



**ENGLISH TRANSLATION**

TECHNICAL METHODS FOR  
FILE-BASED PROGRAM EXCHANGE

ARIB TECHNICAL REPORT

ARIB TR-B31 Version 1.1-E1

Excerpted Version

Established on April 26, 2010

Version 1.0

Revised on March 28, 2011

Version 1.1

### **General Notes to the English translation of ARIB Standards and Technical Reports**

1. The copyright of this document is ascribed to the Association of Radio Industries and Businesses (ARIB).
2. All rights reserved. No part of this document may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without the prior written permission of ARIB.
3. The ARIB Standards and ARIB Technical Reports are usually written in Japanese and approved by the ARIB Standard Assembly. This document is a translation into English of the approved document for the purpose of convenience of users. If there are any discrepancies in the content, expressions, etc., between the Japanese original and this translated document, the Japanese original shall prevail.
4. The establishment, revision and abolishment of ARIB Standards and Technical Reports are approved at the ARIB Standard Assembly, which meets several times a year. Approved ARIB Standards and Technical Reports, in their original language, are made publicly available in hard copy, CDs or through web posting, generally in about one month after the date of approval. The original document of this translation may have been further revised and therefore users are encouraged to check the latest version at an appropriate page under the following

URL:

<http://www.arib.or.jp/english/index.html>

## Introduction

With participation of radio communication equipment manufacturers, broadcasting equipment manufacturers, telecommunication operators, broadcasters and general equipment users, Association of Radio Industries and Businesses (ARIB) defines basic technical requirements for standard specifications of radio equipment, etc. as an "ARIB STANDARD" or an "ARIB TECHNICAL REPORT" in the field of various radio systems.

An ARIB TECHNICAL REPORT provides the industry with specifications designed to ensure the quality and compatibility of radio equipment, particularly with regards to measurement and operational methods, based on "ARIB STANDARD" derived from technical standards released by the national government as well as voluntary standards used in the industry.

An ARIB TECHNICAL REPORT herein is published as "TECHNICAL METHODS FOR FILE-BASED PROGRAM EXCHANGE." In order to ensure fairness and transparency in the defining stage, the technical report was set by consensus of the standard council with participation of interested parties including radio equipment manufacturers, telecommunication operators, broadcasters, testing organizations, general users, etc. with impartiality.

The broadcasting equipments are now evolving by applying the information technology in various fields, and treat broadcasting material not as conventional stream, but as a file has become realistic. For instance, in the post-production area, such as Non-Linear Editing System which process video & audio as files, in the master control, such as server based play out devices, in the video recording area, such as recording devices which record video & audio as files and store in memory devices, are commonly used now.

In case of broadcasting material exchange by online transmission, an experimental attempt has increased not using a large capacity leased line, but using file transmission by general-purpose IP network.

Under the circumstances, needs for the standardization of the file format to enable the file based program exchange between broadcasting stations arose, and this technical report was established.

It is our sincere hope that the technical report would be widely used by radio equipment manufacturers, testing organizations, general users, etc.



## Table of Contents

## Introduction

Chapter 1	General Items .....	1
1.1.	Purpose.....	1
1.2.	Referenced Documents .....	1
1.2.1	Normative References .....	1
1.2.2	Related Documents .....	3
1.3.	Acronyms .....	4
Chapter 2	Scope.....	5
2.1.	Reference Model.....	5
2.2.	Distribution Model .....	5
2.3.	Distribution Package .....	7
2.3.1	Composition of the Distribution Package .....	8
2.3.2	Minimal Composition of the Distribution Package.....	9
2.4.	Dividing an Asset.....	9
2.4.1	Roll.....	9
2.4.2	Relationship of Program and Roll .....	10
2.5.	Operations Example.....	12
Chapter 3	Explanation of MXF Standards .....	14
Chapter 4	Video / Audio Data Encoding .....	15
4.1.	General .....	15
4.2.	Operational Pattern .....	15
4.2.1	Operation of OP-1a.....	16
4.2.2	Operation of OP-Atom .....	17
4.3.	Essence GC Mapping .....	18
4.3.1	OP-1a.....	19
4.3.1.1	Intra-Frame Compression .....	19
4.3.1.2	Long GOP Compression.....	22
4.3.2	OP-Atom .....	24
4.3.2.1	Intra-Frame Compression .....	24
4.3.2.2	Long GOP Compression.....	26
Chapter 5	Closed Caption / Ancillary Data Encoding .....	27
5.1.3.4	Operational Guideline .....	27

5.2.3	Method for Conveying Ancillary Packet .....	28
Chapter 6	Program Exchange Metadata .....	29
6.1.	Outline of Program Exchange Metadata .....	29
6.1.1	Program Exchange Metadata .....	29
6.1.2	Program Exchange Metadata Description Format .....	29
6.1.2.1	Basic Program Exchange Metadata Description Format .....	29
6.1.3	Relations between Program Exchange Metadata Documents and MXF Files .....	29
6.2.	Program Exchange Metadata Encoding .....	31
6.2.1	Description Language .....	31
6.2.2	Character Encoding .....	31
6.2.3	Name Space .....	31
6.2.4	Data Hierarchy .....	32
6.2.5	Version Management .....	32
6.3.	Program Exchange Metadata Description Format .....	33
6.3.1	Program Exchange Metadata Document .....	33
6.3.2	Details of Framework .....	35
6.3.2.1	Program Framework .....	35
6.3.2.2	Roll Framework .....	37
6.3.3	Details of Sets .....	38
6.3.3.1	Titles .....	38
6.3.3.2	Identification .....	39
6.3.3.3	GroupRelationship .....	40
6.3.3.4	Event .....	41
6.3.3.5	Publication .....	43
6.3.3.6	Annotation .....	44
6.3.3.7	Classification .....	44
6.3.3.8	Participant .....	45
6.3.3.9	Person .....	46
6.3.3.10	Organization .....	47
6.3.3.11	Address .....	49
6.3.3.12	Communications .....	50
6.3.3.13	Contract .....	51
6.3.3.14	Rights .....	53
6.3.3.15	Playlist .....	54
6.3.3.16	VideoDescription .....	56

6.3.3.17	AudioDescription.....	57
6.3.3.18	CaptionsDescription.....	59
6.3.3.19	AncillaryDescription .....	61
6.3.3.20	FileDescription .....	62
6.3.3.21	Block.....	65
6.3.3.22	Keypoint.....	66
6.3.3.23	CueSheet.....	68
6.4.	Operational Guidelines of Program Exchange Metadata .....	73
6.4.1	File Format of Program Exchange Metadata Document.....	73
6.4.2	Extension of Program Exchange Metadata Document .....	73
6.4.3	Operational Guideline of Blocks.....	73
6.4.4	Division of Program Exchange Metadata.....	74
6.4.5	Operational Guidelines for Identification element and MXFInfo Element .....	75
6.4.6	Operational Guidelines for Identification Set.....	75
6.5.	Mapping of Program Exchange Metadata Document to DMS-1 .....	76
6.6.	Embedding Method of Program Exchange Metadata Document into MXF File.....	76
Chapter 7	Transfer of Distribution Package.....	77
7.1.	Package Information Document.....	77
7.1.1	Description Language .....	77
7.1.2	Character Encoding .....	77
7.1.3	Filename .....	77
7.1.4	Structure of Package Information Document .....	77
7.1.4.1	The Overview of the Package Information Document Structure.....	78
7.1.4.2	Asset Element.....	79
7.1.5	Operations of the Package Information Document .....	80
7.1.5.1	Operation of ProgramId Element .....	80
7.1.5.2	Operation of MediaType Element .....	80
7.1.5.3	Dividing an Asset .....	81
7.1.6	Version Management .....	81
7.2.	Program Playlist Document.....	82
7.2.1	Description Language .....	84
7.2.2	Character Encoding .....	84
7.2.3	Filename .....	84
7.2.4	Structure of the Program Playlist Document .....	84
7.2.4.1	The Overview of the Program Playlist Document Structure.....	85

