SMPTE ST 2001-2:2014

SMPTE STANDARD

XML Representation of SMPTE Registered Data (Reg-XML) — AAF and MXF Data

Page 1 of 22 pages

Table of Contents

Page

3 Barker Avenue, White Plains, NY 10601
(914) 761-1100

Approved
July 14, 2014

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 its Standards Operations Manual.

SMPTE ST 2001-2 was prepared by Technology Committee 31FS.

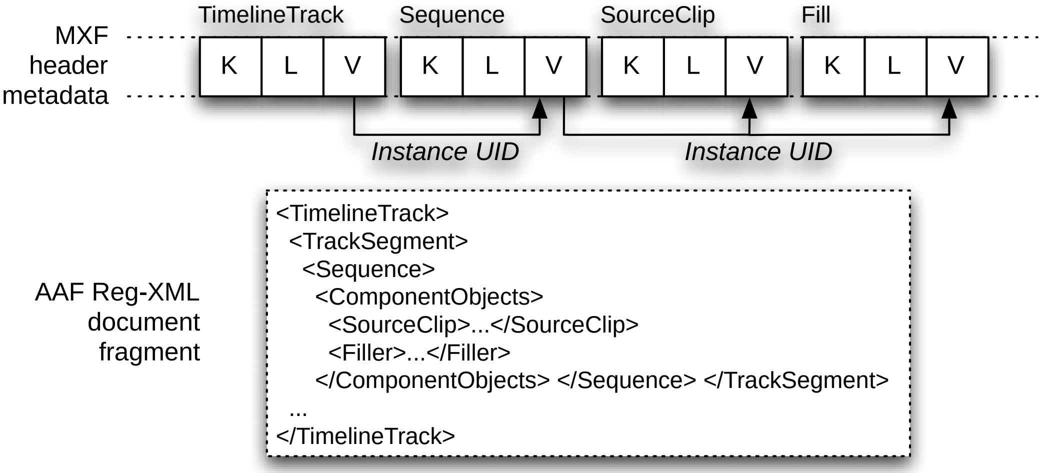
Intellectual Property

At the time of publication no notice had been received by SMPTE claiming patent rights essential to the implementation of this Engineering Document. However, attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. SMPTE shall not be held responsible for identifying any or all such patent rights.

Introduction

This section is entirely informative and does not form an integral part of this Engineering Document.

An *AAF Reg-XML file group* consists of a SMPTE ST 2001-1 *AAF Reg-XML document* and associated stream files that conform to the AAF baseline data model provided by this document. An AAF Reg-XML file group is capable of carrying header metadata and essence streams that can also be carried in MXF and AAF files.



```
<TimelineTrack>
  <TrackSegment>
    <Sequence>
      <ComponentObjects>
        <SourceClip>...</SourceClip>
        <Filler>...</Filler>
      </ComponentObjects> </Sequence> </TrackSegment>
  ...
</TimelineTrack>
```

Figure 1 – Example equivalence of KLV and XML representations

AAF data has a hierarchical structure and is commonly encoded in either MXF or AAF files. As illustrated in Figure 1, MXF files are encoded as a sequence of KLV packets, where the hierarchical data is flattened out into the file by the use of a referencing scheme. An AAF Reg-XML file group is an alternative encoding, with metadata items encoded as elements in an XML document, and where a nesting of XML elements is used to represent the hierarchy of the AAF data. Irrespective of its encoding format, the AAF instance data is the same.

This document provides both the baseline dictionary required to map AAF instance data to an AAF Reg-XML file group and the rules to do so. For developers, informative annexes provide an XML schema for the validation of AAF Reg-XML documents and guidelines for starting with one format of AAF data and encoding it as another. For example, guidelines for how to start out with an MXF file and end up with an AAF Reg-XML file group.

1 Scope

This document specifies:

- a Reg-XML baseline data model for AAF data (Annex B);
- rules for mapping AAF data to AAF Reg-XML file groups (Section 6).

An XSD Schema that can be used to validate AAF data encoded as an AAF Reg-XML document is provided in Annex C. This schema is the result of applying of the model mapping rules of SMPTE ST 2001-1.

Further annexes to this document provide informative guidelines for a developer, describing the process of:

1. starting with a MXF file and encoding it as an AAF Reg-XML file group (Annex D, Section D.2);
2. identifying suitable AAF Reg-XML file groups and encoding them as an MXF file (Annex D, Section D.3);
3. starting with an AAF file and encoding it as an AAF Reg-XML file group (Annex E).

Due to the zero-divergence approach applied at the time, the MXF data model is within the AAF baseline data model. Also, the harmonization of the registered names for data model items enables consistent representation of the two data models as a single Reg-XML data model.

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.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

Unless otherwise specified, the order of of the types of normative information in this document shall be as follows: Normative prose shall be the authoritative definition; Tables shall be next; followed by formal languages; then figures; and then any other language forms.

3 Normative References

The following standards contain provisions that, 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 recommended practice are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

SMPTE ST 2001-1:2013, XML Representation of SMPTE Registered Data (Reg-XML) — Mapping Rules

4 Definition of Acronyms, Terms and Notation

4.1 Acronyms and Terms

AAF: Advanced Authoring Format

Namespace: XML Namespace

Reg-XML file group: A collection containing a Reg-XML document and the stream files that provide the value of stream-type properties.

URN: Uniform Resource Name

UUID: Universally Unique IDentifier

XML: eXtensible Markup Language

XSD: XML Schema Definition – an XML document containing a schema expressed in XSDL

5 AAF Baseline Meta Dictionary (Informative)

5.1 Meta Dictionary Representation

The normative representation for the *AAF baseline meta dictionary* is provided in Annex B. This document element is generated from an authoritative source for the AAF data model provided by the Advanced Media Workflow Association. The same source is used for equivalent definitions contained in the SMPTE registers. In general, the two representations align.

According to the precedence rule of Annex B, in the event of a conflict between the definitions of the AAF baseline meta dictionary and another definition, such as one in the SMPTE registers, the model defined in this document has to take precedence.

5.2 Meta Dictionary and SMPTE Registers

This document contains a complete statement of the AAF and MXF baseline meta dictionary in Annex B. Any AAF Reg-XML file group encoding of AAF instance data, including instance data decoded from an MXF file, is with reference to the baseline provided in this document. For information only, this section describes the implicit relationship between the SMPTE registers and the baseline meta dictionary

The relationship between the definitions of the AAF baseline meta dictionary and the SMPTE registers is illustrated in Figure 2.

