SMPTE 429-9-2007

SMPTE STANDARD

D-Cinema Packaging — Asset Mapping and File Segmentation

Table of Contents

Page

Foreword 2
1 Scope 3
2 Conformance Notation 3
3 Normative References 3
4 Overview 4
4.1 Ingesting a DCP Volume (Informative) 5
4.2 XML File Structure 5
5 AssetMap Structure 6
5.1 Id 6
5.2 AnnotationText [optional] 6
5.3 Creator 7
5.4 VolumeCount 7
5.5 IssueDate 7
5.6 Issuer 7
5.7 AssetList 7
6 Asset Structure 7
6.1 Id 7
6.2 AnnotationText [optional] 8
6.3 PackingList [optional] 8
6.4 ChunkList 8
7 Chunk Structure 8
7.1 Path 8
7.2 VolumeIndex [optional] 8
7.3 Offsest [optional] 9
7.4 Length [optional] 9
8 VolumeIndex Structure 9
9 Media-Specific Constraints 9
9.1 Volume Definition 9
9.2 URL Scheme 9

Approved
June 25, 2007

9.3 URL Path 9
9.4 XML Encoding 10
9.5 Asset Map Location 10
9.6 VolumeIndex Location 10
9.7 Chunk Size 10
10 Asset Map Sample (Informative) 11
11 Volume Index Sample (Informative) 11
12 XML Schema 12
Annex A Basic Map Profile (Normative) 14
A.1 URL Scheme 14
A.2 URL Path 14
A.3 XML Encoding 14
A.4 Asset Map Location 14
A.5 VolumeIndex Location 14
A.6 Chunk Size 14
Annex B XML Diagram Legend (Informative) 15
B.1 Element Symbols 15
B.1.1 Examples 15
B.2 Model Symbols ("Compositors") 16
B.3 Types 16
B.4 Model Groups and References 17
Annex C Bibliography (Informative) 18

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in Part XIII of its Administrative Practices.

SMPTE 429-9 was prepared by Technology Committee DC28.

1 Scope

This document specifies a generic method for mapping a D-Cinema Package (DCP) onto one or more file storage volumes. Data structures are specified which provide for the mapping of D-Cinema asset identifier values onto paths within a particular file storage scheme (e.g., filesystem paths). Where required, assets may be split across multiple storage volumes to allow efficient use of media and the mapping of assets larger than a given storage volume's capacity. The Asset Map and Volume Index structures and the associated provisions detailed herein are intended to provide a framework for simplified mapping of a DCP onto a wide variety of file storage systems.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document. The keyword "reserved" indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword "forbidden" indicates "reserved" and in addition indicates that the provision will never be defined in the future.

3 Normative References

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

1. World Wide Web Consortium (W3C) (2004, February 4). Extensible Markup Language (XML) 1.0 (Third Edition).

2. World Wide Web Consortium (W3C) (2004, October 28). XML Schema Part 1: Structures (Second Edition).

3. World Wide Web Consortium (W3C) (2004, October 28). XML Schema Part 2: Datatypes (Second Edition).

4. Internet Engineering Task Force (IETF) (November 1996) RFC1738 — Uniform Resource Locators (URL)

5. Internet Engineering Task Force (IETF) (1996, November). RFC 2396 — Uniform Resource Identifiers (URI): Generic Syntax

6. Internet Engineering Task Force (IETF) (2005, July). RFC 4122 — A Universally Unique Identifier (UUID) URN Namespace.

7. SMPTE 429-8-2007, D-Cinema Packaging — Packing List

4 Overview

D-Cinema content is composed of a number of distinct assets such as Composition Playlists and Track Files. For delivery between systems, these assets are combined into logical D-Cinema Packages (DCP). A DCP is a single delivery unit defined by a Packing List document [SMPTE 429-8]. The Packing List enumerates all the assets included in the DCP, and provides information necessary for a traceable and error-free delivery.

