE element, indicating multiple equivalent mechanisms for retrieving the same resource data.

```
<?xml version="1.0" encoding="UTF-8"?>
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS"
   xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://purl.org/dc/elements/1.1/">
   <Item>
       <Descriptor>
           <Statement mimeType="text/xml">
               <mpeg7:Mpeg7>
                   <mpeg7:DescriptionUnit xsi:type="mpeg7:CreationInformationType">
                       <mpeg7:Creation>
                           <mpeg7:Title>When the Thistle Blooms</mpeg7:Title>
                           <mpeg7:Creator>
                               <mpeg7:Role
href="urn:mpeg:mpeg7:cs:MPEG7RoleCS:PERFORMER"/>
                               <mpeg7:PersonGroup>
                                   <mpeg7:Name>Always Red</mpeg7:Name>
                               </mpeg7:PersonGroup>
                           </mpeg7:Creator>
                           <mpeg7:Creator>
                               <mpeg7:Role
href="urn:mpeg:mpeg7:cs:MPEG7RoleCS:PUBLISHER"/>
                               <mpeg7:Organization>
                                   <mpeg7:Name>PDQ Records</mpeg7:Name>
                               </mpeg7:Organization>
                           </mpeg7:Creator>
                       </mpeg7:Creation>
                   </mpeg7:DescriptionUnit>
```



```
            </mpeg7:Mpeg7>
        </Statement>
    </Descriptor>
    <Descriptor>
        <Statement mimeType="text/xml">
            <rdf:Description>
                <dc:title>When the Thistle Blooms</dc:title>
                <dc:creator>Always Red</dc:creator>
                <dc:publisher>PDQ Records</dc:publisher>
            </rdf:Description>
        </Statement>
    </Descriptor>
    <Component>
        <Resource ref="rtsp://telemedia1:/v11.mp4" mimeType="audio/mp4a-latm"/>
        <Resource ref="urn:doi:10.1000-1" mimeType="audio/mp4a-latm"/>
    </Component>
  </Item>
</DIDL>
```