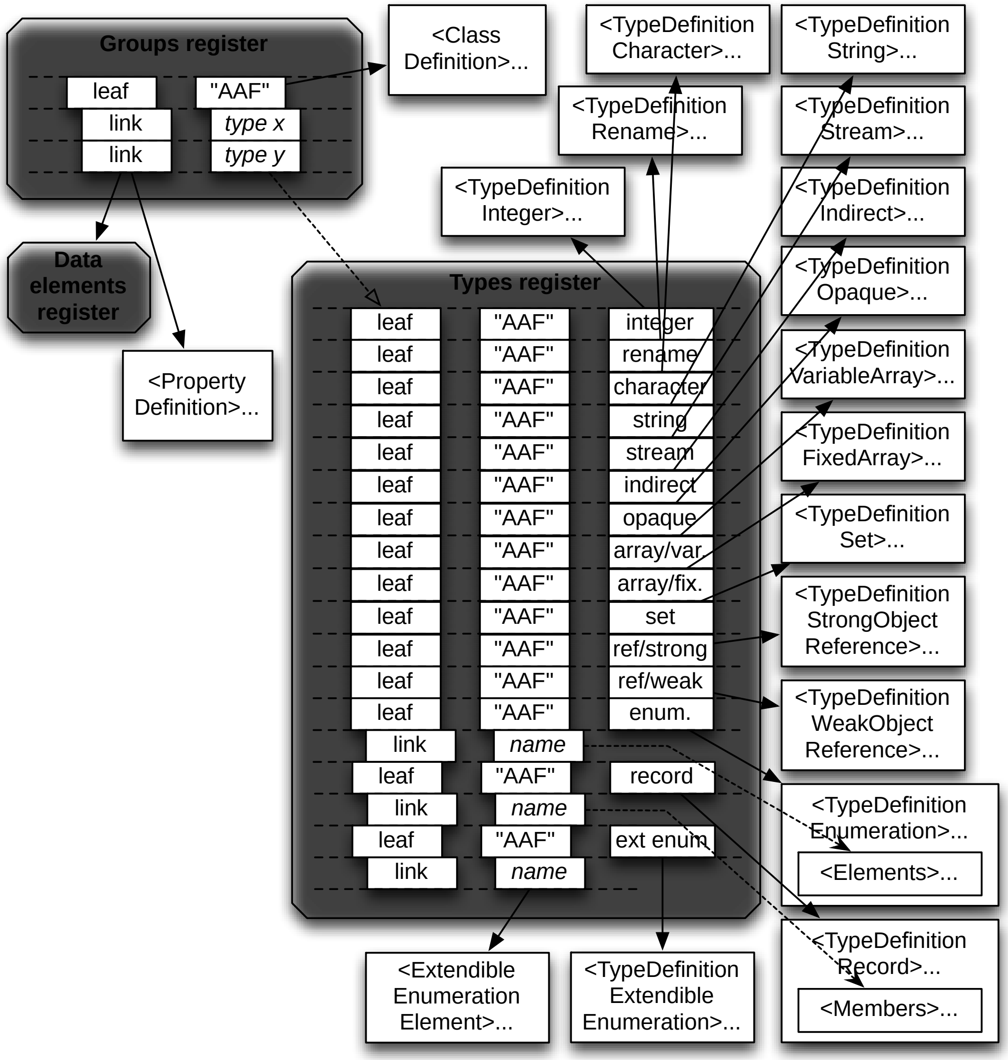

Groups register
leaf
"AAF"
link
type x
link
type y
<Class Definition>...
Data elements register
<Property Definition>...
Types register
leaf
"AAF"
integer
rename
character
string
stream
indirect
opaque
array/var
array/fix.
set
ref/strong
ref/weak
enum.
link
name
record
ext enum
<TypeDefinition Character>...
<TypeDefinition String>...
<TypeDefinition Rename>...
<TypeDefinition Stream>...
<TypeDefinition Integer>...
<TypeDefinition Indirect>...
<TypeDefinition Opaque>...
<TypeDefinition VariableArray>...
<TypeDefinition FixedArray>...
<TypeDefinition Set>...
<TypeDefinition StrongObject Reference>...
<TypeDefinition WeakObject Reference>...
<TypeDefinition Enumeration>...
<Elements>...
<TypeDefinition Record>...
<Members>...
<Extendible Enumeration Element>...
<TypeDefinition Extendible Enumeration>...

Figure 2 – AAF baseline data model and SMPTE registers

As a general rule, meta definitions are included in the baseline as follows:

1. All class definitions in the baseline data model are registered as groups in the SMPTE Groups Register, with “AAF” listed as an application. Only classes that extend “InterchangeObject” are included in the baseline dictionary. The meta dictionary and meta definition classes are not included in the baseline model as these are normatively specified by the Reg-XML data model of ST 2001-1 and are represented accordingly in Reg-XML.

2. All property definitions in the baseline data model are registered as members of groups in the SMPTE Groups Register, with “AAF” listed as an application. According to the defined structure of the registers, these *link* entries refer to leaf entries in the SMPTE Elements Register. The criterion for including a property definition in the AAF baseline is that it is a member of one of the included class definitions.

3. All type definitions in the baseline data model are registered as types in the SMPTE Types Register, with “AAF” listed as an application. The type definitions that are included in the AAF baseline are those that are required to represent values of included property definitions.

6 Mapping Rules for AAF Data

This section specifies the normative rules for mapping AAF instance data to a Reg-XML encoding of that data.

6.1 AAF Reg-XML File Group

An *AAF Reg-XML file group* shall consist of a single Reg-XML document, known as the *AAF Reg-XML document*, and external stream data files.

The single AAF Reg-XML document shall be used to represent metadata items of AAF data.

A separate and non-XML stream file shall be used to represent the value of each property of stream type in the AAF data.

6.2 AAF Reg-XML Document

The baseline meta dictionary of the AAF Reg-XML document shall be the baseline meta dictionary of Annex B of this document. The data mapping rules of SMPTE ST 2001-1 shall be applied to encode the AAF data.

Note 1: According to the rules of SMPTE ST 2001-1, the root element of the AAF Reg-XML document will be "AAF", defined in the default namespace:

http://www.smpte-ra.org/schemas/2001-2/2013/aaf

The root object of the data section of the AAF Reg-XML document shall be the single *Preface* object contained in the AAF data.

Note 2: Valid AAF data contains only one preface object. Applying the Reg-XML part 1 rules to this element will result in a complete serialization of all of the metadata items of the AAF instance data.

6.3 Extension Meta Dictionaries

The baseline meta dictionary shall not be included in the AAF Reg-XML document.

Any data instance not fully defined by definitions in the baseline shall be defined by additional meta definitions contained in an extension scheme. Full definition shall include class definitions, property definitions and type definitions sufficient to represent a value of the data instance. In a way that ensures all data instances are fully defined for an AAF Reg-XML document, all extension schemes containing meta definitions shall be included in the AAF Reg-XML document according to the rules of SMPTE ST 2001-1.