7.2.4.2	Roll Element .....	86
7.2.4.3	Asset Element.....	86
7.2.5	Operation of the Program Playlist Document.....	87
7.2.5.1	Specifying a File in Asset Element.....	87
7.2.5.2	Identifying an Asset.....	87
7.2.5.3	Operation of MediaType Element .....	87
7.2.5.4	Timing parameters in Roll .....	88
7.2.6	Version Management .....	89
7.3.	Distribution Package Transfer .....	90
7.3.1	Transfer by a Physical Medium (recording medium) .....	90
7.3.1.1	Package Information Document .....	90
7.3.1.2	Program Playlist Document .....	91
7.3.2	Transfer through a Network .....	91
7.3.2.1	Package Information Document .....	91
7.3.2.2	Program Playlist Document .....	91
Appendix 1	User Requirement .....	92
Appendix 2	Program Exchange Metadata XML Schema .....	93
Appendix 3	Package Information / Program Playlist Document XML Schema .....	106
1	Package Information Document XML Schema .....	106
2	Program Playlist Document XML Schema.....	107
Appendix 4	Summary of Media Dispatch Protocol .....	109
Appendix 5	XML Diagram .....	110
1	Element Symbol.....	110
2	Model Symbol .....	110



## **Chapter 1      General Items**

### **1.1. Purpose**

The purpose of this technical report is to define a technical specification for exchanging video/audio data, metadata and closed caption data etc., on a file basis among broadcasting stations or between a broadcasting station and a content production, in Japanese digital broadcasting environment.

### **1.2. Referenced Documents**

#### **1.2.1 Normative References**

- (1) SMPTE 298-2009 Television - Universal Labels for Unique Identification of Digital Data
- (2) SMPTE 322M-2004 Television - Format for Transmission of DV Compressed Video, Audio and Data Over a Serial Data Transport Interface
- (3) SMPTE 326M-2000 Television - SDTI Content Package Format (SDTI-CP)
- (4) SMPTE 330M-2004 Television - Unique Material Identifier (UMID)
- (5) SMPTE 336M-2007 Data Encoding Protocol using Key-Length-Value
- (6) SMPTE 359M-2001 Television and Motion Pictures - Dynamic Documents
- (7) SMPTE 377-1-2009 Material Exchange Format (MXF) - File Format Specification (Revision of SMPTE 377M-2004)
- (8) SMPTE 378M-2004 Television - Material Exchange Format (MXF) - Operational Pattern 1A (Single Item, Single Package)
- (9) SMPTE 379-1-2009 Material Exchange Format (MXF) - MXF Generic Container (Revision of SMPTE 379M-2004)
- (10) SMPTE 380M-2004 Television - Material Exchange Format (MXF) - Descriptive Metadata Scheme-1 (Standard, Dynamic)
- (11) SMPTE 381M-2005 Television - Material Exchange Format (MXF) - Mapping MPEG Streams into the MXF Generic Container
- (12) SMPTE 382M-2007 Material Exchange Format - Mapping AES3 and Broadcast Wave Audio into the MXF Generic Container
- (13) SMPTE 383M-2008 Television - Material Exchange Format (MXF) - Mapping DV-DIF Data to the MXF Generic Container
- (14) SMPTE 384M-2005 Television - Material Exchange Format (MXF) - Mapping of Uncompressed Pictures into the Generic Container
- (15) SMPTE 385M-2004 Television - Material Exchange Format (MXF) - Mapping SDTI-CP Essence and Metadata into the MXF Generic Container
- (16) SMPTE 386M-2004 Television - Material Exchange Format (MXF) - Mapping Type D-10

- Essence Data to the MXF Generic Container
- (17) SMPTE 387M-2004 Television - Material Exchange Format (MXF) - Mapping Type D-11  
Essence Data to the MXF Generic Container
  - (18) SMPTE 388M-2004 Television - Material Exchange Format (MXF) - Mapping A-law Coded  
Audio into the MXF Generic Container
  - (19) SMPTE 390M-2004 Television - Material Exchange Format (MXF) - Specialized Operational  
Pattern "Atom" (Simplified Representation of a Single Item)
  - (20) SMPTE 410-2008 Material Exchange Format - Generic Stream Partition
  - (21) SMPTE 436M-2006 Television - MXF Mappings for VBI Lines and Ancillary Data Packets
  - (22) SMPTE RP 210 Metadata Dictionary Registry of Metadata Element Descriptions
  - (23) SMPTE RP 224 SMPTE Labels Register
  - (24) SMPTE EG41-2004 Television - Material Exchange Format (MXF) - Engineering Guideline
  - (25) SMPTE EG42-2004 Television - Material Exchange Format (MXF) - MXF Descriptive Metadata
  - (26) ARIB BTA S-005C - Ancillary Data Packet and Space Formatting of Bit-serial Digital Interface  
for 1125/60 HDTV Systems, Version C1.0, July 2009
  - (27) ARIB STD-B35 - Data Program Exchange Specification for Digital Broadcasting, Version 1.2,  
March 2006
  - (28) ARIB STD-B36 - Exchange Format of the Digital Closed Caption File for Digital Television  
Broadcasting System, Version 2.3, September 2007
  - (29) ARIB STD-B37 - Structure and Operation of Closed Caption Data Conveyed by Ancillary Data  
Packets, Version 2.4, March 2006
  - (30) ARIB STD-B39 - Structure of Inter-Stationary Control Data Conveyed by Ancillary Data Packets,  
Version 1.1, March 2003
  - (31) ARIB TR-B23 - Operational Guidelines for Ancillary Data in Inter-Stationary Information  
Exchange, Version 1.1, July 2003
  - (32) IEC61834-2 Recording - Helical-scan digital video cassette recording system using 6,35 mm  
magnetic tape for consumer use (525-60, 625-50, 1125-60 and 1250-50 systems) - Part 2: SD  
format for 525-60 and 625-50 systems
  - (33) ISO/IEC 10646-1:2000 Information technology - Universal Multiple-Octet Coded Character Set  
(UCS) - Part 1: Architecture and Basic Multilingual Plane
  - (34) ISO/IEC 10646-2:2001 Information technology - Universal Multiple-Octet Coded Character Set  
(UCS) - Part 2: Supplementary Planes
  - (35) ISO/IEC 10646-1:2000/Amd.1:2002 Information technology - Universal Multiple-Octet Coded  
Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane; Amendment 1:  
Mathematical symbols and other characters

- (36) ISO/IEC 11172-2 Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s - Part 2: Video.
- (37) ISO/IEC 13818-1 Information technology - Generic coding of moving pictures and associated audio information: Systems.
- (38) ISO/IEC 13818-2 Information technology - Generic coding of moving pictures and associated audio information: Video.
- (39) W3C XML World Wide Web Consortium (W3C) (2008, November 26). Extensible Markup Language (XML) 1.0 (Fifth Edition).
- (40) W3C XML Schema Part 1 World Wide Web Consortium (W3C) (2004, October 28). XML Schema Part 1: Structures (Second Edition).
- (41) W3C XML Schema Part 2 World Wide Web Consortium (W3C) (2004, October 28). XML Schema Part 2: Data types (Second Edition).