9.3 Example 3: A digital music album

EXAMPLE This example shows how a digital music album might be expressed in DIDL. It shows, among other things, how multiple instances of metadata of various formats can coexist within a single Digital Item, and how a single Digital Item can be made configurable in various ways.

```
<?xml version="1.0"?>
<!-- This is a Digital Item Declaration for the (fictitious) musical album "Always Red".
-->
<DIDL xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://purl.org/dc/elements/1.1/"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xi="http://www.w3.org/2001/XInclude"
   xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
   xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS"
   xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS">
 <Declarations>
  <Descriptor id="ALBUM_RATING">
   <Descriptor>
    <!-- A Descriptor always says something about its parent. In this
       case, the Descriptor is describing the parent Descriptor. -->
    <Statement mimeType="text/plain">
     MPEG-7 description for Parental Content ratings defined by ICRA.
    </Statement>
   </Descriptor>
   <Statement mimeType="text/xml">
    <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
     <DescriptionUnit xsi:type="ClassificationType">
      <ParentalGuidance>
       <!-- violence: none -->
       <ParentalRating href="urn:mpeg:mpeg7:cs:ICRAParentalRatingViolenceCS:2001:6"/>
      </ParentalGuidance>
      <ParentalGuidance>
       <!-- explicit-language: none -->
       <ParentalRating href="urn:mpeg:mpeg7:cs:ICRAParentalRatingCS:2001:6"/>
      </ParentalGuidance>
      <ParentalGuidance>
       <!-- sex: none -->
       <ParentalRating href="urn:mpeg:mpeg7:cs:ICRAParentalRatingSexCS:2001:6"/>
      </ParentalGuidance>
     </DescriptionUnit>
    </Mpeg7>
   </Statement>
  </Descriptor>
 </Declarations>
 <Container>
```

```
  <!-- This Container is acting as a delivery package for a particular
     consumer. The package contains information about the package, in
     this case, the recipient's name and the distributor's name. -->
  <Descriptor>
   <Statement mimeType="text/xml">
    <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
     <DescriptionUnit xsi:type="CreationInformationType">
      <Creation>
       <Title xmlns:lang="en">Always Red</Title>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:DISTRIBUTOR"/>
        <Agent xsi:type="OrganizationType">
         <Name>Digital Music Unlimited</Name>
        </Agent>
       </Creator>
      </Creation>
     </DescriptionUnit>
    </Mpeg7>
   </Statement>
  </Descriptor>
  <Descriptor>
   <Statement mimeType="text/xml">
    <!-- This licenseGroup contains two licenses:
       - one from Digital Music Unlimited to John Q. Consumer and
       - one from Acme Records (the copyright holder) to Digital Music Unlimited. -->
    <licenseGroup xmlns="urn:mpeg:mpeg21:2003:01-REL-R-NS">
     <!-- The following license from Digital Music Unlimited allows
        John Q. Consumer to play the digital item for a fee of $10 or $15.
        In the case of the $10 fee, the user is limited to those sources
        with bit rates less than or equal to 128k. -->
     <license xmlns="urn:mpeg:mpeg21:2003:01-REL-R-NS"
xmlns:sx="urn:mpeg:mpeg21:2003:01-REL-SX-NS" xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-
NS" xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
      <inventory>
       <keyHolder licensePartId="user">
        <info>
         <dsig:KeyName>JohnQConsumer</dsig:KeyName>
        </info>
       </keyHolder>
       <mx:diReference licensePartId="di">
        <mx:identifier>urn:mpegRA:mpeg21:dii:acme:CD-ID:a409c80c</mx:identifier>
       </mx:diReference>
      </inventory>
      <!-- The user can play the DI for $10 flat fee using source bit rates no greater
than 131584. -->
      <grant>
       <keyHolder licensePartIdRef="user"/>
       <mx:play/>
       <mx:diReference licensePartIdRef="di"/>
       <allConditions>
        <sx:feeFlat>
         <serviceReference>
          <sx:anonymousStateService/>
          <serviceParameters>
           <datum>
            <keyHolder licensePartIdRef="user"/>
           </datum>
           <datum>
            <mx:diReference licensePartIdRef="di"/>
           </datum>
          </serviceParameters>
         </serviceReference>
         <sx:rate xmlns:iso="urn:mpeg:mpeg21:2003:01-REL-SX-NS:2003:currency">
          <sx:amount>10.00</sx:amount>
          <sx:currency>iso:USD</sx:currency>
         </sx:rate>
        </sx:feeFlat>
        <dia:changeConstraint xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS">
```



```
        <dia:constraint>
         <dia:AdaptationUnitConstraints>
          <!-- Input bitrate must be less than or equal to 131584. -->
          <dia:LimitConstraint>
           <dia:Argument xsi:type="dia:SemanticalDataRefType"
semantics="urn:mpeg:mpeg21:2003:01-DIA-MediaInformationCS-NS:9"/>
           <dia:Argument xsi:type="dia:ConstantDataType">
            <dia:Constant xsi:type="dia:IntegerType">
             <dia:Value>131584</dia:Value>
            </dia:Constant>
           </dia:Argument>
           <dia:Operation operator="urn:mpeg:mpeg21:2003:01-DIA-
StackFunctionOperatorCS-NS:38"/>
          </dia:LimitConstraint>
         </dia:AdaptationUnitConstraints>
        </dia:constraint>
       </dia:changeConstraint>
      </allConditions>
     </grant>
     <!-- The user can play the DI for $15 flat fee using any source bit rates. -->
     <grant>
      <keyHolder licensePartIdRef="user"/>
      <mx:play/>
      <mx:diReference licensePartIdRef="di"/>
      <allConditions>
       <sx:feeFlat>
        <serviceReference>
         <sx:anonymousStateService/>
         <serviceParameters>
          <datum>
           <keyHolder licensePartIdRef="user"/>
          </datum>
          <datum>
           <mx:diReference licensePartIdRef="di"/>
          </datum>
         </serviceParameters>
        </serviceReference>
        <sx:rate xmlns:iso="urn:mpeg:mpeg21:2003:01-REL-SX-NS:2003:currency">
         <sx:amount>15.00</sx:amount>
         <sx:currency>iso:USD</sx:currency>
        </sx:rate>
       </sx:feeFlat>
      </allConditions>
     </grant>
     <issuer>
      <keyHolder>
       <info>
        <dsig:KeyName>DigitalMusicUnlimited</dsig:KeyName>
       </info>
      </keyHolder>
     </issuer>
    </license>
    <!-- The following license from Acme Records allows
       Digital Music Unlimited to issue consumer licenses. -->
    <license>
     <grant>
      <forAll varName="user"/>
      <forAll varName="conditions"/>
      <keyHolder>
       <info>
        <dsig:KeyName>DigitalMusicUnlimited</dsig:KeyName>
       </info>
      </keyHolder>
      <issue/>
      <grant>
       <keyHolder varRef="user"/>
       <mx:play/>
       <mx:diReference>
```



```
        <mx:identifier>urn:mpegRA:mpeg21:dii:acme:CD-ID:a409c80c</mx:identifier>
       </mx:diReference>
       <allConditions varRef="conditions"/>
      </grant>
     </grant>
     <issuer>
      <keyHolder>
       <info>
        <dsig:KeyName>AcmeRecords</dsig:KeyName>
       </info>
      </keyHolder>
     </issuer>
    </license>
   </licenseGroup>
  </Statement>
 </Descriptor>
 <Item>
  <!-- Define a set of descriptive information associated with the
    outermost item - the album as a whole. -->
  <Descriptor>
   <!-- The following statement is an example of an application-specific
      data instance that can be included, in this case, a format
      specific to the Digital Music Unlimited service. This example
      happens to be encoded in plain text, but it is possible to encode
      such data in any format compatible with well-formed XML. -->
   <Statement mimeType="application/x-dmu-content-organizer-hints">
    DMU 9876:: Item Type="Music Album";
   </Statement>
  </Descriptor>
  <Descriptor>
   <xi:include
    xpointer="xpointer(//Descriptor[@id='ALBUM_RATING']/*)element(ALBUM_RATING/2)"/>
  </Descriptor>
  <Descriptor id="PERFORMING_GROUP">
   <Statement mimeType="text/xml">
    <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
     <DescriptionUnit xsi:type="CreationInformationType">
      <Creation>
       <Title xmlns:lang="en">Always Red</Title>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:PERFORMER">
         <Term termID="urn:mpeg:mpeg7:cs:RoleCS:2001:MUSICIAN"/>
        </Role>
        <Agent xsi:type="PersonGroupType">
         <Name>Once Blue</Name>
        </Agent>
       </Creator>
      </Creation>
     </DescriptionUnit>
    </Mpeg7>
   </Statement>
  </Descriptor>
  <Descriptor>
   <Statement mimeType="text/xml">
    <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
     <DescriptionUnit xsi:type="CreationInformationType">
      <Creation>
       <Title xmlns:lang="en">Always Red</Title>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:PERFORMER"/>
        <Agent xsi:type="PersonGroupType">
         <Name>Always Red</Name>
        </Agent>
       </Creator>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:MUSICIAN"/>
        <Agent xsi:type="PersonType">
         <Name>
```

```
          <GivenName>Jack</GivenName>
          <FamilyName>Jake</FamilyName>
         </Name>
        </Agent>
        <Instrument>
         <Tool>
          <Name>Vocals</Name>
         </Tool>
        </Instrument>
       </Creator>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:MUSICIAN"/>
        <Agent xsi:type="PersonType">
         <Name>
          <GivenName>Jack</GivenName>
          <FamilyName>Juno</FamilyName>
         </Name>
        </Agent>
        <Instrument>
         <Tool>
          <Name>Vocals</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Tambourine</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Finger Snaps</Name>
         </Tool>
        </Instrument>
       </Creator>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:MUSICIAN"/>
        <Agent xsi:type="PersonType">
         <Name>