The encoding of any meta dictionary and meta definition in an AAF Reg-XML document shall use the XML schema provided in Annex A of SMPTE ST 2001-1.

References made from extension meta definitions to other extension meta definitions shall either be resolvable within the same scheme, with reference to a definition in the AAF baseline or with reference to another extension scheme included in the AAF Reg-XML document.

An extension scheme should be included in its entirety.

For example, a SMPTE ST 377-1 MXF Application Metadata Plugin Scheme is defined in a standalone SMPTE document. This can be represented as an extension meta dictionary that is included into any AAF Reg-XML document that uses it.

Note: Artifacts of other encoding mechanisms that are not explicitly defined in the AAF baseline model of Annex B, such as SMPTE ST 377-1 MXF partitions or AAF structured storage, are not encoded in the AAF Reg-XML document.

Annex A Bibliography (Informative)

Note: All references in this document to other SMPTE documents use the current numbering style (e.g. SMPTE ST 336:2007) although, during a transitional phase, the document as published (printed or PDF) may bear an older designation (such as SMPTE 336M-2007). Documents with the same root number (e.g. 336) and publication year (e.g. 2007) are functionally identical.

Advanced Media Workflow Association — MS-01 AAF Object Specification v1.1. Available at http://www.amwa.tv/downloads/specifications/aafobjectspec-v1.1.pdf

Advanced Media Workflow Association — MS-02 AAF Structured Storage Mapping. Available at: http://www.amwa.tv/projects/MS-02.shtml

Advanced Media Workflow Association — MS-03 AAF Structured Storage Specification. Available at: http://www.amwa.tv/projects/MS-03.shtml

SMPTE ST 335:2012, Metadata Element Dictionary Structure

SMPTE ST 336:2007, Data Encoding Protocol Using Key-Length-Value

SMPTE ST 377-1:2011 – Material Exchange Format (MXF) — File Format Specification

Amendment 1:2012 to SMPTE ST 377-1:2011

Amendment 2:2012 to SMPTE ST 377-1:2011

SMPTE ST 379-1:2009, Material Exchange Format (MXF) — MXF Generic Container

SMPTE ST 395:2014, Metadata Groups Register

SMPTE ST 400:2012, SMPTE Labels Structure

SMPTE ST 2003:2012, Types Dictionary Structure

Annex B AAF Baseline Meta Dictionary (Normative)

An XML file representing the baseline meta dictionary for AAF data is available as document element SMPTE ST 2001-2a. This baseline meta dictionary shall be used for all AAF Reg-XML file groups representing AAF data.

The definitions in this AAF baseline meta dictionary shall be normative. For the avoidance of doubt, a definition in this AAF baseline meta dictionary shall take precedence over any conflicting definition.

Note: It is intended that the exposition of the AAF data model of this annex and the relevant SMPTE registers be kept in synchronization.

Annex C AAF and MXF XML Schema (Informative)

An XML schema generated by applying the Reg-XML model mapping rules to the MXF and AAF baseline meta dictionary is provided in document element SMPTE ST 2001-2b. The schema can be used to validate AAF Reg-XML documents that do not contain any extension data.

To create a schema that validates a document with a known set of extension data, the model mapping rules of Reg-XML part 1 can be applied to a combination of the AAF baseline meta dictionary and the extension meta dictionaries. This will create a new XML schema with additional elements that can be used to represent a mix of baseline and extension metadata.

Annex D Guidelines: MXF Files and AAF Reg-XML File Groups (Informative)

D.1 Mapping Between KLV Local Sets and Reg-XML Document Fragments

SMPTE ST 377-1 specifies that MXF files be encoded as a sequence of SMPTE ST 336 KLV packets. This section illustrates the relationship between header metadata values that are encoded as MXF local sets and equivalent values encoded as fragments of AAF Reg-XML documents. The relationship is established through the AAF baseline data model provided in Annex B. Guidelines describing the mapping of MXF header metadata to AAF Reg-XML documents are provided in Section D.2.2.