The DCP and its Packing List are specified independently of any media or file storage scheme. The Packing List contains the UUID identifiers of the assets in the package, not the location of those assets on some volume of media. The Asset Map provides the link between the asset identifier UUIDs and the files containing the assets. This layered approach has two distinct advantages. First, it simplifies specification of the Packing List by removing dependence on past, current or future media formats and constraints. Second, it allows the DCP and its underlying assets to be created once and mapped onto various target media as circumstances require.

The Asset Map structure, depicted in Figure 1 below, provides a mapping from the Packing List asset Ids to actual file locations within the file storage scheme. An Asset Map may contain mappings for more than one DCP, but shall not contain a partial mapping for a particular DCP. A DCP shall not span multiple Asset Maps. The Asset Map provides special identification for Packing List assets so that they may be located without scanning the entire volume.

A DCP storage volume (a distinct container of files) shall contain exactly one Asset Map. In a multi-volume mapping, a single Asset Map shall contain the complete mapping for all volumes, and shall be repeated on each volume. A Volume Index structure (defined in Section 8 below) shall be used to identify each volume in a multi-volume set.

Assets may be divided into multiple segments to allow spanning across multiple volumes (segmentation). The segments are called “chunks”, and are stored in files, without additional structure. An Asset may be recreated by concatenating the constituent chunk files.

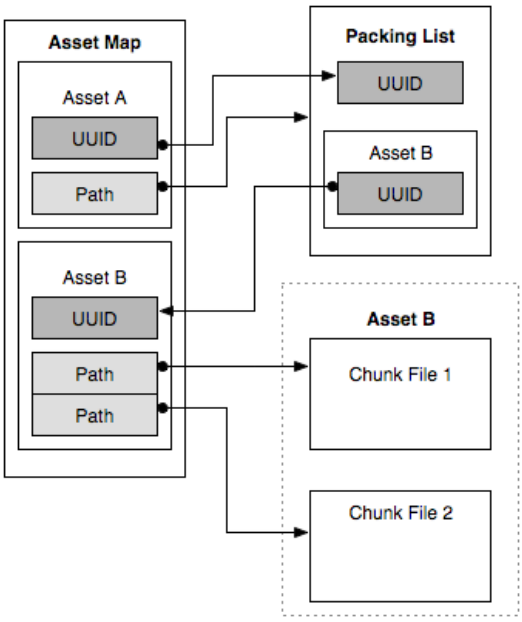


Figure 1 – Relationship between the Packing List and Asset Map structures

The definition of "volume" and the means of identifying the Asset Map on a volume shall be defined by a normative DCP Map Profile (see Section 9 "Media-Specific Constraints" below) for that media type. DCP Map Profiles shall normatively reference this document and shall follow the provisions of this document except where explicitly noted.

4.1 Ingesting a DCP Volume (Informative)

The process of reading a DCP into a D-Cinema system is termed *Ingest*. A system may provide a simple or elaborate ingest control interface - allowing coarse or fine-grained selection of items to be ingested — but the basic process of discovering the contents of a volume will always be the same.

As illustrated using pseudo-code in Figure 2 below, a DCP storage volume is ingested by first opening the Asset Map document on that volume. The means of identifying the Asset Map on the volume is defined by the normative DCP Map Profile (see Section 9) for that media type. The Asset Map is used to locate the Packing List(s) which detail the contents of the available DCP(s). Assets are chosen from the Packing List(s), and the selected assets are ingested by using the Asset Map to locate the chunks of data on the storage volume. The chunks are concatenated to restore the original file. (This example does not illustrate multi-volume ingest).

```
Ingest DCP volume:
  open the Asset Map

  for each Packing List in the Asset Map
    open the Packing List

    for each Asset in the Packing List
      if this Asset is wanted
        locate the corresponding Asset structure in the Asset Map

        for each chunk in the Asset structure
          read the chunk data
          write to destination

        test the message digest
```

Figure 2 – Example ingest process

4.2 XML File Structure

The structures defined in this document are represented using the Extensible Markup Language (XML) [XML 1.0], and specified using XML Schema [XML Schema Part 1: Structures] and [XML Schema Part 2: Datatypes]. This specification shall be associated with a unique XML namespace name [Namespaces in XML]. The namespace name shall be the string value "`http://www.smpte-ra.org/schemas/429-9/2007/AM`". This namespace name conveys both structural and semantic version information, but does not serve the purpose of a traditional version number field.