          <GivenName>Joe</GivenName>
          <FamilyName>Dump</FamilyName>
         </Name>
        </Agent>
        <Instrument>
         <Tool>
          <Name>Acoustic Guitar</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Electric Guitar</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Piano</Name>
         </Tool>
        </Instrument>
       </Creator>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:MUSICIAN"/>
        <Agent xsi:type="PersonType">
         <Name>
          <GivenName>Jeff</GivenName>
          <FamilyName>Kelly</FamilyName>
         </Name>
        </Agent>
        <Instrument>
         <Tool>
```

```
          <Name>Acoustic Bass</Name>
         </Tool>
        </Instrument>
       </Creator>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:MUSICIAN"/>
        <Agent xsi:type="PersonType">
         <Name>
          <GivenName>Jim</GivenName>
          <FamilyName>Jinks</FamilyName>
         </Name>
        </Agent>
        <Instrument>
         <Tool>
          <Name>Drums</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Marimba</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Shaker</Name>
         </Tool>
        </Instrument>
        <Instrument>
         <Tool>
          <Name>Carob Pods</Name>
         </Tool>
        </Instrument>
       </Creator>
       <Creator>
        <Role href="urn:mpeg:mpeg7:cs:RoleCS:2001:PUBLISHER"/>
        <Agent xsi:type="OrganizationType">
         <Name>Acme Records</Name>
        </Agent>
       </Creator>
       <CreationCoordinates>
        <Date>
         <TimePoint>1995-10-24</TimePoint>
        </Date>
       </CreationCoordinates>
       <CopyrightString>Copyright 1995, Acme Records, All Rights Reserved.
        Unauthorized duplication is a violation of applicable laws.
       </CopyrightString>
      </Creation>
     </DescriptionUnit>
    </Mpeg7>
   </Statement>
  </Descriptor>
  <Descriptor id="DI_IDENTIFIER">
   <Statement mimeType="text/xml">
    <Identifier xmlns="urn:mpeg:mpeg21:2002:01-DII-NS">urn:mpegRA:mpeg21:dii:acme:CD-
ID:a409c80c</Identifier>
   </Statement>
  </Descriptor>
  <Descriptor id="RIGHTS">
   <Statement mimeType="text/xml">
    <license xmlns="urn:mpeg:mpeg21:2003:01-REL-R-NS"
         xmlns:mx="urn:mpeg:mpeg21:2003:01-REL-MX-NS"
         xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
     <grant>
      <forAll varName="p"/>
      <forAll varName="r"/>
      <principal varRef="p"/>
      <right varRef="r"/>
```

```
        <mx:diReference>
         <mx:identifier>urn:mpegRA:mpeg21:dii:acme:CD-ID:a409c80c</mx:identifier>
        </mx:diReference>
        <prerequisiteRight>
         <principal varRef="p"/>
         <right varRef="r"/>
         <mx:diReference>
          <mx:identifier>urn:mpegRA:mpeg21:dii:acme:CD-ID:a409c80c</mx:identifier>
         </mx:diReference>
         <keyHolder>
          <info>
           <dsig:KeyName>AcmeRecords</dsig:KeyName>
          </info>
         </keyHolder>
        </prerequisiteRight>
       </grant>
       <issuer>
        <keyHolder>
         <info>
          <dsig:KeyName>DigitalCopyrightService</dsig:KeyName>
         </info>
        </keyHolder>
       </issuer>
      </license>
     </Statement>
    </Descriptor>
    <Descriptor id="ICON_FILE">
     <Descriptor>
      <Statement mimeType="application/x-dmu-content-organizer-hints">
       DMU 9876:: Descriptor Type="Item Icon";
      </Statement>
     </Descriptor>
     <Component>
      <Resource ref="http://www.dmu.com/always_red/B002U0A.t.jpg" mimeType="image/jpeg"/>
     </Component>
    </Descriptor>
    <!-- This is the outermost Item, which represents the musical album
       as a whole. -->
    <Choice choice_id="PLATFORM_Choice" minSelections="1" maxSelections="1">
     <!-- This choice allows the item to be configured for a specific
        target platform: Windows, Linux, or Mac. -->
     <Selection select_id="PLATFORM_WINDOWS">
      <Descriptor>
       <Statement mimeType="text/xml">
        <RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:uaprof="http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405">
         <Description>
          <type resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20000405#SoftwarePlatform"/>
          <uaprof:OSName>Windows</uaprof:OSName>
          <uaprof:OSVersion>XP</uaprof:OSVersion>
          <uaprof:OSVendor>Microsoft</uaprof:OSVendor>
         </Description>
        </RDF>
       </Statement>
      </Descriptor>
     </Selection>
     <Selection select_id="PLATFORM_LINUX">
      <Descriptor>
       <Statement mimeType="text/xml">
        <RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:uaprof="http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405">
         <Description>
          <type resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20000405#SoftwarePlatform"/>
          <uaprof:OSName>Linux</uaprof:OSName>
          <uaprof:OSVersion>9.0</uaprof:OSVersion>
          <uaprof:OSVendor>SuSE</uaprof:OSVendor>
```

```
         </Description>
        </RDF>
       </Statement>
      </Descriptor>
     </Selection>
     <Selection select_id="PLATFORM_MAC">
      <Descriptor>
       <Statement mimeType="text/xml">
        <RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:uaprof="http://www.wapforum.org/profiles/UAPROF/ccppschema-20000405">
         <Description>
          <type resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20000405#SoftwarePlatform"/>
          <uaprof:OSName>MacOS</uaprof:OSName>
          <uaprof:OSVersion>X</uaprof:OSVersion>
          <uaprof:OSVendor>Apple</uaprof:OSVendor>
         </Description>
        </RDF>
       </Statement>
      </Descriptor>
     </Selection>
    </Choice>
    <Choice choice_id="ALL_SONGS">
     <!-- This choice allows the user to decide whether to filter out
        some of the songs on the album. The plain-text statements
        contain text that can be used in GUI dialogs. -->
     <Descriptor>
      <Statement mimeType="text/xml">
       <!-- UserSelection from ISO/IEC 21000-7 indicates Choice is
          intended to be configured by the user. -->
       <diac:UserSelection xmlns:diac="urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS"/>
      </Statement>
     </Descriptor>
     <Selection select_id="PICK_SONGS">
      <Descriptor>
       <Statement mimeType="text/plain">
        I want to choose among the individual songs.
       </Statement>
      </Descriptor>
     </Selection>
    </Choice>
    <Choice choice_id="SONG_PICKER" default="SONG1 SONG2 SONG3 SONG4">
     <!-- This choice presents the user with the list of songs to choose
        from. It is conditional upon the PICK_SONGS selection from the
        previous choice, so it will be processed only if the PICK_SONGS
        selection is made. -->
     <Condition require="PICK_SONGS"/>
     <Descriptor>
      <Statement mimeType="text/xml">
       <diac:UserSelection xmlns:diac="urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS"/>
      </Statement>
     </Descriptor>
     <Descriptor>
      <Statement mimeType="text/plain">
       Choose the songs you would like:
      </Statement>
     </Descriptor>
     <Selection select_id="SONG1">
      <Descriptor>
       <!-- This descriptor contains an XInclude reference to
          the content of another descriptor. The effect is
          that the included items from the referenced
          descriptor (which defines the title of the first
          song) are logically duplicated within this, the
          referring descriptor. This allows the actual value
          of the title to be kept in a single location within
          the document. -->
       <xi:include xpointer="element(SONG1_TITLE/1)"/>
```

```
      </Descriptor>
     </Selection>
     <Selection select_id="SONG2">
      <Descriptor>
       <xi:include xpointer="element(SONG2_TITLE/1)"/>
      </Descriptor>
     </Selection>
     <Selection select_id="SONG3">
      <Descriptor>
       <xi:include xpointer="element(SONG3_TITLE/1)"/>
      </Descriptor>
     </Selection>
     <Selection select_id="SONG4">
      <Descriptor>
       <xi:include xpointer="element(SONG4_TITLE/1)"/>
      </Descriptor>
     </Selection>
    </Choice>
    <Choice choice_id="BITRATE_Choice" default="LOW_BITRATE" minSelections="1"
     maxSelections="1">
     <Descriptor>
      <Statement mimeType="text/xml">
       <diac:UserSelection xmlns:diac="urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS"/>
      </Statement>
     </Descriptor>
     <Descriptor>
      <Statement mimeType="text/plain">
       Please select the fidelity you would prefer.
      </Statement>
     </Descriptor>
     <Selection select_id="LOW_BITRATE">
      <Descriptor>
       <Descriptor>
        <Statement mimeType="text/plain">Low (128 Kbits/sec, $10.00).</Statement>
       </Descriptor>
       <Statement mimeType="text/xml">
        <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
         <DescriptionUnit xsi:type="MediaFormatType">
          <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:1"/>
          <BitRate>131584</BitRate>
         </DescriptionUnit>
        </Mpeg7>
       </Statement>
      </Descriptor>
     </Selection>
     <Selection select_id="HIGH_BITRATE">
      <Descriptor>
       <Descriptor>
        <Statement mimeType="text/plain">High (192 Kbits/sec, $15.00).</Statement>
       </Descriptor>
       <Statement mimeType="text/plain">
        <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
         <DescriptionUnit xsi:type="MediaFormatType">
          <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:1"/>
          <BitRate>196608</BitRate>
         </DescriptionUnit>
        </Mpeg7>
       </Statement>
      </Descriptor>
     </Selection>
    </Choice>
    <Choice choice_id="EXTRA_CONTENT_Choice" minSelections="1" maxSelections="1">
     <Descriptor>
      <Statement mimeType="text/xml">