### **1.2.2 Related Documents**

- (1) SMPTE 2032-1-2007 Media Dispatch Protocol (MDP) - Protocol Specification
- (2) SMPTE 2032-2-2007 Media Dispatch Protocol (MDP) - MDP/XML/HTTP Mapping Specification
- (3) SMPTE 2032-3-2007 Media Dispatch Protocol (MDP) - Basic Target Pull Profile Specification

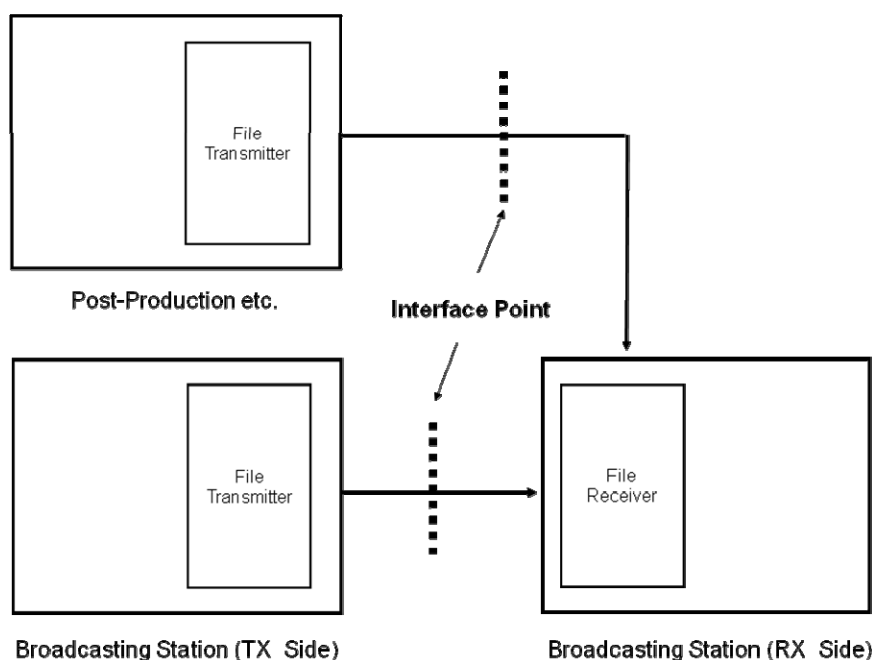
### 1.3. Acronyms

AAF	Advanced Authoring Format
BER	Basic Encoding Rules
CBE	Constant Bytes per Element
CBR	Constant Bit Rate
CDCI	Color Difference Component Image
DM	Descriptive Metadata
GOP	Group Of Pictures
KAG	KLV Alignment Grid
KLV	Key Length Value
MPEG	Moving Picture Experts Group
NDE	Number of Delta Entries in an Index Table
NIE	Number of Index Entries in an Index Table
NSL	Number of Slices in an Index Table minus 1
RIP	Random Index Pack
RGBA	Red Green Blue Alpha
UID	Unique Identifier
UL	Universal Label
UML	Unified Modeling Language
VBE	Variable Bytes per Element
VBR	Variable Bit Rate

## Chapter 2 Scope

### 2.1. Reference Model

This technical report provides a technical specification for broadcast content exchange on a file basis among broadcasting stations or between a broadcasting station and a content production. This report also defines the interface point for the file exchange among broadcasting stations or between a broadcasting station and an organization creating contents such as a content production, and specifies file formats, material types and the method of the file exchange used at the interface point. The reference model for the file exchange along with the interface point specified in this report is as illustrated in Figure 2-1.



**Figure 2-1 Reference model**

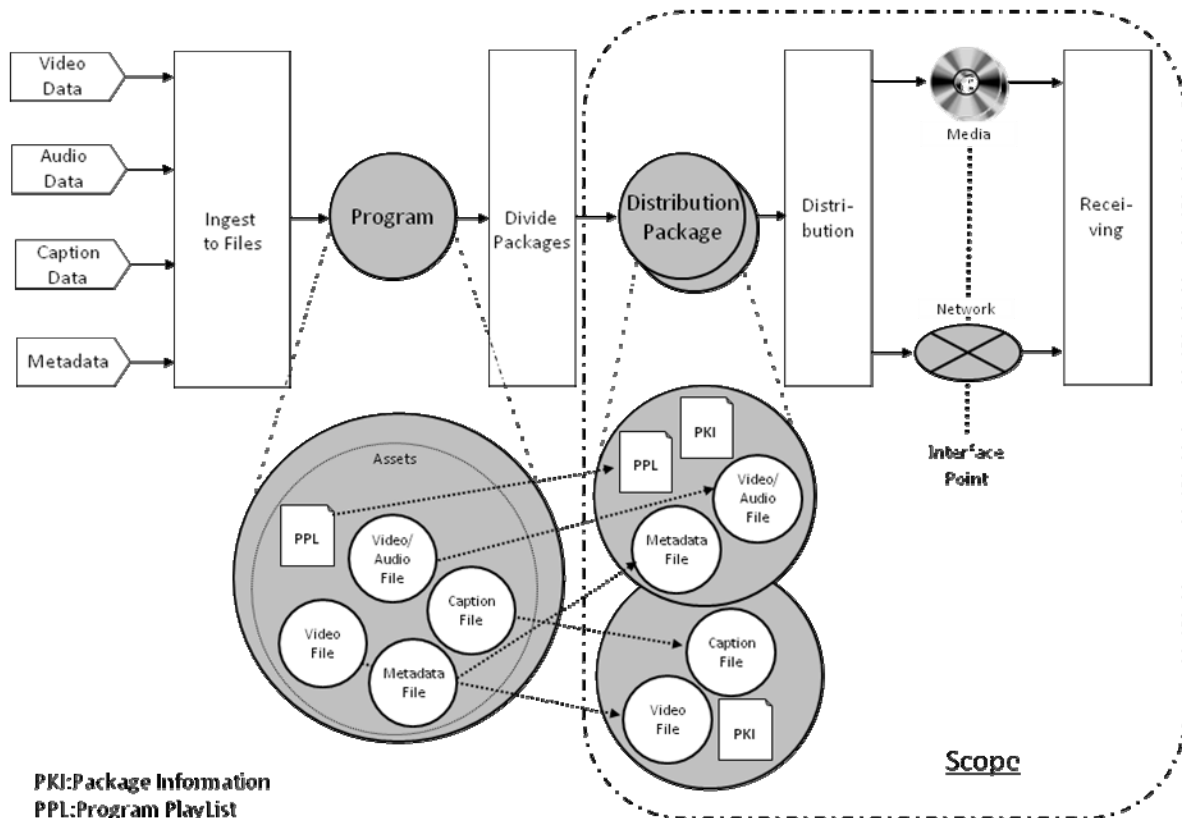
It should be aware that this report specifies the file formats and the other elements used for the file exchange between systems (e.g. among broadcasting stations or between a broadcasting station and a production) as stated above and is not intended to specify file formats to be handled by individual equipment.

### 2.2. Distribution Model

Digital broadcasting consists of two or more elementary streams of data representing the video and the audio. Therefore, broadcasting is not possible by simply distributing separate individual files. This document describes a method of grouping two or more files together in a form suitable for

distribution, together with the additional information required for broadcasting. Importantly, when the file is encoded, it takes a form in which the video data and the audio data are not externally recognizable.

A distribution model is shown in Figure 2-2.



**Figure 2-2 Example of distribution model**

After appropriate processing to create the file appropriate for distribution, data representing video/audio and closed captions are recognized as a program file group. The file group is created appropriate to the distribution means (=distribution package), and is distributed for the destination.

#### ■ Broadcast program

This is composed of a group of the finalized files suitable for broadcasting. The file group consists of individual files representing video/audio, closed captions, and metadata.

#### ■ Distribution package

This is used for program distribution. Each file that composes the program can be delivered in multiple distribution packages. Refer to the next paragraph for details of the distribution package.

#### ■ Asset

This is a generic name given to an individual file included in the program and in the distribution package. An asset represents the video/audio file, the closed caption file and the additional information file which is the program exchange metadata, the Package Information document and the Program Playlist document.

#### ■ Video/Audio data (Video/Audio file)

This indicated the video/audio data required for digital broadcasting. This may be uncompressed, or with appropriate compression as indicated. They represent the video/audio data which complete the editing and MA processes and finalize formatting ready for "on the air" (generally referred as "Kampake").

This document assumes that video/audio data that was made to the file and exchanged is ready for broadcasting without editing or with the editing of a simple trim. Therefore materials (e.g. raw materials acquired by a field camcorder, and materials sent from the field to studio facility and then requiring various editing works before broadcasting ) for which the editing work is necessary for broadcasting is out of the scope of this technical document..

Files created from video/audio data as described in Chapter 4 are called 'video file', 'audio file', and 'video/audio file'.

#### ■ Closed caption data (Closed caption file)

This is the textual and control information representing dialog and others that are displayed on screen and synchronized to video and audio when broadcast.