Table 1 lists the XML namespace names used in this specification. Namespace names are represented as Uniform Resource Identifier (URI) values [RFC 2396][1].

[1] Readers unfamiliar with URI values as XML namespace names should be aware that although a URI value begins with a "method" element ("http" in this case), the value is designed primarily to be a unique string and does not necessarily correspond to an actual on-line resource. Applications implementing this standard should not attempt to resolve URI values on-line.

Table 1 – XML Namespaces

Qualifier	*URI*
am	`http://www.smpte-ra.org/schemas/429-9/2007/AM`
xs	`http://www.w3.org/2001/XMLSchema`

The URIs found in Table 1 are normative. The namespace qualifier values (also called namespace prefixes in XML jargon) used in Table 1 and elsewhere in this document, namely "am" and "xs", are not normative. Specifically, they may be replaced in instance documents by any XML compliant namespace prefix. In other words, implementations shall expect any arbitrary XML compliant namespace prefix value that is associated with a URI from table 1.

5 AssetMap Structure

The `AssetMap` element is the top-level element of an Asset Map XML document. A single Asset Map document shall be present in a file on each DCP storage volume. The elements and structure of an AssetMap element are illustrated in Figure 3 and defined in the subsections that follow.

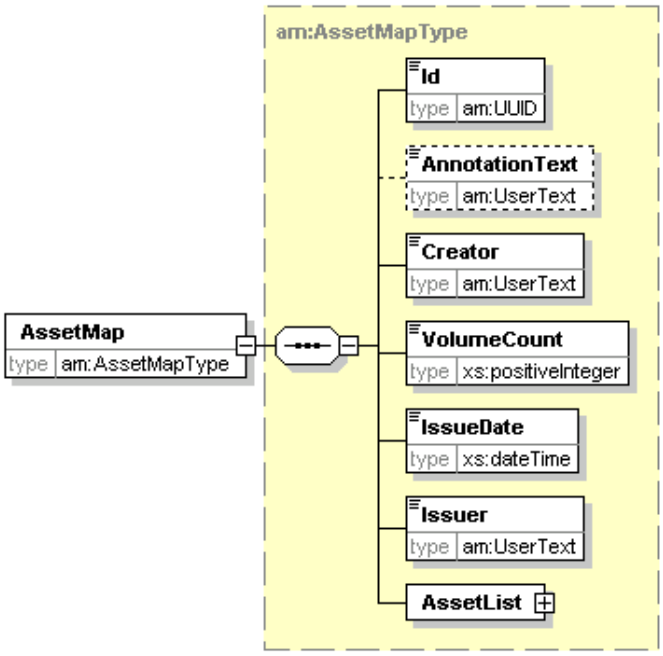


Figure 3 – AssetMap structure

5.1 Id

The `Id` element uniquely identifies the Asset Map structure. It is encoded as a `urn:UUID` [RFC 4122].

5.2 AnnotationText [optional]

The `AnnotationText` element is a free-form, human-readable annotation describing the Asset Map. It is meant strictly as a display hint to the user. The optional `language` attribute is an `xs:language` language code and indicates the language used. If the `language` attribute is not present, the default value en shall be used.

5.3 Creator

The `Creator` element is a free-form, human-readable annotation describing the system (hardware/software) that was used to create the Asset Map for distribution. It is meant strictly as a display hint to the user. The optional `language` attribute is an `xs:language` language code and indicates the language used. If the `language` attribute is not present, the default value `en` shall be used.

5.4 VolumeCount

The `VolumeCount` element indicates the total number of volumes that are referenced by this Asset Map. The first volume in a set shall be numbered 1 (one).

5.5 IssueDate

The `IssueDate` element indicates the time and date at which the Asset Map was issued. It may be displayed to the user.