       <diac:UserSelection xmlns:diac="urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS"/>
      </Statement>
     </Descriptor>
     <Descriptor>
```



```
     <Statement mimeType="text/plain">
      Would you like to get supplemental content for this album?
     </Statement>
    </Descriptor>
    <Selection select_id="WANT_EXTRA_CONTENT">
     <Descriptor>
      <Statement mimeType="text/plain">
       Yes, get all of the supplemental content.
      </Statement>
     </Descriptor>
    </Selection>
    <Selection select_id="ASK_EXTRA_CONTENT">
     <Descriptor>
      <Statement mimeType="text/plain">
       Yes, but let me choose which items to get.
      </Statement>
     </Descriptor>
    </Selection>
    <Selection select_id="DONT_ASK_ABOUT_CONTENT">
     <Descriptor>
      <Statement mimeType="text/plain">
       No, I don't want any of the extra stuff.
      </Statement>
     </Descriptor>
    </Selection>
   </Choice>
   <Choice choice_id="COVER_ART_Choice" minSelections="1" maxSelections="1">
    <Condition require="ASK_EXTRA_CONTENT"/>
    <Descriptor>
     <Statement mimeType="text/xml">
      <diac:UserSelection xmlns:diac="urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS"/>
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/plain">
      Supplemental content:
     </Statement>
    </Descriptor>
    <Selection select_id="GET_ART">
     <Descriptor>
      <Statement mimeType="text/plain">
       Include the cover art.
      </Statement>
     </Descriptor>
    </Selection>
    <Selection select_id="GET_LYRICS">
     <Descriptor>
      <Statement mimeType="text/plain">
       Include the song lyrics.
      </Statement>
     </Descriptor>
    </Selection>
    <Selection select_id="GET_VIDEO">
     <Condition require="PLATFORM_WINDOWS"/>
     <Condition require="PLATFORM_MAC"/>
     <Descriptor>
      <Statement mimeType="text/plain">
       Include video footage from the latest concert.
      </Statement>
     </Descriptor>
    </Selection>
    <Selection select_id="GET_REVIEWS">
     <Descriptor>
      <Statement mimeType="text/plain">
       Include press reviews.
      </Statement>
     </Descriptor>
    </Selection>
```

```
    </Choice>
    <Choice choice_id="IMAGE_FORMAT_Choice" default="JPEG_IMAGE">
     <!-- This choice allows the package to be configured for platforms
        that support specific image formats. -->
     <Condition require="GET_ART"/>
     <Condition require="WANT_EXTRA_CONTENT"/>
     <Selection select_id="JPEG_IMAGE">
      <Descriptor>
       <Statement mimeType="text/xml">
        <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
         <DescriptionUnit xsi:type="MediaFormatType">
          <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:4.1"/>
          <VisualCoding>
           <Format href="urn:mpeg:mpeg7:cs:VisualCodingFormatCS:2001:4">
            <Name>JPEG</Name>
           </Format>
          </VisualCoding>
         </DescriptionUnit>
        </Mpeg7>
       </Statement>
      </Descriptor>
     </Selection>
     <Selection select_id="GIF_IMAGE">
      <Descriptor>
       <Statement mimeType="text/xml">
        <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
         <DescriptionUnit xsi:type="MediaFormatType">
          <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:4.1"/>
          <VisualCoding>
           <Format href="urn:mpeg:mpeg7:cs:FileFormatCS:2001:12">
            <Name>gif</Name>
           </Format>
          </VisualCoding>
         </DescriptionUnit>
        </Mpeg7>
       </Statement>
      </Descriptor>
     </Selection>
     <Selection select_id="BMP_IMAGE">
      <Condition require="PLATFORM_WINDOWS"/>
      <Descriptor>
       <Statement mimeType="text/xml">
        <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
         <DescriptionUnit xsi:type="MediaFormatType">
          <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:4.1"/>
          <VisualCoding>
           <Format href="urn:mpeg:mpeg7:cs:FileFormatCS:2001:11">
            <Name>bmp</Name>
           </Format>
          </VisualCoding>
         </DescriptionUnit>
        </Mpeg7>
       </Statement>
      </Descriptor>
     </Selection>
    </Choice>
    <Choice choice_id="IMAGE_SIZE_Choice" default="LARGE_IMAGE">
     <!-- This choice allows the user to decide what size the cover image
        should be. However, since we only have one size of the BMP version
        of the image, the choice is made conditional on the Windows BMP
        selection *not* being made in the previous choice. -->
     <Condition require="GET_ART" except="BMP_IMAGE"/>
     <Condition require="WANT_EXTRA_CONTENT" except="BMP_IMAGE"/>
     <Descriptor>
      <Statement mimeType="text/xml">
       <diac:UserSelection xmlns:diac="urn:mpeg:mpeg21:2003:01-DIA-DIAC-NS"/>
      </Statement>
```

```
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/plain">
      Please select the image size you would like:
     </Statement>
    </Descriptor>
    <Selection select_id="LARGE_IMAGE">
     <Descriptor>
      <Descriptor>
       <Statement mimeType="text/plain">Large (301x300)</Statement>
      </Descriptor>
      <Statement mimeType="text/xml">
       <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
        <DescriptionUnit xsi:type="MediaFormatType">
         <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:4"/>
         <VisualCoding>
          <Frame width="301" height="300"/>
         </VisualCoding>
        </DescriptionUnit>
       </Mpeg7>
      </Statement>
     </Descriptor>
    </Selection>
    <Selection select_id="SMALL_IMAGE">
     <Descriptor>
      <Descriptor>
       <Statement mimeType="text/plain">Small (130x130)</Statement>
      </Descriptor>
      <Statement mimeType="text/plain">
       <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
        <DescriptionUnit xsi:type="MediaFormatType">
         <Content href="urn:mpeg:mpeg7:cs:ContentCS:2001:4"/>
         <VisualCoding>
          <Frame width="130" height="130"/>
         </VisualCoding>
        </DescriptionUnit>
       </Mpeg7>
      </Statement>
     </Descriptor>
    </Selection>
   </Choice>
   <!-- The components listed here pertain to the album as a whole. The
      first two components contain anchors. An anchor is a point (or
      range) of interest in a resource. The result of the arrangement
      of the given three anchors is that the application can present three
      different "entry points" into the content. The default "entry point"
      will be the first anchor, since it has the highest precedence value. -->
   <Component>
    <Condition except="PICK_SONGS"/>
    <Condition require="SONG1 SONG2 SONG3 SONG4"/>
    <Resource ref="http://www.dmu.com/always_red/always_red.m3u"
      mimeType="audio/x-mpegurl"/>
    <Anchor precedence="1000">
     <!-- This anchor does not specify a fragment, so it is denoting
        the beginning of the M3U file as its "entry point" -->
     <Descriptor>
      <Statement mimeType="text/plain">
       Play Entire Album
      </Statement>
     </Descriptor>
    </Anchor>
   </Component>
   <Component>
    <Resource ref="http://www.dmu.com/always_red/always_red.html"
      mimeType="text/html"/>
    <Anchor precedence="500">
     <!-- This anchor does not specify a fragment, so it is denoting
```



```
      the beginning of the HTML file as its "entry point" -->
   <Descriptor>
    <Statement mimeType="text/plain">
     View Table of Contents
    </Statement>
   </Descriptor>
  </Anchor>
  <Anchor precedence="100">
   <!-- This anchor specifies a fragment value of "credits", so its
      entry point is determined by appending "#credits" onto the
      URI specified in the ref attribute of the resource (above). -->
   <Descriptor>
    <Statement mimeType="text/plain">
     View Credits
    </Statement>
   </Descriptor>
   <Fragment fragmentId="credits"/>
  </Anchor>
 </Component>
 <Component>
  <Condition require="GIF_IMAGE LARGE_IMAGE"/>
  <Descriptor id="COVER_ART_DESC">
   <Statement mimeType="application/x-dmu-content-organizer-hints">
    DMU 9876:: Descriptor Type="Album Cover Art";
   </Statement>
  </Descriptor>
  <Resource ref="http://www.dmu.com/always_red/B002U0A.l.gif"
    mimeType="image/gif"/>
 </Component>
 <Component>
  <Condition require="JPEG_IMAGE LARGE_IMAGE"/>
  <Condition require="WANT_EXTRA_CONTENT"/>
  <Descriptor>
   <xi:include xpointer="element(COVER_ART_DESC/1)"/>
  </Descriptor>
  <Resource ref="http://www.dmu.com/always_red/B002U0A.l.jpg"
    mimeType="image/jpeg"/>
 </Component>
 <Component>
  <Condition require="BMP_IMAGE LARGE_IMAGE"/>
  <Descriptor>
   <xi:include xpointer="element(COVER_ART_DESC/1)"/>
  </Descriptor>
  <Resource ref="http://www.dmu.com/always_red/B002U0A.l.bmp"
    mimeType="image/x-ms-bmp"/>
 </Component>
 <Component>
  <Condition require="GIF_IMAGE SMALL_IMAGE"/>
  <Descriptor>
   <xi:include xpointer="element(COVER_ART_DESC/1)"/>
  </Descriptor>
  <Resource ref="http://www.dmu.com/always_red/B002U0A.m.gif"
    mimeType="image/gif"/>
 </Component>
 <Component>
  <Condition require="JPEG_IMAGE SMALL_IMAGE"/>
  <Descriptor>
   <xi:include xpointer="element(COVER_ART_DESC/1)"/>
  </Descriptor>
  <Resource ref="http://www.dmu.com/always_red/B002U0A.m.jpg"
    mimeType="image/jpeg"/>
 </Component>
 <Component>
  <Resource ref="http://www.dmu.com/always_red/B002U0A.t.gif"
    mimeType="image/gif"/>
 </Component>
 <Component>
  <Condition require="GET_VIDEO"/>
```

```
    <Condition require="WANT_EXTRA_CONTENT"/>
    <Resource ref="http://www.dmu.com/always_red/clueful.mov"
      mimeType="video/quicktime"/>
   </Component>
   <Component>
    <Condition require="GET_REVIEWS"/>
    <Condition require="WANT_EXTRA_CONTENT"/>
    <Descriptor id="REVIEW_DESC">
     <Statement mimeType="application/x-dmu-content-organizer-hints">
      DMU 9876:: Component Type="Press Review";
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <rdf:RDF>
       <rdf:Description>
        <dc:creator>Bob Jones</dc:creator>