A file created from closed caption data according to the technique described in Chapter 5 is referred to as a 'closed caption file'.

#### ■ Metadata

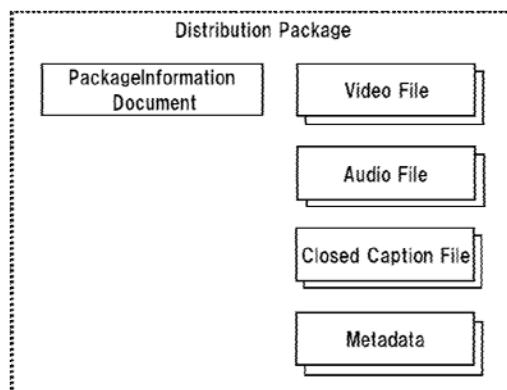
This is data describing the content of an asset. Unless otherwise noted, 'Metadata' indicates 'Program exchange metadata' as described in Chapter 6. Refer to Chapter 6 for details of the program exchange metadata.

### 2.3. Distribution Package

In this technical report, the distribution package is used as a concept for grouping several sets of files constituting a program properly in the delivering.

The distribution package is a concept that it organizes each file to deliver properly to its destination, and at the destination it is possible to unwrap it easily in order to reproduce with the consistency. A program can be transferred with one or multiple packages.

Figure 2-3 shows structure of the typical distribution package.

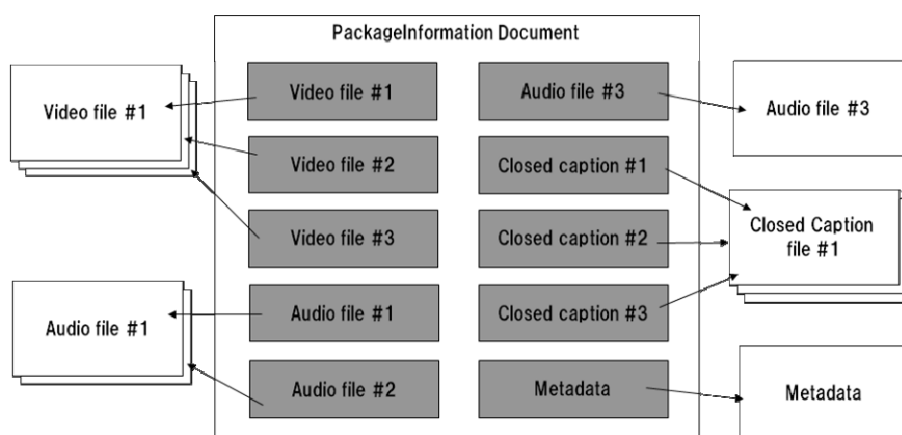


**Figure 2-3 Sample of distribution package**

The distribution package consists of one package information document and one or multiple files (i.e. asset).

The package information document includes information and identifier about individual files (i.e. asset) delivered as a distribution package. Figure 2-4 shows an example of reference relation between package information document and asset.

See Chapter 7 for details of package information document.



**Figure 2-4 Sample of reference relationship package information document and asset**

### 2.3.1 Composition of the Distribution Package

Asset that can be delivered as Distribution Package is shown below.

- Video and Audio files



Video and Audio file is a MXF file which unifies with video data and audio data. Individual form for the video file or audio file is also possible. See Chapter 4 for details of the MXF file.

■ Closed Caption file

Closed Caption file is a substance making a closed caption data to a file. See Chapter 5 for details of the closed caption file.

■ Video, Audio and Closed Caption files

Video, Audio and Closed Caption file is a MXF file which unifies with video, audio and closed caption data. See Chapter 4 and 5 for details of the MXF file.

■ Program Exchange Metadata

See Chapter 6 for details of the Program Exchange Metadata.

■ Package Information Document

See Chapter 7 for details of the Package Information Document.

■ Program Playlist Document

See Chapter 7 for details of the Package Information Document

It is not prohibited to include a file except an asset appointed as above in a distribution package.

### 2.3.2 Minimal Composition of the Distribution Package

A distribution package consists of one package information document and assets more than one (for example Video and Audio file) referred from a package information document.

## 2.4. Dividing an Asset

### 2.4.1 Roll

To make the program to a file, due to limitation of the editing and media resource size, video and audio data (Including closed caption) may need to be divided into multiple sections on timeline and make it to a file. In this report, these divided sections are called “roll”.

Roll is a group of video, audio, and closed caption file for which should be played in a same timeline.

Structure of roll shall be based on constraints as follows;

- One roll shall contain one asset (file) or more. Therefore, one asset shouldn't cross over multiple rolls.
- One roll shall not contain multiple assets which will not be played in a same time.

Dividing in a time line on Video, audio and closed caption shall be in timing. Start of the play time and duration need to be the same. But as for closed caption, one file can appear to multiple rolls as exceptional case. For more information, please refer Chapter 7.

### 2.4.2 Relationship of Program and Roll

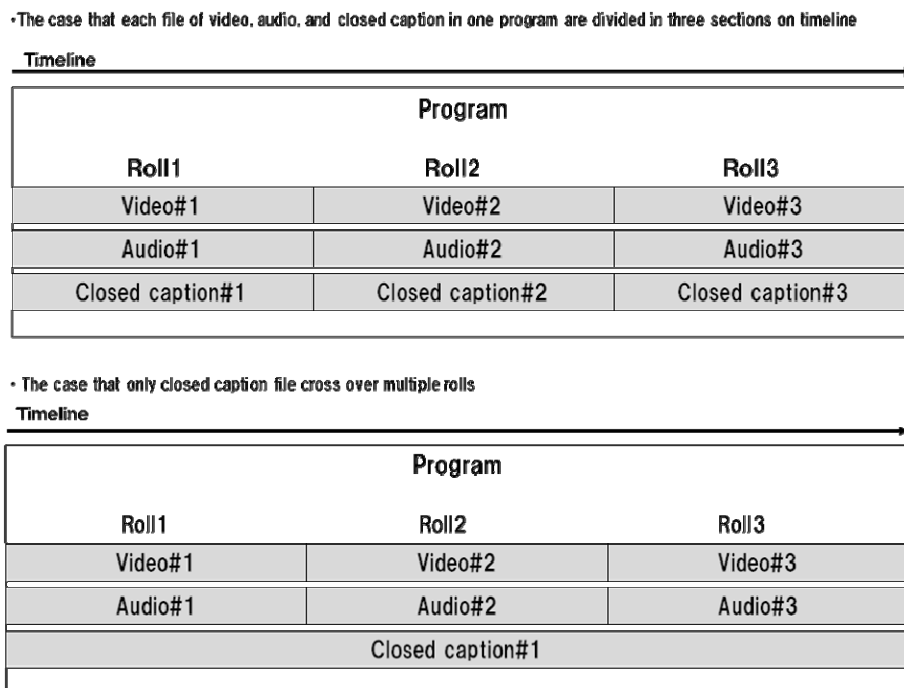
Figure 2-5 and Figure 2-6 show relation of program and roll.

Figure 2-5 shows the case that each file of video, audio, and closed caption in one program are divided in three sections on timeline; and only closed caption file cross over multiple rolls.

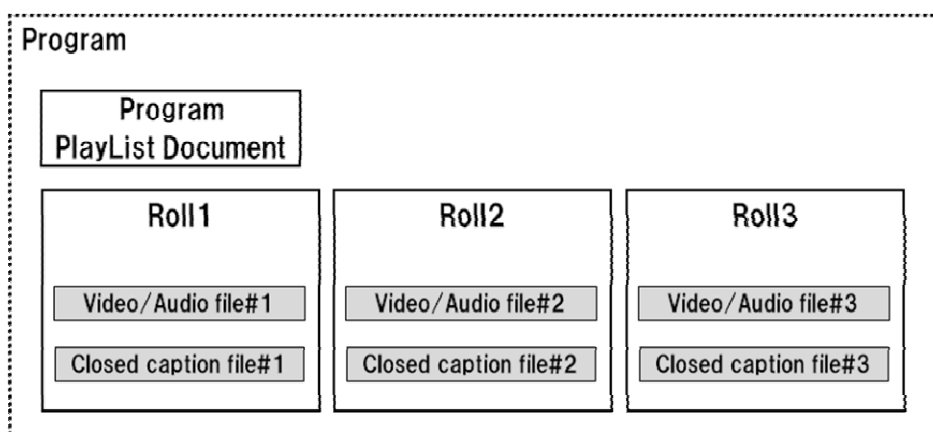
Video, audio and closed caption, or individual video and audio, which belong to each roll should be part of a file.