5.6 Issuer

The `Issuer` element is a free-form, human-readable annotation describing the person or company who has created the Asset Map for distribution. It is meant strictly as a display hint to the user. The optional `language` attribute is an `xs:language` language code and indicates the language used. If the `language` attribute is not present, the default value `en` shall be used.

5.7 AssetList

The `AssetList` element contains a list of `Asset` elements. The structure of the `Asset` element is described in the following section.

6 Asset Structure

The `Asset` element is used to represent assets within the `AssetList` element of an `AssetMap`. The `Asset` element is of the type `AssetType`, illustrated in Figure 4 and defined in the subsections that follow.

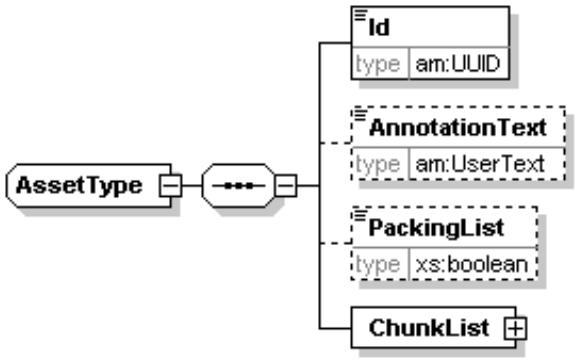


Figure 4 – Asset structure

6.1 Id

The `Id` element uniquely identifies the asset for management purposes. It is encoded as a `urn:UUID` [RFC 4122].

6.2 AnnotationText [optional]

The `AnnotationText` element is a free-form, human-readable annotation describing the asset. It is meant strictly as a display hint to the user. The optional `language` attribute is an `xs:language` language code and indicates the language used. If the `language` attribute is not present, the default value `en` shall be used.

6.3 PackingList [optional]

The `PackingList` element indicates whether the asset is a Packing List [SMPTE 429-8]. It is encoded as an `xs:boolean`. If the asset is a Packing List, the element shall be present and the value shall be `true`. The default value is `false`.

6.4 ChunkList

The `ChunkList` element contains an ordered list of asset `Chunk` elements. The structure of the `Chunk` element is described in the following section.

7 Chunk Structure

The `Chunk` element is used to represent chunks within the `ChunkList` element of an `Asset`. The `Chunk` element is of the type `ChunkType`, illustrated in Figure 5 and defined in the subsections that follow.

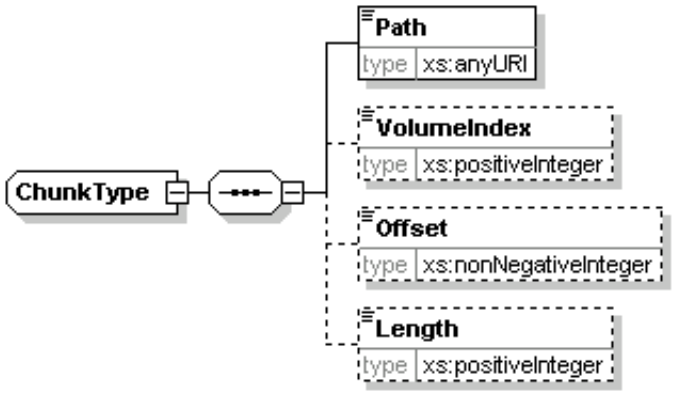


Figure 5 – Chunk structure

7.1 Path

The `Path` element indicates the complete path for the chunk, represented as a URL per [RFC 1738]. Its semantics and format are delivery-medium dependent. The value is encoded as an `xs:anyURI`. The `scheme` segment of the URL shall be defined by the defining DCP Map Profile.

NOTE – Annex A presents a basic Map Profile.

7.2 VolumeIndex [optional]

The `VolumeIndex` element indicates the index of the volume that contains the chunk. The first volume shall be Volume 1. If the `VolumeIndex` parameter is absent, the chunk shall belong to volume 1.

7.3 Offset [optional]

The `Offset` element indicates the offset from the start of the asset to the first byte of the asset segment referenced by this chunk. If the `Offset` parameter is absent, the chunk is assumed to start at the beginning of the asset.