        <dc:source>
     http://www.foo-review.com/exe/music-reviews/B002U0A/qid=9183/002-0154-2910
        </dc:source>
       </rdf:Description>
      </rdf:RDF>
     </Statement>
    </Descriptor>
    <Resource ref="http://www.dmu.com/always_red/review1.txt"
      mimeType="text/plain"/>
   </Component>
   <Component>
    <Condition require="GET_REVIEWS"/>
    <Condition require="WANT_EXTRA_CONTENT"/>
    <Descriptor>
     <xi:include xpointer="element(REVIEW_DESC/1)"/>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <rdf:RDF>
       <rdf:Description>
        <dc:creator>Anonymous</dc:creator>
        <dc:source>
     http://www.foo-review.com/exe/cust-reviews/B002U0A/qid=9183/002-0154-2910
        </dc:source>
       </rdf:Description>
      </rdf:RDF>
     </Statement>
    </Descriptor>
    <Resource ref="http://www.dmu.com/always_red/review2.txt"
      mimeType="text/plain"/>
   </Component>
   <Component>
    <Condition require="GET_REVIEWS"/>
    <Condition require="WANT_EXTRA_CONTENT"/>
    <Descriptor>
     <xi:include xpointer="element(REVIEW_DESC/1)"/>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <rdf:RDF>
       <rdf:Description>
        <dc:creator>Sander Wolf</dc:creator>
        <dc:source>
   http://www.foopaperbar.com/alt1/archive/music/reviews/03-14-96/OTR/ALWAYS_RED.html
        </dc:source>
       </rdf:Description>
      </rdf:RDF>
     </Statement>
    </Descriptor>
    <Resource ref="http://www.dmu.com/always_red/review3.txt"
      mimeType="text/plain"/>
```



```
    </Component>
    <!-- Each of the following items represents a single song on the album -->
    <Item>
     <Condition require="SONG1"/>
     <Descriptor>
      <Statement mimeType="application/x-dmu-content-organizer-hints">
       DMU 9876:: Item Type="Song";
      </Statement>
     </Descriptor>
     <Descriptor id="SONG1_TITLE">
      <Statement mimeType="text/xml">
       <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
        <DescriptionUnit xsi:type="CreationInformationType">
         <Creation>
          <Title xmlns:lang="en">Save It</Title>
         </Creation>
        </DescriptionUnit>
       </Mpeg7>
      </Statement>
     </Descriptor>
     <Descriptor>
      <Statement mimeType="text/xml">
       <rdf:RDF>
        <rdf:Description>
         <dc:coverage>233</dc:coverage>
        </rdf:Description>
       </rdf:RDF>
      </Statement>
     </Descriptor>
     <Descriptor>
      <xi:include xpointer="element(RIGHTS/1)"/>
     </Descriptor>
     <Descriptor>
      <xi:include
      xpointer="xpointer(//Descriptor[@id='ALBUM_RATING']/*)element(ALBUM_RATING/2)"/>
     </Descriptor>
     <Component>
      <Condition require="LOW_BITRATE"/>
      <Resource ref="http://www.dmu.com/always_red/01_Save_It.mp3"
        mimeType="audio/mpeg"/>
      <Anchor precedence="50">
       <Descriptor>
        <Statement mimeType="text/plain">
         Play Song
        </Statement>
       </Descriptor>
      </Anchor>
     </Component>
     <Component>
      <Condition require="HIGH_BITRATE"/>
      <Resource ref="http://www.dmu.com/always_red/01_Save_It_192.mp3"
        mimeType="audio/mpeg"/>
      <Anchor precedence="100">
       <Descriptor>
        <Statement mimeType="text/plain">
         Play Song
        </Statement>
       </Descriptor>
      </Anchor>
     </Component>
     <Component>
      <Condition require="GET_LYRICS"/>
      <Condition require="WANT_EXTRA_CONTENT"/>
      <Resource ref="http://www.dmu.com/always_red/Save_It.txt"
        mimeType="text/plain"/>
      <Anchor precedence="25">
       <Descriptor>
        <Statement mimeType="text/plain">
```



```
        View Lyrics
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
   </Item>
   <Item>
    <Condition require="SONG2"/>
    <Descriptor>
     <Statement mimeType="application/x-dmu-content-organizer-hints">
      DMU 9876:: Item Type="Song";
     </Statement>
    </Descriptor>
    <Descriptor id="SONG2_TITLE">
     <Statement mimeType="text/xml">
      <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
       <DescriptionUnit xsi:type="CreationInformationType">
        <Creation>
         <Title xmlns:lang="en">I Haven’t been Anywhere</Title>
        </Creation>
       </DescriptionUnit>
      </Mpeg7>
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <rdf:RDF>
       <rdf:Description>
        <dc:coverage>193</dc:coverage>
       </rdf:Description>
      </rdf:RDF>
     </Statement>
    </Descriptor>
    <Descriptor>
     <xi:include xpointer="element(RIGHTS/1)"/>
    </Descriptor>
    <Descriptor>
     <xi:include
     xpointer="xpointer(//Descriptor[@id='ALBUM_RATING']/*)element(ALBUM_RATING/2)"/>
    </Descriptor>
    <Component>
     <Condition require="LOW_BITRATE"/>
     <Resource ref="http://www.dmu.com/always_red/02_I_Haven't_been_Anywhere.mp3"
       mimeType="audio/mpeg"/>
     <Anchor precedence="50">
      <Descriptor>
       <Statement mimeType="text/plain">
        Play Song
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
    <Component>
     <Condition require="HIGH_BITRATE"/>
     <Resource ref="http://www.dmu.com/always_red/02_I_Haven't_been_Anywhere_192.mp3"
       mimeType="audio/mpeg"/>
     <Anchor precedence="100">
      <Descriptor>
       <Statement mimeType="text/plain">
        Play Song
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
    <Component>
     <Condition require="WANT_EXTRA_CONTENT"/>
     <Condition require="GET_LYRICS"/>
     <Resource ref="http://www.dmu.com/always_red/I_Havent_been_Anywhere.txt"
```



```
       mimeType="text/plain"/>
     <Anchor precedence="25">
      <Descriptor>
       <Statement mimeType="text/plain">
        View Lyrics
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
   </Item>
   <Item>
    <Condition require="SONG3"/>
    <Descriptor>
     <Statement mimeType="application/x-dmu-content-organizer-hints">
      DMU 9876:: Item Type="Song";
     </Statement>
    </Descriptor>
    <Descriptor id="SONG3_TITLE">
     <Statement mimeType="text/xml">
      <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
       <DescriptionUnit xsi:type="CreationInformationType">
        <Creation>
         <Title xmlns:lang="en">Sawdust and Sticks</Title>
        </Creation>
       </DescriptionUnit>
      </Mpeg7>
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <rdf:RDF>
       <rdf:Description>
        <dc:coverage>209</dc:coverage>
       </rdf:Description>
      </rdf:RDF>
     </Statement>
    </Descriptor>
    <Descriptor>
     <xi:include
     xpointer="xpointer(//Descriptor[@id='ALBUM_RATING']/*)element(ALBUM_RATING/2)"/>
    </Descriptor>
    <Descriptor>
     <xi:include xpointer="element(RIGHTS/1)"/>
    </Descriptor>
    <Component>
     <Condition require="LOW_BITRATE"/>
     <Resource ref="http://www.dmu.com/always_red/03_Sawdust_and_Sticks.mp3"
       mimeType="audio/mpeg"/>
     <Anchor precedence="50">
      <Descriptor>
       <Statement mimeType="text/plain">
        Play Song
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
    <Component>
     <Condition require="HIGH_BITRATE"/>
     <Resource ref="http://www.dmu.com/always_red/03_Sawdust_and_Sticks_192.mp3"
       mimeType="audio/mpeg"/>
     <Anchor precedence="100">
      <Descriptor>
       <Statement mimeType="text/plain">
        Play Song
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
```

```
    <Component>
     <Condition require="GET_LYRICS"/>
     <Condition require="WANT_EXTRA_CONTENT"/>
     <Resource ref="http://www.dmu.com/always_red/Sawdust_and_Sticks.txt"
       mimeType="text/plain"/>
     <Anchor precedence="25">
      <Descriptor>
       <Statement mimeType="text/plain">
        View Lyrics
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
   </Item>
   <Item>
    <Condition require="SONG4"/>
    <Descriptor>
     <Statement mimeType="application/x-dmu-content-organizer-hints">
      DMU 9876:: Item Type="Song";
     </Statement>
    </Descriptor>
    <Descriptor id="SONG4_TITLE">
     <Statement mimeType="text/xml">
      <Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001">
       <DescriptionUnit xsi:type="CreationInformationType">
        <Creation>
         <Title xmlns:lang="en">When the Thistle Blooms</Title>
        </Creation>
       </DescriptionUnit>
      </Mpeg7>
     </Statement>
    </Descriptor>
    <Descriptor>
     <Statement mimeType="text/xml">
      <rdf:RDF>
       <rdf:Description>
        <dc:coverage>235</dc:coverage>
       </rdf:Description>
      </rdf:RDF>
     </Statement>
    </Descriptor>
    <Descriptor>
     <xi:include
     xpointer="xpointer(//Descriptor[@id='ALBUM_RATING']/*)element(ALBUM_RATING/2)"/>
    </Descriptor>
    <Descriptor>
     <xi:include xpointer="element(RIGHTS/1)"/>
    </Descriptor>
    <Component>
     <Condition require="LOW_BITRATE"/>
     <Resource ref="http://www.dmu.com/always_red/04_When_the_Thistle_Blooms.mp3"
       mimeType="audio/mpeg"/>
     <Anchor precedence="50">
      <Descriptor>
       <Statement mimeType="text/plain">
        Play Song
       </Statement>
      </Descriptor>
     </Anchor>
    </Component>
    <Component>
     <Condition require="HIGH_BITRATE"/>
     <Resource ref="http://www.dmu.com/always_red/04_When_the_Thistle_Blooms_192.mp3"
       mimeType="audio/mpeg"/>
     <Anchor precedence="100">
      <Descriptor>
       <Statement mimeType="text/plain">
        Play Song
```



```
        </Statement>
       </Descriptor>
      </Anchor>
     </Component>
     <Component>
      <Condition require="GET_LYRICS"/>
      <Condition require="WANT_EXTRA_CONTENT"/>
      <Resource ref="http://www.dmu.com/always_red/When_the_Thistle_Blooms.txt"
        mimeType="text/plain"/>
      <Anchor precedence="25">
       <Descriptor>
        <Statement mimeType="text/plain">
         View Lyrics
        </Statement>
       </Descriptor>
      </Anchor>
     </Component>
    </Item>
   </Item>
  </Container>
</DIDL>
```