Figure 2-6 shows if program divided into three rolls, each video, audio and closed caption files become 2files (total of 6 files) as shown.

“Roll”s are assumed to be divided physically on timeline before making into file of each data on program; after the file is made, it does not specify the specific section on timeline in essence of the file. “Roll” should be assigned to a whole section of file itself. When assigned to specific section on timeline in file, concept of “block” is going to be used. (For more detail on “block”, refer on Chapter 6)



**Figure 2-5 Relation of program and roll**

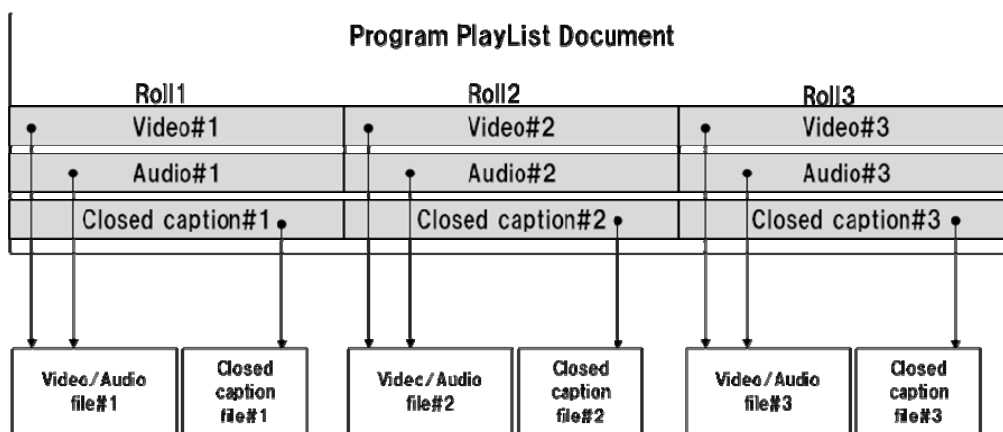


**Figure 2-6 Relation of roll and asset**

In this report, attached file called program playlist document is going to be used to see the relation between program and roll.

Program playlist document is a list of XML format file which shows order and structure of each roll of program.

Figure 2-7 shows structure of three rolls that describe program playlist and material of asset (video, audio, closed caption files)



**Figure 2-7 Example of program playlist and referred three assets**

## 2.5. Operations Example

The operations example is shown in figure below. These operations are just for example purposes and do not relate to actual operations.

Each program contains four video/audio files as shown in Figure 2-8, or contains four video files and four audio files as shown in Figure 2-9. Each program also contains one closed caption file, metadata and ProgramPlayList file. Four distribution packages are being used for program delivery.

In this case, three out of four packages (package1 to 3) are distributed via storage media, whereas network distribution is being used for package4.

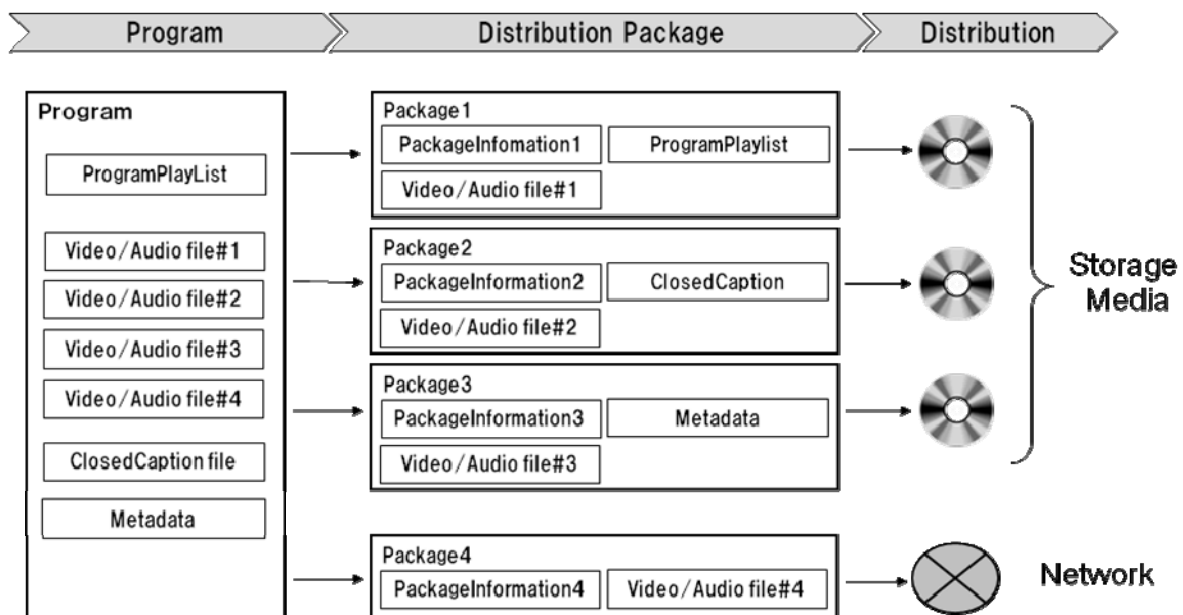


Figure 2-8 Operation example1 for program delivery with storage media and network

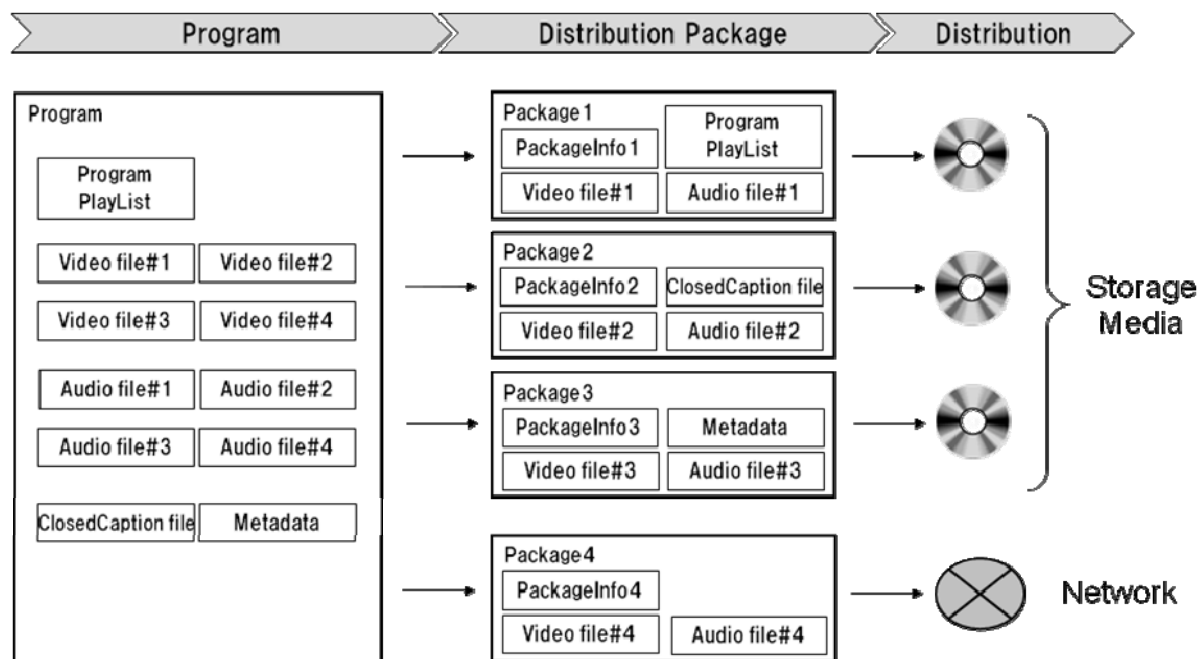


Figure 2-9 Operation example2 for program delivery with storage media and network

## **Chapter 3      Explanation of MXF Standards**

### **NO TRANSLATION**

This chapter describes summary of SMPTE MXF standards, therefore the translation is omitted.