7.4 Length [optional]

The `Length` element identifies the length, in bytes, of the chunk. If the `Length` parameter is absent, the length of the chunk shall be that of the asset as expressed by the respective Packing List.

8 VolumeIndex Structure

The `VolumeIndex` element is the top-level element of a Volume Index XML document. A single Volume Index document shall be present in a file on each DCP storage volume. The Volume Index is optional for a single volume DCP.

The `VolumeIndex` element encodes the numerical index of a particular volume in a DCP set. The element has a single child element, `Index`, which encodes the volume index number as an `xs:positiveInteger` value. An example is given in Section 9 below.

9 Media-Specific Constraints

The structures defined above do not alone provide sufficient specification for mapping a DCP onto a file storage system. Many details must be defined to form a complete, standardized mapping, including the format and access method of the storage system and the paths of the Asset Map and Volume Index files on an instance of that system.

A *DCP Map Profile* document specifies, or references specifications of, a file storage system and the electronic and/or logical means of retrieving files from it. A DCP Map Profile specifies the manner by which a DCP shall be retrieved from the specified file storage system, including any necessary constraints on the use of the file storage system and the structures in this document. A DCP Map Profile is distinguished from other file storage specifications by its normative reference to this document and its adherence to the provisions of the following sub-sections.

A basic, normative Map Profile for random-access filesystem is given in Annex A below.

9.1 Volume Definition

The Profile shall explicitly define the term “volume” in the context of the particular storage system being addressed. For example, a disk-based mapping may define a volume as a set of constraints on an existing file system and disk interface, while a network mapping might specify a host addressing method and file retrieval protocol.

9.2 URL Scheme

The Profile shall specify the URL scheme to be used for `Path` element values (e.g. “`file`”). The Profile shall be restricted to a single URL scheme.

9.3 URL Path

The Profile shall define any necessary constraints on URL paths, such as path length, directory depth and directory separator, character set, etc.

9.4 XML Encoding

The Profile should specify an XML encoding method (e.g. UTF-8) for the Asset Map and Volume Index structures.

9.5 Asset Map Location

The Profile shall specify the path of the Asset Map structure. Each volume shall contain exactly one Asset Map structure. Each volume in a multi-volume set shall contain the same Asset Map structure. It is recommended that a fixed path (i.e. not calculated) be used to locate the Asset Map structure.

The Asset Map structure shall be an XML document containing a valid `AssetMap` element per Sections 5-7 of this document.

9.6 VolumeIndex Location

The Profile shall specify the path of the Volume Index structure. Each volume in a multi-volume set shall contain exactly one distinct Volume Index structure. A single volume set may optionally contain a Volume Index structure. It is recommended that a fixed path (i.e. not calculated) be used to locate the Volume Index structure.

The Volume Index structure shall be an XML document containing a valid `VolumeIndex` element per Section 8 of this document. The contents of the `Index` element of the Volume Index structures in a multi-volume set shall be consecutive, with the first volume in a set having the index value of 1 (one).

9.7 Chunk Size

The Profile may define a maximum size for files, for example, to allow support for older file systems.