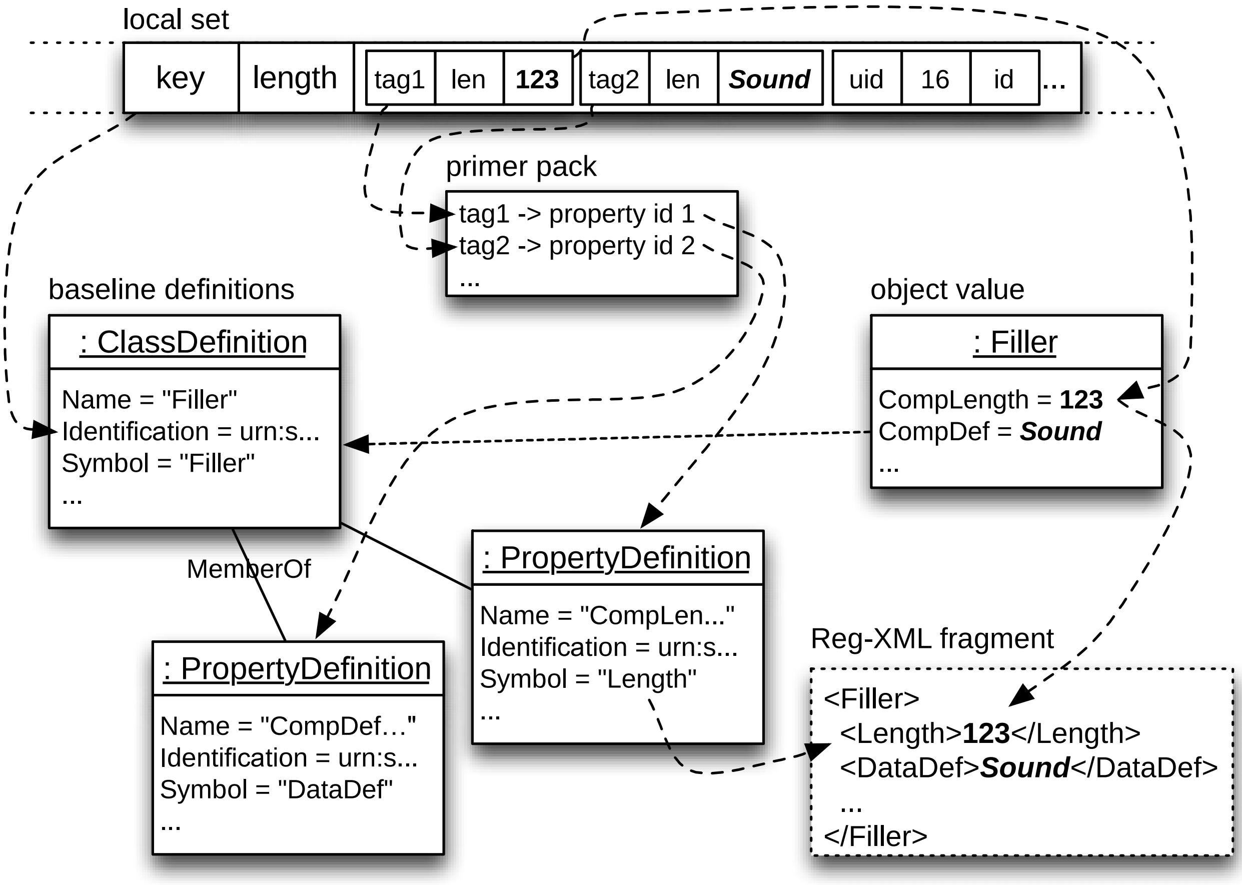


Figure D.1 – KLV local set to XML fragment

Each local set is decoded to AAF instance data according to the rules of SMPTE ST 377-1 and, with reference to the AAF baseline meta dictionary, encoded into an XML representation using the data mapping rules of SMPTE ST 2001-1. For a single MXF local set instance of class *Filler*, a decode of a local set and encode as an XML fragment is illustrated in Figure D.1 and described below:

1. The key of the local set identifies the class of data that is encoded by the set. For local sets defined as classes in SMPTE ST 377-1, this can be found as the identifier of a class definition in the AAF baseline meta dictionary. The symbol field of the class definition is used as the element name of the XML fragment.

2. A KLV packet with a 2-byte key – known as a *tag* – begins the encoding of each property value of the local set. Each property value, apart from the Instance UID ("uid" in Figure D.1), is encoded in XML as a child element of the class to which it belongs. Using the lookup table provided in the primer pack at the start of the header metadata, the full 16-byte identifier for the property is determined from the tag. This 16-byte identifier corresponds to a property definition in a meta dictionary. The symbol field of the property definition provides the XML element name used for the property in the XML representation, with the value of the property encoded according to the type-specific Reg-XML rules.

D.2 Decode MXF Files, Encode as AAF Reg-XML File Groups

D.2.1 Selecting Header Metadata

A single MXF file is encoded to exactly one AAF Reg-XML file group.

An MXF file contains at least one copy of header metadata. Only one copy of header metadata contained in an MXF file is selected, decoded and and then encoded in an AAF Reg-XML document.

In making the selection, a copy of header metadata from a closed and complete partition is chosen over any other copy of metadata. Where header metadata is carried in a footer partition that is closed and complete, it is best to select this copy over any other.

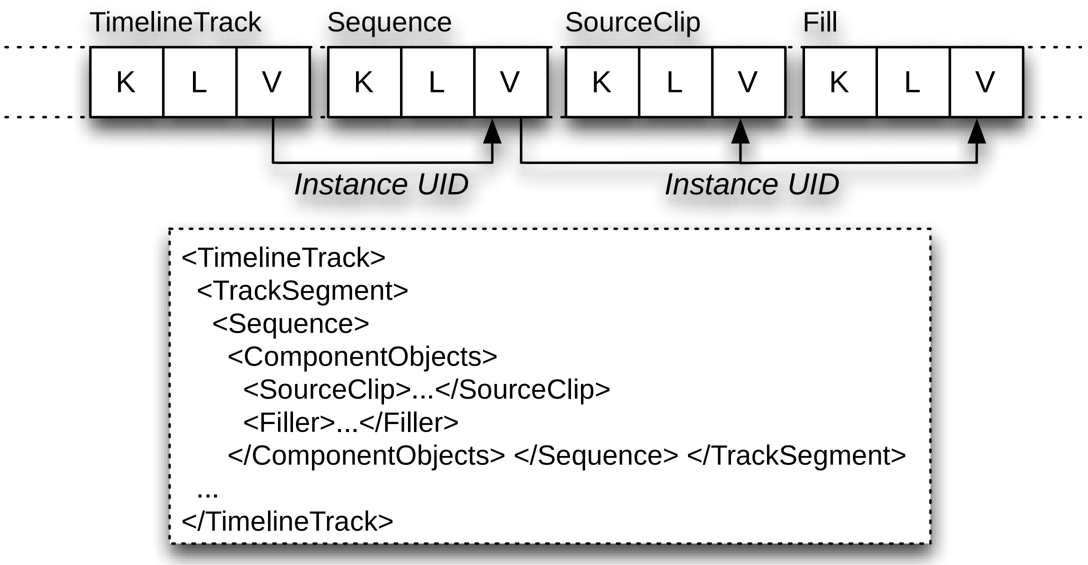
An AAF Reg-XML file group of MXF data does not have explicit support for growing files.

D.2.2 Header Metadata

A Reg-XML document is created according to data mapping rules defined in Section 6. The *root element name* of the document is "AAF".

The *root object* of the document is the single preface instance contained in the selected header metadata set, encoded as XML according to data mapping rule 3 of SMPTE ST 2001-1.

The hierarchical structure of the baseline data model will cause all valid encodings of the MXF data model to be encoded to XML as child elements of the preface element. Figure D.2 illustrates how the hierarchical relationships encoded as Instance UID strong references between the MXF local sets in the context of the same header metadata are encoded to XML parent-child relationships. The order in which the local sets is stored in the MXF header metadata is not specified, so it is sometimes necessary to read all local sets before references can be resolved so as to create the hierarchy of elements in the XML document.



```
<TimelineTrack>
 <TrackSegment>
  <Sequence>
   <ComponentObjects>
    <SourceClip>...</SourceClip>
    <Filler>...</Filler>
   </ComponentObjects> </Sequence> </TrackSegment>
 ...
</TimelineTrack>
```

Figure D.2 – Hierarchical equivalence of KLV and XML representations

The structural, descriptive and application-specific metadata encoded as MXF local sets within the selected header metadata are encoded in XML according to the following rules:

1. KLV fill items are skipped.

2. The key of the local set is used to find the corresponding class definition, with the definition's symbol name providing the XML element name, as follows:

   a. If the key equals the identifier of a class definition in the AAF baseline, the local set is encoded as an XML element with its property values encoded as child elements as described in rule 3.

   b. If the key in the local set identifies a meta dictionary or meta definition, the MXF local set defines an extension and is encoded in an extension meta dictionary as described in Section D.2.3.

   c. If the key corresponds to the identifier of a class definition that is encoded internally within the same file and header metadata, the MXF local set is encoded as an XML element with its property values encoded as child elements as described in rule 3.

   d. If the key corresponds to the identifier of a class definition defined in an application metadata scheme or descriptive metadata scheme defined externally and referenced from the preface or partition pack, the MXF local set is encoded if and only if the corresponding meta definitions are included as an extension meta dictionary in the same AAF Reg-XML document. Where the local set is encoded, it is as an XML element with its property values encoded as child elements as described in rule 3.

   e. Otherwise, the key does not correspond to a known class definition and is treated as dark metadata. Reg-XML does not provide a means to carry undefined metadata, and hence dark metadata, in AAF Reg-XML documents.