9.4 Example 4: Implementing numeric comparisons in Item configuration

EXAMPLE This example shows how to implement more sophisticated kinds of selection criteria than that which has been shown in the previous configuration examples. In the following example, the first CHOICE is to determine the communication speed. In the choice_id="SPEED", there are three sets of speeds to select such as "HIGH_SPEED", "MED_SPEED" and "LOW_SPEED". The "HIGH_SPEED" corresponds to the communication line speed exceeding 8Mbps. The "MED_SPEED" corresponds to the communication line speed between 128Kbps and 8Mbps. The "LOW_SPEED" corresponds to the communication line speed less than 128Kbps. These relational criteria can be expressed in custom STATEMENTS within each SELECTION. In this example, these comparisons are expressed in a short script fragment. The second CHOICE with choice_id="VIDEO_FORMAT" contains three select_ids like "MPEG2_FORMAT", "MPEG4_FORMAT" and "QTIME_FORMAT" as described in the XML document example. Each SELECTION is connected by CONDITION require="HIGH_SPEED", MED_SPEED ", or "LOW_SPEED", respectively. Hence, if some end-user who has a 2 Mbps communication line tries to configure the Digital Item using the following XML example, the MED_SPEED can be selected. This selection, in turn, would allow the end-user to select one of the two components such as MPEG4_FORMAT and QTIME_FORMAT that are located in the "movie1-mpg4-mpl2.asf" and "movie1.mov", respectively.


```
<?xml version="1.0"?>
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
   <Item id="MOVIE_Item1">
       <Descriptor>
           <Statement mimeType="text/plain"> The Terminator. </Statement>
       </Descriptor>
       <!-- ################################## Choice
##################################-->
       <Choice minSelections="1" maxSelections="1" choice_id="SPEED">
           <Descriptor>
               <Statement mimeType="text/plain">
                   Select a proper communication line for your Digital Item
Configuration.
               </Statement>
           </Descriptor>
           <!-- This choice uses an application-specific format. The format identifier
                "x-app/x-script" identifies a software module that can interpret
                the script fragments, which can help the application make the appropriate
                selection. -->
           <Descriptor>
               <Statement mimeType="application/x-script">
                   var speed = system.connectionSpeed();
               </Statement>
           </Descriptor>
           <Selection select_id="HIGH_SPEED">
               <Descriptor>
                   <Statement mimeType="text/plain"> ADSL </Statement>
               </Descriptor>
               <Descriptor>
                   <Statement mimeType="application/x-script ">
                       (speed &gt;= 8000 ? 1 : 0);
                   </Statement>
               </Descriptor>
           </Selection>
           <Selection select_id="MED_SPEED">
               <Descriptor>
                   <Statement mimeType="text/plain"> ISDN </Statement>
               </Descriptor>
               <Descriptor>
                   <Statement mimeType="application/x-script ">
                       (((speed &gt;= 128) &amp;&amp; (speed &lt; 8000)) ? 1 : 0);
                   </Statement>
               </Descriptor>
           </Selection>
           <Selection select_id="LOW_SPEED">
               <Descriptor>
                   <Statement mimeType="text/plain"> MODEM </Statement>
               </Descriptor>
               <Descriptor>
                   <Statement mimeType="application/x-script ">
                       (speed &lt; 128 ? 1 : 0);
                   </Statement>
               </Descriptor>
           </Selection>
       </Choice>
       <Choice minSelections="1" maxSelections="1" choice_id="VIDEO_FORMAT">
           <Descriptor>
               <Statement mimeType="text/plain">
                   Select a proper Video Format for your Digital Item Configuration.
               </Statement>
           </Descriptor>
           <Selection select_id="MPEG2_FORMAT">
               <Condition require="HIGH_SPEED"/>
               <Descriptor>
                   <Statement mimeType="text/plain"> MPEG-2 </Statement>
               </Descriptor>
           </Selection>
           <Selection select_id="MPEG4_FORMAT">
```

```
            <Condition require="LOW_SPEED"/>
            <Condition require="MED_SPEED"/>
            <Descriptor>
               <Statement mimeType="text/plain"> MPEG-4 </Statement>
            </Descriptor>
         </Selection>
         <Selection select_id="QTIME_FORMAT">
            <Condition require="LOW_SPEED"/>
            <Descriptor>
               <Statement mimeType="text/plain"> QuickTime </Statement>
            </Descriptor>
         </Selection>
      </Choice>
      <!-- ################################# Component
#################################-->
      <Component id="MyMovie1-MPEG2-MPML">
         <Condition require="HIGH_SPEED MPEG2_FORMAT"/>
         <Resource ref="movie1-mpg2-mpml.mpg" mimeType="video/mpeg"/>
      </Component>
      <Component id="MyMovie1-MPEG4-SPL2">
         <Condition require="LOW_SPEED MPEG4_FORMAT"/>
         <Resource ref="movie1-mpg4-spl2.asf" mimeType="video/x-ms-asf"/>
      </Component>
      <Component id="MyMovie1-MPEG4-MPL2">
         <Condition require="MED_SPEED MPEG4_FORMAT"/>
         <Resource ref="movie1-mpg4-mpl2.asf" mimeType="video/x-ms-asf"/>
      </Component>
      <Component id="MyMovie1-QTIME">
         <Condition require="LOW_SPEED QTIME_FORMAT"/>
         <Resource ref="movie1.mov" mimeType="video/quicktime"/>
      </Component>
   </Item>
</DIDL>
```