## **Chapter 4      Video / Audio Data Encoding**

### **4.1.    General**

To create an audio-visual file, the file format shall be MXF.

The MXF file format is a comprehensive and scalable, and the MXF standard (SMPTE 377-1) defines the structure of the files and metadata.

An audio-visual file created shall be compliant with SMPTE 377-1. For more information about the MXF standard, see SMPTE 377-1.

This chapter defines MXF operational pattern and the mapping of audio-visual streams into MXF generic container.

The mapping into the MXF generic container shall be SMPTE 377-1, SMPTE 379-1, SMPTE 381M and SMPTE 382M compliant.

The operational patterns shall be SMPTE 378M and SMPTE 390M compliant, and this chapter defines constraints to ensure compatibility.

The MXF file extension shall be .mxf or .MXF.

This chapter defines constraints for major implementations, but it is not intended to exclude the undefined implementation.

### **4.2.    Operational Pattern**

It is preferred to narrow down the operational pattern in order to ensure compatibility of files and straightforward implementation of hardware in MXF file operation. Complex operational pattern in MXF file operation can provide the capability of performing various operations. On the other hand, it causes difficulty in ensuring compatibility of files between different vendors and difficulty in implementation of hardware.

As in the chapter 2, this technical report provides a technical specification for broadcast content exchange on a file basis among broadcasting stations or between a broadcasting station and a content production. The target operation will be exchange of broadcast program files. Therefore, this technical report specifies the program exchange by MXF file in operational pattern OP-1a, which Video/Audio essence can be monitored during data transmission.

Considering that each file that composed a program may be edited, this technical report also specifies the program exchange by MXF file in operational pattern OP-Atom, which each essence can be mapped to separate files.

The target is not necessarily the same in OP-1a and OP-Atom. Therefore, implementation can be done with only one depending on the operation. Moreover, an operational pattern is recommended to be able to convert mutually if interoperability of OP-1a and OP-Atom is necessary.

#### 4.2.1 Operation of OP-1a

In the Program Exchange by MXF File using OP-1a, completed video/audio essence is mapped into essence container of a single MXF file. Essence container shall be composed of frame interleaved video essence, audio essence, and if necessary, various data of closed caption etc.

It is optional to implement Index Table in OP-1a standards. However to be able to access each frame randomly, it is mandatory to implement Index Table that sets byte offset to all frames. Table 4-1 shows restrictions for OP-1a operation, and table 4-2 shows the setting of Operational Pattern UL Byte15 (Operational Pattern: Qualifiers) based on the restrictions.

Refer to SMPTE 378M for details of OP-1a.

**Table 4-1 Restrictions for OP-1a**

Items	Restrictions
Essence	The number of essence containers: 1 The essence container is an interleave form of video and audio essence, and must be contained in a file.
Material Package (Playback information of essence)	The number of Material Package: 1 Essence Track: Picture Track, Sound Track are mandatory. The number of SourceClip: 1 (Refer to single File Package)
Top-level File Package (Entity information of essence)	The number of Top-level File Package: 1 Essence Track: Picture Track, Sound Track are mandatory.
Lower-level Source Package (History information of essence)	Option
Primary Package (Prioritized Package)	Material Package
Partition restrictions	None
Body Partitions	Partition can be divisible as necessary.
Index Table	It is mandatory to implement Index Table that sets byte offset to all frames.
Streaming support	It shall be streamable file which each essence is interleaved. (Refer to Table 4-2).



**Table 4-2 Setting of Operational Pattern UL Byte 15 in OP-1a operation**

Bit number	Settings
0	1 : Marker bit
1	0 : Internal essence
2	0 : Stream file
3	1 : Multi-track
4-7	All set to 0 as reserve bit.

#### 4.2.2 Operation of OP-Atom

The Video Essence and (each channel of) the Audio Essence shall be wrapped in individual MXF files and the Material Package shall relate their synchronization, in case of program exchange by the MXF file using the OP-Atom as an operational pattern.

The default behavior of OP-Atom file playback is to playback the contained Essences according to description of Top-level File Package, because the OP-Atom standard describes that the File Package shall be the Primary Package which is prioritized Package.

In fact, playback referring to Material Package is treated as optional; however, it is mandatory to be able to playback based on Material Package in this technical report because related Video Essence and Audio Essence are treated as one content.

Table 4-3 shows the restrictions for OP-Atom operation, Table 4-4 shows the setting of Operational Pattern UL Byte 14 for OP-Atom operation.

Refer to SMPTE 390M for the detail of OP-Atom.

**Table 4-3 Restrictions for OP-Atom operation**

Items	Restrictions
Essence	The number of Essence Container : 1 Essence Container must have only one Essence and must be contained by File.
Material Package (Playback information of Essence)	The number of Material Package : 1 Essence Track: Picture Track and Sound Track are mandatory. The number of Source clip : one or more It is mandatory to implement the cross-reference function in order to synchronize the each Essence.
Top-level File Package (Entity information of Essence)	The number of Top-level File Package : 1 The number of Essence Track : 1
Lower-level Source Package (History information of Essence)	Option

Primary Package (Prioritized Package)	Top-level File Package
Partition restrictions	Header Partition : Closed and Complete Footer Partition : Complete
Body Partitions	It is prohibited to divide Essence into Partitions.
Index Table	It is mandatory to implement Index Table that is set byte offset for the all frames in Footer Partition.
Streaming Support	Depends on Essence Container.

**Table 4-4 Setting of Operational Pattern UL Byte 14 for OP-Atom operation**

Bit number	Set Value
0	0: Essence Track of Material Package contains an unique Source Clip. 1: Essence Track of Material Package contains one or more Source Clips.
1	1: Material Package contains one or more Essence Tracks.
2-7	All are set to zero as reserved bit.

### 4.3. Essence GC Mapping

The existing all video and audio streams should use widely-used format called MXF. To provide extended area for future release of new stream, Generic Container which is based on SMPTE 377-1, should use MXF based Essence Container (SMPTE 379-1), to use to wrap each essences.

MXF format, Generic Container, MPEG mapping, and Audio Mapping Standards enable to provide flexibility and broad-based format for option functions and implementation, but implementing to all MXF wrapper might be difficult.

This chapter describes about three layers of MXF implementation file based operation for interoperable file interchange.

- Element Key

Element key implementation is based on SMPTE379-1, SMPTE381M, SMPTE 382M, SMPTE 385M, and SMPTE436M.

- Essence Container Label

Essence Container Label implementation is based on SMPTE 379-1, SMPTE 381M,

SMPTE382M, and SMPTE436M.

- Essence Coding Label and Essence Descriptor

Essence Coding Label and Essence Descriptor implementation are based on SMPTE 377-1, SMPTE 381M, SMPTE 382M, and RP224.

Element Key and Essence Container Label of each element are standardized as follows;

- System Element:	SMPTE 379-1、	SMPTE 385M
- Picture Element:	SMPTE 379-1、	SMPTE 381M
- Sound Element:	SMPTE 379-1、	SMPTE 382M
- ANC Frame Element:	SMPTE 379-1、	SMPTE 436M
- GC MPEG ES Frame Wrap:	SMPTE 379-1、	SMPTE 381M
- GC AVC Byte stream Frame Wrap:	SMPTE 379-1、	SMPTE 381M
- GC AVC Byte stream Clip Wrap:	SMPTE 379-1、	SMPTE 381M
- GC AES Frame Wrap:	SMPTE 379-1、	SMPTE 382M
- GC AES Clip Wrap:	SMPTE 379-1、	SMPTE 382M
- GC BWF Frame Wrap:	SMPTE 379-1、	SMPTE 382M
- GC Generic ANC data:	SMPTE 379-1、	SMPTE 436M

Each coding labels are registered to RP224, MPEG Video Essence Descriptor and AES/BWF Audio Essence Descriptor are based as follows;

- Picture Essence:	SMPPE 377-1、 SMPTE 381M
- Sound Essence:	SMPPE 377-1、 SMPTE 382M

#### 4.3.1 OP-1a

In case of OP-1a, the operation for each compression scheme is as follows.