3. For each property of a MXF local set with a known class definition and parent XML element, the local tag for the property is mapped to its full 16-byte identifier via the header metadata's primer pack. This property identifier is used to find a corresponding property definition. This definition provides a symbol to be used as an XML element name and a property type that links to a Reg-XML rule for encoding the property value. To find the property definition:

    a. If the property identifier equals the identifier of a property definition in the AAF baseline:

        i. If the property definition is a member of the associated class definition, the property value is encoded as an XML element with its value encoded in XML according to the type-specific rules defined in SMPTE ST 2001-1.

        ii. Otherwise, the type of the property is not defined and the property value cannot be encoded as XML as no rule for encoding the value is known.

    b. If the property identifier corresponds to the identifier of a property definition that is included in an extension scheme in the AAF Reg-XML document:

        i. If the property definition is a member of the associated class definition, the property value is encoded as an XML element with its value encoded in XML according to the type-specific rules defined in SMPTE ST 2001-1.

        ii. Otherwise, the type of the property is not defined and the property value cannot be encoded as XML as no rule for encoding the value is known.

    c. If the property definition is not known then nor is its type. The property value cannot be encoded as XML as it is not possible to determine a type-specific mapping rule to apply.

D.2.3 Extension Meta Dictionaries

Meta definitions may be contained as objects encoded as MXF local set in the selected header metadata set. These meta definitions are considered as dark metadata by many MXF processors. Many existing SMPTE ST 390 MXF OP-Atom files contain complete copies of their meta dictionary and future KLV Extension Syntax (KXS) compliant files will contain extension-only meta dictionaries.

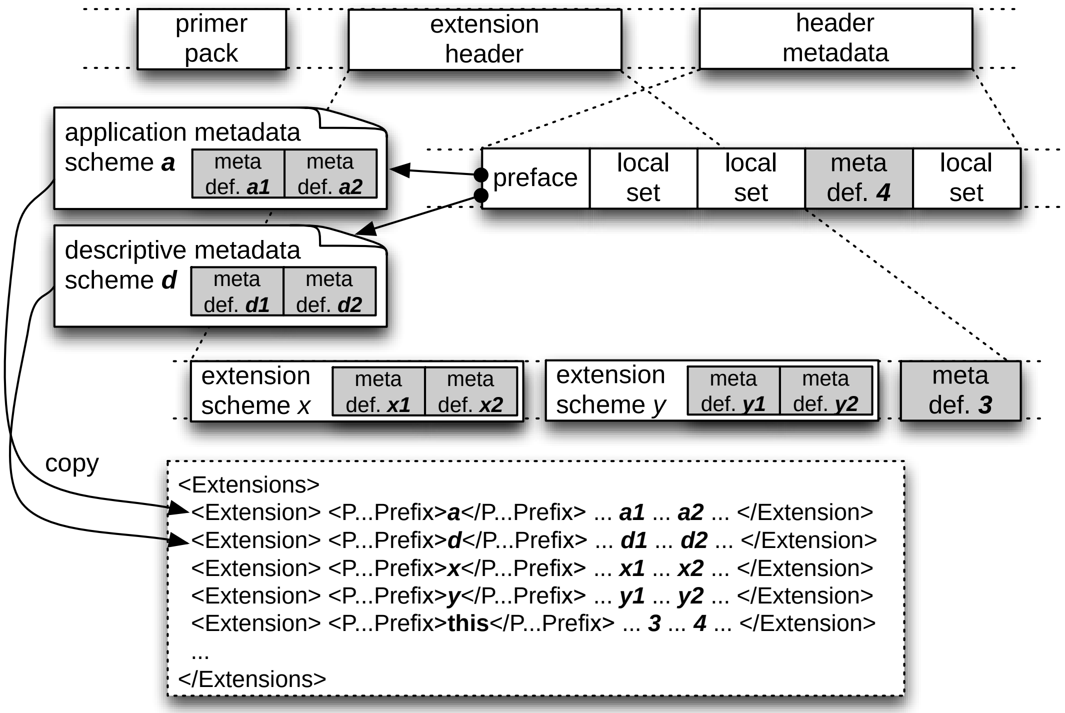

primer pack
extension header
header metadata
application metadata scheme *a*
meta def. *a1*
meta def. *a2*
preface
local set
local set
meta def. *4*
local set
descriptive metadata scheme *d*
meta def. *d1*
meta def. *d2*
extension scheme *x*
meta def. *x1*
meta def. *x2*
extension scheme *y*
meta def. *y1*
meta def. *y2*
meta def. *3*
copy
<Extensions>
<Extension> <P...Prefix>**a**</P...Prefix> ... ***a1*** ... ***a2*** ... </Extension>
<Extension> <P...Prefix>**d**</P...Prefix> ... ***d1*** ... ***d2*** ... </Extension>
<Extension> <P...Prefix>**x**</P...Prefix> ... ***x1*** ... ***x2*** ... </Extension>
<Extension> <P...Prefix>**y**</P...Prefix> ... ***y1*** ... ***y2*** ... </Extension>
<Extension> <P...Prefix>**this**</P...Prefix> ... ***3*** ... ***4*** ... </Extension>
...
</Extensions>

Figure D.3 – Adding extension meta dictionaries to Reg-XML documents

Figure D.3 shows the collecting together of application metadata schemes, descriptive metadata schemes, extension schemes and individual meta definitions to create the extension meta dictionaries of a Reg-XML document. In the figure:

- Application metadata plugin schema *a* and descriptive metadata scheme *d* are defined by SMPTE engineering documents external to the MXF file.

- Complete extension schemes *x* and *y* are encoded as meta dictionaries in the KLV stream before the metadata values according to KXS.

- Definitions *3* and *4* are found outside the scope of a scheme and so included in the local extension meta dictionary with prefix *this*.

The preface element has properties *DM Schemes* and *Application Schemes Batch* that provide the identifiers of the plug-in schemes that may be found in the header metadata set. Where a Reg-XML compliant data model and corresponding meta dictionary for a plug-in external metadata scheme is available, this is included as an extension metadata scheme for the resulting AAF Reg-XML document, according to data mapping rule 2 of SMPTE ST 2001-1. Inclusion is achieved by embedding a copy of the meta dictionary in the Reg-XML file.