Annex A
(informative)

Patent statements

The International Organization for Standardization and the International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this part of ISO/IEC 21000 may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Information may be obtained from the companies listed below.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 21000 may be the subject of patent rights other than those identified in this annex. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

The list below summarizes the formal patent statements received.

- Matsushita Electric Industrial Co., Ltd
- Mitsubishi Electric Corp.

Annex B
(informative)

Differences with ISO/IEC 21000-2:2003

B.1 Introduction

This Annex lists key differences with the first edition of this part of ISO//IEC 21000.

B.2 Attribute-based descriptors

In the first edition of this part of ISO/IEC 21000, a *descriptor* is represented by a DESCRIPTOR element. This second edition extends the representation of a *descriptor* by also allowing any attributes from other namespaces on elements that may have a DESCRIPTOR child element. That is, a *descriptor* may be represented by a child DESCRIPTOR element and/or an attribute from an other namespace.

B.3 <Resource>

A number of revisions have been made to the RESOURCE element to expand its capabilities and to clarify certain cases.

The content model for the RESOURCE element has been expanded to allow well-formed XML content. This enables a RESOURCE element to represent an XML-based *resource* that is defined by value.

In the first edition of this part of ISO/IEC 21000, if a text-based *resource* was defined by value, the *resource*'s data could not be included as base64 encoded data, since there was no way to signal that the text data was base64 encoded data. Further, a *resource* in general might be encoded, regardless of whether or not it was defined by value in the DID. Thus, a generic mechanism to signal the content-encoding of a *resource* has been added by adding an optional CONTENTENCODING attribute to the RESOURCE element. This allows, for example, a GZIP compressed MP3 *resource* to be represented in a DID. However, if a *resource* is defined by value, it is still necessary to indicate whether the *resource*'s data included in the content of the RESOURCE element is base64 encoded or not. This is indicated by the ENCODING attribute.

The LOCALPATH attribute of the RESOURCE element in the first edition of this part of ISO/IEC 21000 is not necessary in regards to enabling the RESOURCE element to represent a *resource*. Its functionality is primarily application dependent. Hence the LOCALPATH attribute has been removed from the RESOURCE element in this second edition. If required, any application dependent functionality previously implemented the LOCALPATH attribute can now be implemented by way of an attribute-based descriptor (see B.2).

B.4 <Statement>

In the first edition of this part of ISO/IEC 21000 a *statement* could only be defined by value, that is the value of the *statement* was required to be included in the content of the STATEMENT element. This second edition extends the functionality of the STATEMENT element by allowing the *statement* to be defined by reference. This is done in a manner that is synchronized with the RESOURCE element, that is by adding a REF attribute to the STATEMENT element. The STATEMENT element is further synchronized with the RESOURCE element in regards to the CONTENTENCODING and ENCODING attributes, as described in B.3.

B.5 <Anchor> and <Fragment>

In the first edition of this part of ISO/IEC 21000 a *fragment* was represented by the FRAGMENT attribute of the ANCHOR element. The value of the FRAGMENT attribute identified the *resource* fragment by a URI fragment identifier. This second edition extends the representation of a *fragment* by also allowing the *resource* fragment to be identified by well-formed XML metadata (for example using MPEG-7 media locators). The resource fragment can be identified by a URI fragment identifier, by XML-based metadata, or by a combination whereby the XML-based metadata further locates a sub-location or sub-part within the location or part identified by the URI fragment identifier.

B.6 <Condition>

This second edition of ISO/IEC 21000 clarifies the validation rule for the CONDITION element regarding the location of associated SELECTION elements. In the first edition the rule stated that the associated SELECTION elements are required to be "located somewhere within an ITEM element that is an ancestor of the CONDITION". This meant that the associated SELECTION elements could be located anywhere within the hierarchy of the top-most ancestor Item (and provided other validation rules were met), potentially requiring an entire search of the document. The validation rule has been clarified to state that the associated SELECTION elements are required to be located "in a CHOICE that is a child of an ITEM element that is an ancestor of the CONDITION". This limits the location of the associated SELECTION elements to CHOICE elements in the direct ancestral line of the CONDITION.

B.7 <Reference> and XInclude

In December 2004 W3C published the XML Inclusions (XInclude) 1.0 Recommendation (see W3C XINCLUDE). The functionality provided by the REFERENCE element in the first edition of ISO/IEC 21000 can be substantially provided by XInclude. XInclude also provides functionality that extends beyond that of the REFERENCE element. For example, the document author can provide a fallback in case the referenced elements are unable to be retrieved. Hence in this second edition the REFERENCE element has been removed and similar functionality is achieved using XInclude.

Some important differences between REFERENCE and XInclude to consider when using XInclude instead of REFERENCE include

— A REFERENCE element operates on its parent element, in that the content of the referenced element is included in to the content of the parent element of the REFERENCE element. Whereas an XInclude `include` element operates on itself, in that the included content replaces the XInclude `include` element.

— XInclude processing can result in addition of xml:base and xml:lang attributes to the top-level included elements.

NOTE This causes no problems for those elements for which a REFERENCE child element was allowed in the first edition (namely ANCHOR, COMPONENT, CONTAINER, DESCRIPTOR, ITEM, and ANNOTATION) since in this second edition these elements are allowed to have any attribute from an other namespace.

— XInclude processing does not include the attribute value inheritance of the REFERENCE element.