##### 4.3.1.1 Intra-Frame Compression

###### 4.3.1.1.1 Element Key/Essence Container Label

As the Element Key for the System Element, the Picture Element, and the Sound Element, the System Element Key, the Picture Element Key, the Sound Element Key, and AES/BWF Sound Element Key described in Table 3-9, 3-11, and 3-12 are used. The value of the Byte14 which defines the count of the essence elements is set to 01h (SDTI-CP Version1) for the System Element and 01h (single element) for the Picture Element. The Byte14 for the Sound Element Key is the

count of the audio channels multiplexed.

And the Byte15 which is the essence element type is set to 01h for the System Metadata Pack, 02h for the Package Metadata Set, 05h for the Picture Element (Frame Wrapping), and 03h for the Sound Element (AES3 Frame Wrapping).

As the Essence Container Label for the Picture Element, MPEG Essence Container Label in Table 3-7 is used. The Byte14 is set to 04h (MPEG ES) or 10h (AVC byte stream) according to the kind of the essence. And the Byte15 is set to 60h and Byte16 is set to 01h (Frame Wrapping).

As the Essence Container Label for the Sound Element, the label in Table 3-8 is used. The Byte15 is set to 01h (Wave Frame Wrapping) or 03h (AES3 Frame Wrapping).

The typical value for the element key and essence container are shown as follows.

(MPEG-2 Intra compression)

Picture Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.15.01.05.00
Sound Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.16.[C].03.[N]
(C: Count of audio channels N: The number of audio channel 00h - 07h)	
ANC Frame Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.17.01.02.01
GC MPEG ES Frame Wrap	: 06.0E.2B.34.04.01.01.02.0D.01.03.01.02.04.60.01
GC BWF Frame Wrap	: 06.0E.2B.34.04.01.01.01.0D.01.03.01.02.06.01.00
GC AES Frame Wrap	: 06.0E.2B.34.04.01.01.01.0D.01.03.01.02.06.03.00
GC Generic ANC data	: 06.0E.2B.34.04.01.01.09.0D.01.03.01.02.0E.00.00

(H.264/AVC Intra compression)

Picture Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.15.01.05.[E]
(E: Essence Element Number)	
Sound Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.16.[C].03.[N]
(C: Count of audio channels N: The number of audio channel 00h - 07h)	
GC AVC Byte stream Frame Wrap	: 06.0E.2B.34.04.01.01.0A.0D.01.03.01.02.10.60.01
GC AES Frame Wrap	: 06.0E.2B.34.04.01.01.01.0D.01.03.01.02.06.03.00

(MPEG-4 SStP Intra Compression)

System Metadata Pack	: 06.0E.2B.34.02.05.01.01.0D.01.03.01.04.01.01.00
Package Metadata Set	: 06.0E.2B.34.02.43.01.01.0D.01.03.01.04.01.02.[X]
(X: Number of metadata blocks in the element)	
Picture Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.15.01.05.00
Sound Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.16.[C].03.[N]
(C: Count of audio channels, N: The number of audio channel 00h - 07h)	
ANC Frame Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.17.01.02.01
GC MPEG ES Frame Wrap	: 06.0E.2B.34.04.01.01.02.0D.01.03.01.02.04.60.01
GC AES Frame Wrap	: 06.0E.2B.34.04.01.01.01.0D.01.03.01.02.06.03.00
GC Generic ANC data	: 06.0E.2B.34.04.01.01.09.0D.01.03.01.02.0E.00.00

#### 4.3.1.1.2 Required Essence Coding Label/Essence Descriptor properties for decoding

The Essence Coding Label in the Essence Descriptor specifies the kind of codec. The other metadata specifies the detail of the codec along with Essence Coding Label.

In case of MPEG-2 Intra compression, the Essence Coding Label and Essence Description decode mandatory items are as follows.

- Essence Coding Label

MPEG-2 MP@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.03.02.00
MPEG-2 422P@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.04.02.00
MPEG-2 MP@H14L	: 06.0E.2B.34.04.01.01.08.04.01.02.02.01.05.02.00

- Essence Descriptor decode mandatory item

All of the required items  
 MPEG Video Descriptor  
 StoredWidth, StoredHeight, AspectRatio // Best Effort  
 ClosedGOP, MaxGOP, IdenticalGOP, BitRate, CodedContentType  
 Generic Sound Essence Descriptor  
 ChannelCount, QuantizationBits, AudioSamplingRate

In case of H.264/AVC Intra compression, the Essence Coding Label and Essence Description decode mandatory items are as follows.

- Essence Coding Label

AVC High 10 Intra @L4, 50Mbps,1080/59.94i : 06.0E.2B.34.04.01.01.0A.04.01.02.02.01.32.21.01 AVC High 4:2:2 Intra @L4.1,100Mbps,1080/59.94i : 06.0E.2B.34.04.01.01.0A.04.01.02.02.01.32.31.01
---

- Essence Descriptor decode mandatory item

All of the required and best effort items MPEG Video Descriptor VerticalSubsampling, ContainerDuration
--

In case of MPEG-4 SStP Intra compression, the Essence Coding label and Essence Descriptor decode mandatory items are as follows.

- Essence Coding Label

MPEG-4 SStP@L2, 1080/59.94i 4:2:2 : 06.0E.2B.34.04.01.01.03 04.01.02.02.01.20.10.02 MPEG-4 SStP@L3, 1080/59.94i 4:4:4 : 06.0E.2B.34.04.01.01.03 04.01.02.02.01.20.10.03 MPEG-4 SStP@L4, 1080/59.94p 4:4:4 : 06.0E.2B.34.04.01.01.03 04.01.02.02.01.20.10.04
--

- Essence Descriptor decode mandatory items

All of the required and best effort items
---

#### 4.3.1.2 Long GOP Compression

##### 4.3.1.2.1 Element Key

Element Key for System Element, Picture Element and Sound Element shall use the System Element Key, Picture Element Key, and AES3/BWF Sound Element Key listed in the Table 3-9, Table 3-11 and Table 3-12 respectively.

The Byte14 specifying count of Essence Elements shall be set to 01h (SDTI-CP Version 1) in System Element and to 01h (single Element) in Picture Element. The Byte14 of the Element Key in Sound Element indicates the total sound channels multiplexed and shall be set to either 02h/04h/08h (2ch/4ch/8ch).

The Byte 15 specifying Essence Element Type shall be set to 01h for System Metadata Pack or 02h

for Package Metadata Set depending on Element Type, and to 05h (Frame Wrapping) for Picture Element and to 03h (AES3 Frame Wrapping).

For wrapping the System Metadata Pack and Package Metadata Set as System Element, the values of the Keys shall be as follows:

System Metadata Pack	: 06.0E.2B.34.02.05.01.01.0D.01.03.01.04.01.01.00
Package Metadata Set	: 06.0E.2B.34.02.43.01.01.0D.01.03.01.04.01.02.[X]
(X: number of Metadata blocks in the Element)	

For the Picture Element and Sound Element, the values of the Keys shall be as follows:

Picture Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.15.01.05.00
Sound Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.16.[C].03.[N]
(C: 02h/04h/08h (Count of audio channels), N: The number of audio channel 00h - 07h)	

For wrapping Ancillary data as Data Element, ANC Frame Element Key shall be used. The value of the Key shall be as follows:

ANC Frame Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.17.01.02.01
-------------------	---

#### 4.3.1.2.2 Essence Container Label

The MPEG Essence Container Label given in Table 3-7 shall be used as the Essence Container Label for the Picture Element.