According to KXS, separate meta dictionaries can be encoded within a header metadata set with their own SchemeID, SchemeURI and PreferredPrefix. These are encoded as separate extension meta dictionaries in the Reg-XML document.

Any local meta definitions encoded independently from (outside of) any particular scheme of the header metadata set are considered to be in the scope of this header metadata set only. A new local extension meta dictionary is created specifically for the Reg-XML document, included as a child of the extensions element, identified by a newly created UUID SchemeID and with PreferredPrefix set to “this”.

Meta definitions that are part of the AAF baseline meta dictionary defined in Annex B of this document are not encoded in extension meta dictionaries, even if they are repeated as MXF local sets. The meta definitions of the AAF baseline meta dictionary, as defined in this document, take precedence over any meta definitions with the same identifier encoded in an MXF file.

D.2.4 Streams

The body streams and index streams of the MXF file are extracted and written to stream files or resources of the AAF Reg-XML file group. These files are referenced from the header metadata in the AAF Reg-XML document according to stream-type data mapping in rule 5.11 of SMPTE ST 2001-1.

The source values of stream type property values are stored within in an MXF file using an MXF essence container. The essence container is split across one or more partitions of an MXF file. The Reg-XML external data representation contains the same essence container bytes with all partitions removed. Any specific container used to wrap essence into the stream is preserved, such as the SMPTE ST 379-1 MXF generic container.

Writing the essence containers to external stream files does not cause a change to the stated essence container of the associated file descriptor. Also, writing the essence containers to external stream files does not cause a change the stated essence containers of the preface.

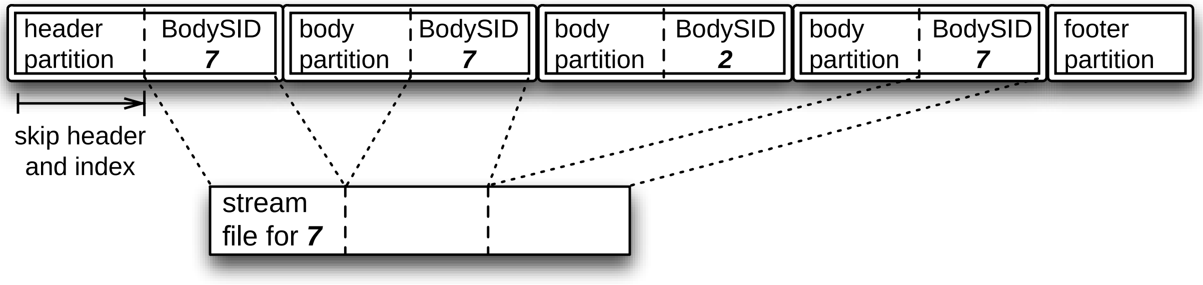


Figure D.4 – Concatenating parts of an MXF essence container to make a stream file

The relationship between MXF essence container streams and an external stream is illustrated in Figure D.4.

Typically, a body stream will consist of KLV packets of a MXF generic essence container distributed across partitions that have the same body SID. These KLV packets are concatenated together in the same order as they appear in the source MXF file. Details of how a body stream is partitioned in the originating MXF file are not preserved in the Reg-XML stream representation.

Streams containing MXF-defined index tables are mapped into a stream file containing a sequence of KLV-encoded index table segments. These are concatenated together in the same order they appear in the MXF file. Details of how the index table was partitioned in the originating MXF file are not preserved in the Reg-XML stream representation.

D.2.5 AAF Dictionary

AAF data decoded from SMPTE ST 377-1 files does not normally contain an AAF dictionary. AAF dictionaries contain the data definitions, codec definitions and container definitions that are the target of weak references in the file. An AAF Reg-XML file is also not required to contain an AAF dictionary. However, an AAF Reg-XML document may create a dictionary as a child element of the preface.

Weak-reference-type-specific rules in SMPTE ST 2001-1 define how to resolve weak references to definitions contained in a dictionary within the same file or to definitions that are external to the file.

D.3 Decode an AAF Reg-XML File Group, Encode as an MXF File

Any MXF file created from an AAF Reg-XML file group has to be a legal MXF file according to SMPTE ST 377-1. This section describes issues relating to starting with AAF instance data in an AAF Reg-XML file group format and encoding it as an MXF file. This includes options for how the MXF file is structured and identifying the kinds of AAF Reg-XML file group that are appropriate for mapping to MXF. Guidelines for encoding header metadata and mapping extension meta dictionaries into the MXF are provided.

D.3.1 MXF File Structure

A MXF file created from an AAF Reg-XML representation has to be compliant with the operational pattern specified in the preface root object. This is unlikely to cause an issue as the package structure within the AAF Reg-XML file ought to be compliant already.

To be appropriate for encoding in an MXF file, the essence containers specified in the preface root object of the Reg-XML document need to be kinds of KLV-encoded MXF essence container, such as the SMPTE ST 379-1 MXF generic container. Other kinds of essence container require modification of the wrapping of essence data to make it suitable for inclusion in an MXF file and such AAF Reg-XML file groups are not directly encoded to MXF files.

Where the AAF Reg-XML file group is not appropriate for directly decoding and encoding as a MXF file, an interim AAF Reg-XML file group could be created. In the interim representation, the wrapping of the essence data is changed to make use of MXF essence containers. In this case, immutability of essence conditions require that these new containers are described by a new source package with a new package identifier. The new source package will have a new file descriptor, with its essence container label set to identify a kind of MXF essence container. In turn, updates need to be made to the essence containers listed in the preface.

The MXF file or files created from an AAF Reg-XML representation can have any legal partition layout that preserves the streams of the essence containers. Header metadata derived from the Reg-XML document may be repeated in the file and is carried in closed complete partitions.

In the context of the structural choices made when encoding AAF data to an MXF file, an AAF Reg-XML file group is a form of structure-independent representation of AAF data. Decoding an AAF Reg-XML file group to encode it as an MXF file and then doing the same process produces an identical AAF Reg-XML file group. This is the case independently from the partition structure of the intermediate MXF file. However, starting with an MXF file and carrying out a round-trip through Reg-XML does not necessarily produce an identical MXF file as the partition layout can be altered.

D.3.2 Simplest MXF File Structure

The simplest legal MXF file structure that can be created for an AAF Reg-XML file group is illustrated in Figure D.5.