10 Asset Map Sample (Informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<am:AssetMap xmlns:am="http://www.smpte-ra.org/schemas/429-9/2007/AM">
   <am:Id>urn:uuid:585ae93a-1fc3-4908-b96c-839b0588d367</am:Id>
   <am:AnnotationText>Magic Packager 3.2</am:AnnotationText>
   <am:Creator>Asset Mapper 1.0</am:Creator>
   <am:VolumeCount>2</am:VolumeCount>
   <am:IssueDate>2001-12-17T09:30:47-00:00</am:IssueDate>
   <am:Issuer>Analog Labs</am:Issuer>
   <am:AssetList>
      <am:Asset>
         <am:Id>urn:uuid:b4ab6bd6-a8ba-4a9a-9d0a-00668792ae20</am:Id>
         <am:AnnotationText>Packing List for The Jazz Singer</am:AnnotationText>
         <am:PackingList>true</am:PackingList>
         <am:ChunkList>
            <am:Chunk>
               <am:Path>file:///JazzSingerEN-0003278645.pkl.xml</am:Path>
            </am:Chunk>
         </am:ChunkList>
      </am:Asset>
      <am:Asset>
         <am:Id>urn:uuid:ff3209c7-f27e-444f-9dce-330397a5dd42</am:Id>
         <am:AnnotationText>Picture Track File for Reel #1 for The Jazz
                           Singer</am:AnnotationText>
         <am:ChunkList>
            <am:Chunk>
               <am:Path>file:///JS-235-2K-R1.ptrk.mxf.aa</am:Path>
               <am:VolumeIndex>1</am:VolumeIndex>
               <am:Offset>0</am:Offset>
               <am:Length>8000000</am:Length>
            </am:Chunk>
            <am:Chunk>
               <am:Path>file:///JS-235-2K-R1.ptrk.mxf.ab</am:Path>
               <am:VolumeIndex>2</am:VolumeIndex>
               <am:Offset>8000001</am:Offset>
               <am:Length>16000000</am:Length>
            </am:Chunk>
         </am:ChunkList>
      </am:Asset>
   </am:AssetList>
</am:AssetMap>
```

11 Volume Index Sample (Informative)

```
<?xml version="1.0" encoding="UTF-8"?>
<am:VolumeIndex xmlns:am="http://www.smpte-ra.org/schemas/429-9/2007/AM">
  <am:Index>1</am:Index>
</am:VolumeIndex>
```

12 XML Schema

The XML Schema document presented in this section normatively defines the structure of the Asset Map and Volume Index using XSDL. While this schema is intended to faithfully represent the structure presented in the normative prose portions (Sections 4 through 8) of this document, conflicts in definition may occur. In the event of such a conflict, the normative prose shall be the authoritative expression of the standard.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.smpte-ra.org/schemas/429-9/2007/AM"
   xmlns:am="http://www.smpte-ra.org/schemas/429-9/2007/AM"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified" attributeFormDefault="unqualified">
   <xs:simpleType name="UUID">
      <xs:restriction base="xs:anyURI">
         <xs:pattern value="urn:uuid:[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-
                            [0-9a-fA-F]{12}"/>
      </xs:restriction>
   </xs:simpleType>

   <xs:complexType name="UserText">
      <xs:simpleContent>
         <xs:extension base="xs:string">
            <xs:attribute name="language" type="xs:language" use="optional" default="en"/>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>

   <xs:complexType name="ChunkType">
      <xs:sequence>
         <xs:element name="Path" type="xs:anyURI"/>
         <xs:element name="VolumeIndex" type="xs:positiveInteger" minOccurs="0"/>
         <xs:element name="Offset" type="xs:nonNegativeInteger" minOccurs="0"/>
         <xs:element name="Length" type="xs:positiveInteger" minOccurs="0"/>
      </xs:sequence>
   </xs:complexType>

   <xs:complexType name="AssetType">
      <xs:sequence>
         <xs:element name="Id" type="am:UUID"/>
         <xs:element name="AnnotationText" type="am:UserText" minOccurs="0"/>
         <xs:element name="PackingList" type="xs:boolean" minOccurs="0"/>
         <xs:element name="ChunkList">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Chunk" type="am:ChunkType" maxOccurs="unbounded"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
 <!-- continued next page... -->
```

```
<!-- Asset Map schema continued -->
   <xs:complexType name="AssetMapType">
      <xs:sequence>
         <xs:element name="Id" type="am:UUID"/>
         <xs:element name="AnnotationText" type="am:UserText" minOccurs="0"/>
         <xs:element name="Creator" type="am:UserText"/>
         <xs:element name="VolumeCount" type="xs:positiveInteger"/>
         <xs:element name="IssueDate" type="xs:dateTime"/>
         <xs:element name="Issuer" type="am:UserText"/>
         <xs:element name="AssetList">
            <xs:complexType>
               <xs:sequence>
                  <xs:element name="Asset" type="am:AssetType" maxOccurs="unbounded"/>
               </xs:sequence>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>

   <xs:element name="AssetMap" type="am:AssetMapType"/>

   <xs:complexType name="VolumeIndexType">
      <xs:sequence>
         <xs:element name="Index" type="xs:positiveInteger"/>
      </xs:sequence>
   </xs:complexType>

   <xs:element name="VolumeIndex" type="am:VolumeIndexType"/>
</xs:schema>
```