EXAMPLE 1 The following fragment is from a DIDL document named "otherdoc.xml".

```
<Item>
 <Component id="otherComp">
  <Descriptor>
   <Statement mimeType="text/plain">example 1</Statement>
  </Descriptor>
  <Resource mimeType="audio/mpeg" ref="audio.mp3"/>
 </Component>
</Item>
```

EXAMPLE 2 An ISO/IEC 21000-2 first edition DIDL document in the same location can use a REFERENCE element as follows.

```
<Item>
 <Component id="myComp">
  <Reference target="otherdoc.xml#otherComp"/>
 </Component>
</Item>
```

EXAMPLE 3 The result of resolving the REFERENCE for the above example is as follows.

```
<Item>
 <Component id="myComp">
  <Descriptor>
   <Statement mimeType="text/plain">example 1</Statement>
  </Descriptor>
  <Resource mimeType="audio/mpeg" ref="audio.mp3"/>
 </Component>
</Item>
```

EXAMPLE 4 An ISO/IEC 21000-2 second edition DIDL document can use XInclude as follows.

```
<Item>
 <include xmlns="http://www.w3.org/2001/XInclude"
       href="otherdoc.xml" xpointer="otherComp"/>
</Item>
```

EXAMPLE 5 The result of the XInclude processing for the above example is as follows.

```
<Item>
 <Component id="otherComp" xml:base="otherdoc.xml">
  <Descriptor>
   <Statement mimeType="text/plain">example 1</Statement>
  </Descriptor>
  <Resource mimeType="audio/mpeg" ref="audio.mp3"/>
 </Component>
</Item>
```

EXAMPLE 6 Another ISO/IEC 21000-2 first edition use of a REFERENCE element might be as follows.

```
<Item>
 <Component id="myComp">
  <Descriptor>
   <Statement mimeType="text/plain">example 6</Statement>
  </Descriptor>
  <Reference target="otherdoc.xml#otherComp"/>
 </Component>
</Item>
```

EXAMPLE 7 The result of resolving the REFERENCE for the above example is as follows.

```
<Item>
 <Component id="myComp">
  <Descriptor>
   <Statement mimeType="text/plain">example 6</Statement>
  </Descriptor>
  <Descriptor>
   <Statement mimeType="text/plain">example 1</Statement>
  </Descriptor>
  <Resource mimeType="audio/mpeg" ref="audio.mp3"/>
 </Component>
 </Component>
</Item>
```

EXAMPLE 8 An ISO/IEC 21000-2 second edition DIDL document can use XInclude as follows.

```
<Item>
 <Component id="myComp" xmlns:xi="http://www.w3.org/2001/XInclude">
  <Descriptor>
   <Statement mimeType="text/plain">example 6</Statement>
  </Descriptor>
  <xi:include href="otherdoc.xml" xpointer="element(otherComp/1)"/>
  <xi:include href="otherdoc.xml" xpointer="element(otherComp/2)"/>
 </Component>
</Item>
```

EXAMPLE 9 The result of the XInclude processing for the above example is as follows.

```
<Item>
 <Component id="myComp" xmlns:xi="http://www.w3.org/2001/XInclude">
  <Descriptor>
   <Statement mimeType="text/plain">example 6</Statement>
  </Descriptor>
  <Descriptor>
   <Statement mimeType="text/plain">example 1</Statement>
  </Descriptor>
  <Resource mimeType="audio/mpeg" ref="audio.mp3"/>
 </Component>
</Item>
```

B.8 Schema definitions

DID second edition introduces a DID Model abstract schema (see 8.2). This schema (uses W3C XMLSCHEMA and) defines DID Model abstract types for the entities of the DID Model. However it does not define the content model for any of these DID Model abstract types. That is, it does not explicitly define, within the DID Model abstract schema, the relationships amongst the DID Model abstract types. This DID Model abstract schema can serve as a base to derive schemas for any part of MPEG-21. If these derived schemas use the DID Model abstract types as heads of substitution groups, it can enable a level of interoperability among the schemas whereby elements of one schema can substitute for elements of another schema, provided that they are part of a substitution group with the same head.

DID second edition modifies the DIDL schema (see 8.3) to be based on the DID Model abstract schema. The DIDL schema defines DIDL concrete types based on the corresponding DID Model abstract types. Each DIDL concrete type is placed in a substitution group headed by the corresponding DID Model abstract type. The DIDL schema also defines the content model for these DIDL concrete types based on the DID Model. That is, it defines the DID Model based relationships amongst the DIDL concrete types. The content model for the DIDL concrete types is defined using the DID Model abstract types. This enables a DIDL element in a DIDL document to be substituted validly (according to the DIDL schema) by an element from another schema that also derives from the DID Model abstract schema and has the same substitution group head.

B.9 Converting a first edition DIDL document to a second edition DIDL document

This subclause lists one possible procedure that might be applicable for some applications for converting an ISO/IEC 21000-2 first edition DIDL document to an ISO/IEC 21000-2 second edition DIDL document.

a) Change any references to DIDL first edition namespace `urn:mpeg:mpeg21:2002:01-DIDL-NS` to the DIDL second edition namespace `urn:mpeg:mpeg21:2002:02-DIDL-NS`.

b) For each RESOURCE element in the document

   1) If the *resource* is defined by value (that is its data is included in the content of the RESOURCE element), and that data is base64 encoded, add an encoding attribute with value `base64`.

   2) If there is a LOCALPATH attribute present, replace it with an attribute from an application specific namespace.

c) For each STATEMENT element in the document

   1) If there is no MIMETYPE attribute present, add a MIMETYPE attribute with an appropriate value.

   2) If the *statement* is base64 encoded, add an ENCODING attribute with value `base64`.

d) For each ANCHOR element in the document

   1) If a FRAGMENT attribute is present

      i) Remove the FRAGMENT attribute

      ii) Add a FRAGMENT child element with a FRAGMENTED attribute with a value equivalent to the value of the removed FRAGMENT attribute.

e) For each CONDITION element in the document

   1) For each SELECTION element associated with the CONDITION

      i) If the parent CHOICE element of the SELECTION is not a child of an ITEM that is an ancestor of the CONDITION, move the CHOICE element such that it is the child of an ITEM that is an ancestor of the CONDITION.

f) For each REFERENCE element in the document

 1) If the REFERENCE element is the only child element of the parent element

  i) Replace the parent element with an XInclude `include` element that references the target element of the REFERENCE element.

 2) If the parent element contains other child elements (that is the REFERENCE element has siblings)

  i) Remove the REFERENCE element

  ii) For each child element of the target element of the REFERENCE element

   I) Add to the parent element (of the REFERENCE element) an XInclude `include` element that references the child element of the target element, at the valid location within the parent element

  iii) Copy the attributes and their values from the target element to the parent element (of the REFERENCE) if those attributes are not present on the parent element

Bibliography

[1] *Extensible Markup Language 1.0 (Second Edition)*, W3C Recommendation, 6 October 2000, http://www.w3.org/TR/2000/REC-xml-20001006

[2] *XML Schema Part 1: Structures Second Edition and Part 2: Datatypes Second Edition*, W3C Recommendation, 28 October 2004, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/

[3] *Canonical XML Version 1.0*, W3C Recommendation, 15 March 2001, http://www.w3.org/TR/2001/REC-xml-c14n-20010315

[4] ISO/IEC 14977:1996, *Information technology — Syntactic metalanguage — Extended BNF*

[5] *XML Pointer Language (XPointer) Version 1.0*, W3C Last Call Working Draft, 8 January 2001, http://www.w3.org/TR/2001/WD-xptr-20010108

[6] ISO/IEC 21000 (all parts), *Information technology — Multimedia framework (MPEG-21)*

[7] *XML Inclusions (XInclude) Version 1.0*, W3C Recommendation, 20 December 2004, http://www.w3.org/TR/2004/REC-xinclude-20041220/

[8] *Namespaces in XML, W3C Recommendation*, 14 November 1999, http://www.w3.org/TR/1999/REC-xml-names-19990114

ICS 35.040

Price based on 88 pages