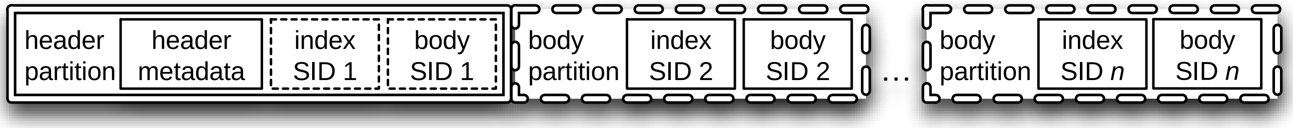


Figure D.5 – Simplest MXF file structure for an AAF Reg-XML file group

The simplest file structure consists of one partition per essence container, as follows:

1. A header partition containing a single copy of the header metadata decoded from the AAF Reg-XML document of the AAF Reg-XML representation.

2. The essence container associated with the first AAF essence data encountered when decoding the header metadata is copied from its AAF Reg-XML representation external file or resource into the header partition. If the essence container has an index table, the table is also copied into the same header metadata partition.

3. Subsequent AAF essence data objects in the header metadata result in additional body partitions in the MXF file, with each partition containing a complete essence container and its associated index table.

D.3.3 MXF File Header Metadata

The AAF Reg-XML document of an AAF Reg-XML file group that contains AAF instance data is used to create at least one complete header metadata item in the MXF file.

According to SMPTE ST 2001-1, every object encoded in the Reg-XML representation has a corresponding class definition in either the associated baseline data model or an extension data model described by an embedded extension meta dictionary. The fully qualified element name of the class (namespace and name) of the object will correspond to the SchemeURI of the dictionary and symbol of the class definition it defines. The value of each object is encoded in the MXF header metadata as an MXF local set with the key of that set equal to the identifier of the class definition.

Every object in the Reg-XML document is assigned a unique *Instance UID* property to be used within the context of the header metadata. Where the object is the target of a strong reference, the instance UID is used to make the reference. The instance UID is encoded as a property of the object it identifies using the item UL and tag specified in SMPTE ST 377-1:

Item UL: *urn:smpte:ul:060E2B34.01010101.01011502.00000000*, local tag: *0x3c0a*.

The property values of each object have corresponding property definitions in either the associated baseline data model or an extension data model described by an embedded extension meta dictionary. The fully qualified element name of the property value (namespace and name) will correspond to the SchemeURI and symbol of the property definition.

The identifier of the property definition is used to create a local tag entry in the primer pack of the header metadata. Where the local identification of the property definition is greater than zero and less than 0x8000, the local identification of the property definition is used as the tag. Otherwise, a new tag is generated for use locally, unique within the scope of the header metadata set, with a value in the range 0x8000 to 0xffff.

The value of the property is in the MXF file as a property of the MXF local set representing the object that the property is a member of. The local set tag used to signal the property is that which assigned to the property in the primer pack. The property value is encoded by type according to the rules of SMPTE ST 377-1.

Encoding of MXF local sets starts with the preface root object. Every property of strong reference type, including strong reference sets and strong reference vectors, has a target object. All target objects are encoded in the sequence of MXF local sets of the header metadata.

D.3.4 Extension Meta Dictionaries

Carriage of meta dictionaries and meta definitions is not specified by SMPTE ST 377-1. According to currently published SMPTE standards, any class that is not defined in SMPTE ST 377-1 will be encoded as if dark metadata in a MXF file encoded from an AAF Reg-XML file group. Best effort ought to be made to encode extension data instances in the MXF file, even though no guarantee exists that this data will be accessible by a decoder.

A MXF decoder that is configured with access to the application metadata schemes and descriptive metadata schemes contained in the MXF file has access to data instances defined with reference to these schemes.

Extension meta dictionaries can be encoded as MXF local sets in header metadata using the KLV Extension Syntax (KXS) as envisioned by the ongoing SMPTE ST 377-2 project. It is recommended that the extensions are located between the end of the primer pack and the start of the MXF local set encoding of AAF interchange objects.

D.3.5 MXF File and the AAF Dictionary

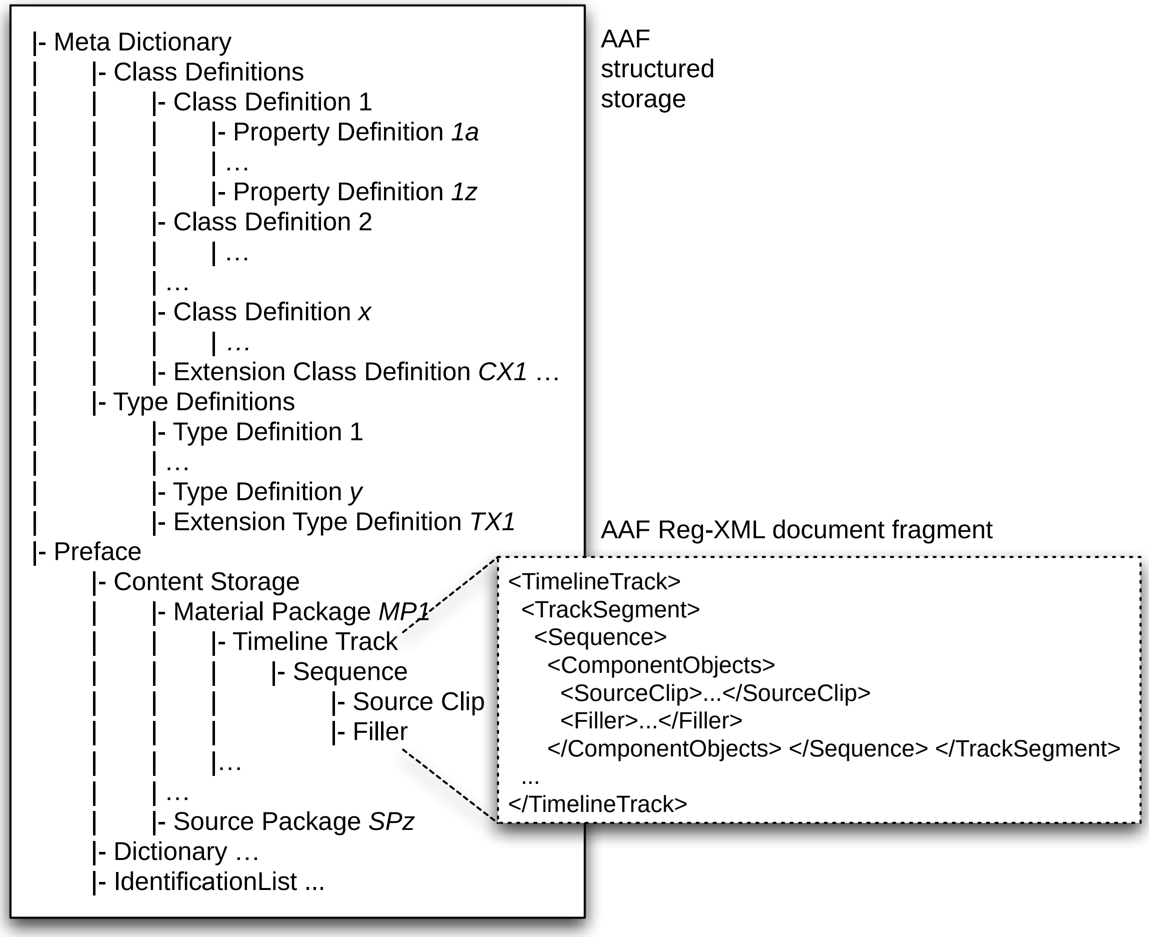
MXF files do not normally contain a copy of the AAF dictionary, describing details of data, codec and container definitions found in the file. Encoding the dictionary in the MXF file will appear as dark metadata to many MXF decoders. However, it is recommended that the dictionary be preserved during the encoding of the MXF file so that round trip conversions of the file also preserves the dictionary data. Sufficient meta definitions to carry out the encoding of an AAF dictionary are provided in the AAF baseline data model; e.g. DataDefinition, CodecDefinition, etc.