Annex A (Normative)
Basic Map Profile

This annex defines a Map Profile which provides a DCP mapping onto any hierarchical, random-access filesystem which supports long path names and files larger than 4GB. The profile supports spanning a package across multiple volumes, but does not support segmentation (chunking).

A.1 URL Scheme

The URL scheme to be used for AssetMap `Path` element values shall be "`file`". The hostname segment of these URL values shall be null, thus all URL values shall begin with the string "file://". No other URL scheme shall be used.

A.2 URL Path

The path element of the URL shall consist of 8-bit ASCII characters from the set a-z, A-Z, 0-9, "-" (dash), "_" (underscore) and "." (period). The path shall be optionally divided into segments using the "/" (slash) character. No segment shall have more than 128 characters, and no path value shall exceed 1024 characters in length. A path value shall have no more than 10 segments. The path value shall preserve case (the path and the filename on the filesystem shall have identical case). No two paths in an Asset Map shall have identical value, regardless of case.

The Path element of the URL shall not begin with a '/' (slash) character, i.e. all paths shall be relative to the directory which contains the Asset Map file.

A.3 XML Encoding

The Asset Map and Volume Index structures shall be encoded using the UTF-8 encoding [XML 1.0].

A.4 Asset Map Location

The Asset Map structure shall be in a file named "`ASSETMAP.xml`", located in the root directory of the filesystem. When spanning multiple volumes, each volume shall contain exactly one Asset Map structure, identical to that on each of the other volumes in the set.

The Asset Map structure shall be an XML document containing a valid `AssetMap` element per Sections 5-7 of this document.

A.5 VolumeIndex Location

The Volume Index structure shall be in a file named "`VOLINDEX.xml`", located in the root directory of the filesystem. When spanning multiple volumes, each volume shall contain exactly one distinct Volume Index structure. A single volume set shall contain a Volume Index structure.

The Volume Index structure shall be an XML document containing a valid `VolumeIndex` element per Section 8 of this document. The contents of the `Index` element of the Volume Index structures in a multi-volume set shall be consecutive, with the first volume in a set having the index value of 1 (one).

A.6 Chunk Size

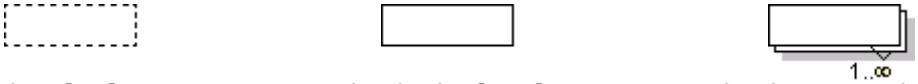
The maximum size for any file shall be limited only by the filesystem. Assets shall not be segmented (chunked) under this profile. The Asset Map will thus contain exactly one `Chunk` element inside each `Asset` element's `ChunkList`.

Annex B (Informative)
XML Diagram Legend

The following provides a legend for notation used in diagrams depicting XML structures.

B.1 Element symbols

In the schema design diagrams presented above in this document, only the elements are drawn. Attributes are not visible. The cardinality of the element (0..1, 1 exactly, 0..n, 1..n) is indicated by the border of the elements. Optional elements are drawn with a dashed line, required elements with a solid line. A maximum occurrence greater one is indicated by a double border.



Optional element
Min. occurrence = 0,
Max. occurrence = 1

Required single element
Min. occurrence = 1,
Max. occurrence = 1

Required repeated element
Min. occurrence = 1,
Max. occurrence = unbounded

The content model of elements is symbolized on the left and right side of the element boxes. The left side indicates whether the element contains a simple type (text, numbers, dates, etc.) or a complex type (further elements). The right side of the element symbol indicates whether it contains child elements or not:

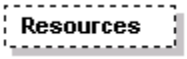
simple content

complex content

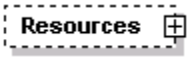
complex content
with child elements

no element content
(simple type, attributes
only, or empty element)

B.1.1 Examples



Optional single element without child elements. Minimum Occurrence = 0, Maximum Occurrence = 1, content = complex.



As above, but with child elements. The "plus" at the right side indicates the presence of one or more undisplayed child elements.

Mandatory single element. Minimum Occurrence = 1, Maximum Occurrence = 1, content = complex, no child elements (i.e. this denotes an *empty element*). The gray or green text below the element displays the xml-schema annotation associated with the element.

Mandatory multiple element containing child elements (content = complex). This element must occur at least once (Minimum Occurrence = 1) and may occur as often as desired (Maximum Occurrence = unbounded).

Mandatory single element containing simple content (e.g. text) or mixed complex content (e.g. text with xhtml markup). Minimum Occurrence = 1, Maximum Occurrence = 1, type = xsd:string (for example), content = simple. The three lines in the upper left corner are used for both text and numeric content.

B.2 Model symbols ("compositors")

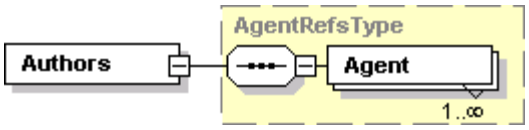
A sequence of elements. The elements must appear exactly in the sequence in which they appear in the schema diagram.

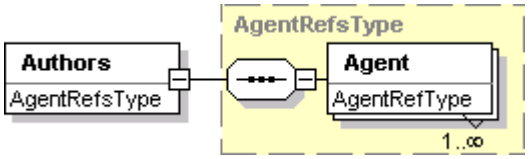
A choice of elements. Only a single element from those in the choice may appear at this position.

The "all" model, in which the sequence of elements is not fixed.

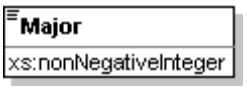
B.3 Types

If an element refers to a complex global type, the type is shown with a border.





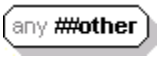
The type names of simple types are shown as well:



B.4 Model groups and references

An *element group* is a named container with one or several elements. The group of elements can be reused at multiple places in the schema. Model groups are invisible in the instance document. Model groups have been used sparingly since they do not map to a feature in object-oriented programming languages (unless they support multiple inheritance).

Important note in reading the diagrams for model groups: If the model group symbol is drawn with simple lines (i.e. not dashed), this does not imply that the elements in the model group are required. The optionality of the group depends on the optionality of elements contained in the model group. (Model groups can be made optional, e.g. to make a model group with required elements optional in some cases, but this has not been used.)



The *"any"* group is a special kind of model group. It is a placeholder for elements not defined in the schema. The "any" element defines points where the schema can be extended. After the "Any" keyword the namespace from which the elements may come is defined, for example, "##other" specifies that the extension elements may come from any namespace, except from the current schema namespace.

Element references are indicated through a link arrow in the lower left corner. They are similar to references to model groups within a schema, but instead of refining the model group, they directly refer to a single global element. The global element can then be reused in multiple places.

Annex C (Informative)
Bibliography

SMPTE 429-7-2006, D-Cinema Packaging — Composition Playlist

SMPTE 429-8-2007, D-Cinema Packaging — Packing List

Internet Engineering Task Force (IETF) (1994, December). RFC 1738 — Uniform Resource Locators (URL). http://www.ietf.org/rfc/rfc3986.txt

World Wide Web Consortium (W3C) — RDDL — Resource Directory Description Language J. Border and T. Bray 2002. http://www.rddl.org/

World Wide Web Consortium (W3C) — Namespaces in XML, http://www.w3.org/TR/REC-xml-names/

World Wide Web Consortium (W3C) — QA Framework: Specification Guidelines, Formal Languages, http://www.w3.org/TR/2004/WD-qaframe-spec-20041122/

World Wide Web Consortium (W3C) — XML Schema Primer, http://www.w3.org/