For reasons of interoperability, when choosing to not encode a definition into a file, it is important to consider whether the definition is available in a location that is accessible to the MXF decoder that will be decoding the file. For example, it is best to make use of labels that are registered in the SMPTE Labels Register.

Annex E Guidelines: AAF Files and AAF Reg-XML File Groups (Informative)

E.1 Mapping Between Structured Storage Files and AAF Reg-XML File Groups

The relationship between the AAF structured storage encoding format for AAF data and AAF Reg-XML file groups is illustrated in Figure E.1.



```
|- Meta Dictionary
|       |- Class Definitions
|       |       |- Class Definition 1
|       |       |       |- Property Definition 1a
|       |       |       | …
|       |       |       |- Property Definition 1z
|       |       |- Class Definition 2
|       |       |       | …
|       |       | …
|       |       |- Class Definition x
|       |       |       | …
|       |       |- Extension Class Definition CX1 …
|       |- Type Definitions
|               |- Type Definition 1
|               | …
|               |- Type Definition y
|               |- Extension Type Definition TX1
|- Preface
        |- Content Storage
        |       |- Material Package MP1
        |       |       |- Timeline Track
        |       |       |       |- Sequence
        |       |       |               |- Source Clip
        |       |       |               |- Filler
        |       |       |…
        |       | …
        |       |- Source Package SPz
        |- Dictionary …
        |- IdentificationList ...
```

```
<TimelineTrack>
  <TrackSegment>
    <Sequence>
      <ComponentObjects>
        <SourceClip>...</SourceClip>
        <Filler>...</Filler>
      </ComponentObjects> </Sequence> </TrackSegment>
  ...
</TimelineTrack>
```

Figure E.1 – AAF structured storage file and AAF Reg-XML document fragment

Structured storage format files, the underlying format for AAF binary files, contain a filing system within the file. The AAF Structured Storage Mapping Specification defines how AAF data is encoded within the generic structured storage format container.

The root folder of an AAF file contains two sub-folders:

1. A meta dictionary with all the class, property and type definitions used in the file. This will be a mix of the AAF baseline definitions contained in this document and extension definitions. Extension definitions are not organized into separate schemes in this format.

2. The preface object (still known as *header* within these files for legacy reasons) that is the root object of the encoding of essence and metadata within the file. The hierarchical sub-folder nesting within the structured storage format is similar to the element nesting within an AAF Reg-XML document.

Note: The folder-to-sub-folder nesting in a structured storage file is not as deep as the element-to-child-element nesting of an AAF Reg-XML document. In a structured storage file, the properties of an object are stored as a *properties* file within the folder representing the object's value. Other siblings at the same level in folder structure contain the values targeted by strong object references. In contrast for a Reg-XML document, each property is represented by separate child-elements. Targets of strong object references are encoded as child elements of the property that defines them, as grandchildren of the object's value, rather than as siblings to the property value.

E.2 Decode AAF Structured Storage Files, Encode as AAF Reg-XML File Groups

E.2.1 Matching Objects to Meta Definitions

One AAF XML file group is created for one AAF structured storage file. The AAF XML representation contains a Reg-XML document according to the data mapping rules of SMPTE ST 2001-1.

To encode an interchange object (sub-class of InterchangeObject) using the data mapping rules of SMPTE ST 2001-1, corresponding meta definitions are located and used. Meta definitions can be located as follows:

1. The ObjectClass property provides a weak reference to the class definition.

    a. If the identifier of the class definition matches a class definition of the AAF baseline meta dictionary of Annex B, the encoding of the object in a Reg-XML document uses the matched baseline class definition.

    b. Otherwise, the class is an extension class and its class definition, associated property definitions and type definitions are encoded as local meta definitions as described in Section E.2.2. The encoding of the object's value in the Reg-XML document then uses the newly created class definition in the extension meta dictionary.

2. Within the AAF structured storage file, each property of the object is described by a property definition contained within the associated class definition.

    a. If the identifier of the property definition contained within the AAF structured storage file matches that of a property definition in the AAF baseline data model, the encoding of the property in a Reg-XML document uses the matched baseline property definition.

    b. Otherwise, the property is an extension property and its property definition is encoded as described in Section E.2.2. The encoding of the property's value in the Reg-XML document uses the newly crated property definition in the extension meta dictionary.

Where the class definition in the AAF structured storage file and the matched class definition in the AAF baseline contain conflicting information, the version in the AAF baseline meta dictionary takes precedence.

E.2.2 Extension Meta Dictionary

Meta definitions of the meta dictionary contained in an AAF file that are also contained in the AAF baseline data model, as matched by their identifier, are not encoded within the AAF Reg-XML document.

Meta definitions of the meta dictionary contained in a source AAF file that are not also contained in the AAF baseline data model, as matched by their identifier, are encoded as an extension meta dictionary of the AAF Reg-XML document.

Extension meta definitions contained in an AAF structured storage file currently lack the symbol property required to determine their XML element name. Before inserting a meta definition into the extension meta dictionary, a symbol name is generated according to the following rules:

1. If the MetaDefinitionName property of the meta definition is a valid XML element name (XML schema NCName type), this name is used.

2. If the MetaDefinition Name is not a valid XML element name (XML Schema NCName type), the MetaDefinitionName property has to be converted to a valid NCName. This conversion replaces non-NCName characters with underscore (‘_’) characters.

3. Symbols need to be unique within the same meta dictionary (SchemeURI/Namespace) and so additional characters can be added to the symbol name to preserve the uniqueness constraint.