Byte 14 indicates the mapping kind and shall be set to 04h for MPEG ES. Byte 15 shall be set to 60h and Byte 16 shall be set to 01h for frame wrapping.

The Essence Container Label given in Table 3-8 shall be used as the Essence Container Label for the Sound Element. Byte 15 shall be set to 03h for the AES3 frame wrapping. The values of the Essence Container Labels are as listed below.

GC MPEG ES Frame Wrap	: 06.0E.2B.34.04.01.01.02.0D.01.03.01.02.04.60.01
GC AES Frame Wrap	: 06.0E.2B.34.04.01.01.01.0D.01.03.01.02.06.03.00

The ANC Packet Essence Container Label shall be used for wrapping Ancillary data as the Data Essence. The value of the Essence Container Label is as given below.

GC Generic ANC data	: 06.0E.2B.34.04.01.01.09.0D.01.03.01.02.0E.00.00
---------------------	---

#### 4.3.1.2.3 Picture Essence Coding Label

The Picture Essence Coding property in the Essence Descriptor specifies the Essence coding scheme in use. The Label values for the MPEG-2 long GOP compression are as listed below.

MPEG-2 MP@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.03.03.00
MPEG-2 422P@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.04.03.00

#### 4.3.1.2.4 Required Essence Descriptor properties for decoding

Metadata properties in the Essence Descriptor specify the detailed information on the Essence Coding in use along with the Picture Essence Coding Label. The Essence Descriptor for the MPEG-2 long GOP compression shall be the MPEG Video Descriptor defined in SMPTE 381M. Required properties for decoding the MPEG-2 Long GOP stream are as listed below.

All Required, Best Effort properties
MPEG Video Descriptor
DisplayWidth, Vertical Subsampling
LowDelay, ClosedGOP, MaxGOP, BPictureCount, BitRate, ProfileAndLevel

### 4.3.2 OP-Atom

In case of OP-Atom, the operation for each compression scheme is as follows.

#### 4.3.2.1 Intra-Frame Compression

##### 4.3.2.1.1 Element Key

Implementations shall use the Picture Element Key and Sound Element Key as defined in the Table 3-11 and 3-12 respectively. In OP-Atom, only a single essence element shall be wrapped and Byte14 shall be set to 01h (a single essence element).

Byte15 of the Picture Element Key shall be set to 06h (clip wrapping), and Byte15 of the Sound Element Key shall be set to 04h (AES3 clip wrapping).

The values of the specific Keys shall be as follows.



Picture Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.15.01.06.nn
Sound Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.16.01.04.nn

The ANC Frame Element Key shall be used for wrapping the Ancillary data as a Data Element. The value of the specific Key shall be as follows.

ANC Frame Element	: 06.0E.2B.34.01.02.01.01.0D.01.03.01.17.01.02.01
-------------------	---

#### 4.3.2.1.2 Essence Container Label

The MPEG Essence Container Label given in Table 3-7 shall be used as the Essence Container Label for the Picture Element.

Byte 14 may vary in setting depending on the coding scheme in use. It shall be set to 04h (MPEG ES) for MPEG-2 streams and to 10h (AVC byte stream) for H.264/AVC streams.

Byte 15 shall be set to 60h and Byte 16 shall be set to 02h (Clip wrapping). The values of the specific Labels shall be as follows.

GC MPEG ES Clip Wrap	: 06.0E.2B.34.04.01.01.02.0D.01.03.01.02.04.60.02
GC AVC Byte stream Clip Wrap	: 06.0E.2B.34.04.01.01.0A.0D.01.03.01.02.10.60.02

The Essence Container Label given in Table 3-8 shall be used as the Essence Container Label for the Sound Element. Byte 15 shall be set to 04h (AES3 clip wrapping). The value of the specific Label shall be as follows.

GC AES Clip Wrap	: 06.0E.2B.34.04.01.01.01.0D.01.03.01.02.06.04.00
------------------	---

The ANC Packet Essence Container Label shall be used for wrapping Ancillary Data as the Data Element. The value of the specific Label shall be as follows.

GC Generic ANC data	: 06.0E.2B.34.04.01.01.09.0D.01.03.01.02.0E.00.00
---------------------	---

#### 4.3.2.1.3 Picture Essence Coding Label

The Picture Essence Coding property in the Essence Descriptor may have a different Label depending on the coding scheme in use. The values of Labels for the MPEG-2 Intra and H.264/AVC Intra compressions shall be as follows.

(MPEG-2 Intra compression)

MPEG-2 4:2:2 422P@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.04.02.00
----------------------	---

(H.264/AVC Intra compression)

High 10 Intra @L4, 50Mbps,1080/59.94i	: 06.0E.2B.34.04.01.01.0A.04.01.02.02.01.32.21.01
High 4:2:2 Intra @L4.1,100Mbps,1080/59.94i	: 06.0E.2B.34.04.01.01.0A.04.01.02.02.01.32.31.01

#### 4.3.2.2 Long GOP Compression

##### 4.3.2.2.1 Element Key

Refer to Element Key Value for OP-Atom/Intra-Frame in 4.3.2.1.1.

##### 4.3.2.2.2 Essence Container Label

Refer to Essence Container Label for OP-Atom/Intra-Frame in 4.3.2.1.2.

##### 4.3.2.2.3 Picture Essence Coding Label

The Picture Essence Coding property in the Essence Descriptor specifies the Essence coding scheme in use. The Label values for the MPEG-2 long GOP compression are as listed below.

MPEG-2 MP@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.03.03.00
MPEG-2 422P@HL	: 06.0E.2B.34.04.01.01.03.04.01.02.02.01.04.03.00

## Chapter 5 Closed Caption / Ancillary Data Encoding

### 5.1.

Section 5.1 through 5.1.3.3 describes summary of Japanese closed caption standard ARIB STD-B36, STD-B37 and SMPTE 436M, therefore the translation is omitted.

#### 5.1.3.4 Operational Guideline

This technical report specifies wrapping method of VANC closed caption data in accordance with ARIB STD-B37 in MXF file following to SMPTE 436M for closed caption data exchange. Wrapping VBI closed caption data into the MXF is out of scope of this technical report.

It is preferred to note as follows for sample conversion method of 8bit/10bit.

- In general MXF file, it is preferred to correspond to both sample coding method in order to operate without consideration whether 8bit and 10bit.
- However, it is not applicable in case of the restriction for the number of Sample Coding bits has been installed in a specific format of the manufacturer.

In production process, MXF file is edited from two or more materials.

In this case, it is undesirable for closed caption insertion processing that multiple VANC wrapping format is used in each material. (For instance, presence of SMPTE 436M Data Element, number of ancillary data packets, and 8bit/10bit etc.)

Therefore, the wrapping format of VANC data shall be same format in one MXF file.

Note that this technical report does not provide the specification of equipment processing MXF file, but specifies the guidelines between broadcasting systems.

Table 5-3 shows the operational guidelines for ancillary data packet of closed caption.

The area of packet 1 through 3 is prepared per field as ancillary data packet (ARIB STD-B37) of closed caption, and it becomes six packets in one frame.

In general, when the data area of MXF file is expanded in postprocessing, processing becomes complex. Because re-wrapping process for data insertion including Index Table and RIP is needed.

For program exchange, by considering ancillary data packet format of closed caption is a fixed length, there is method of reserving area for six caption ancillary data packets per frame in MXF file beforehand.

When there is no closed caption data in ancillary data area 1 through 3, though data area for ancillary data packet of closed caption is reserved in MXF file, "Invalid packet" specified in ARIB TR-B23 shall be inserted in ancillary data area 1 through 3, otherwise identifier specifies in ARIB STD-B37 shall be "no caption" .

**Table 5-3 Operational guidelines for ancillary data packet of closed caption**

Item	Operation guidelines
Applicable standard	SMPTE 436M VANC format
Line number	1125/60i:19 line/582 lines
Sample Coding method	8bit or 10bit Luma Samples (Y data series)
Others	Wrapping format (mentioned above) must be same in one MXF file.

#### 5.1.4

Section 5.1.4 through 5.2.2 describes summary of SMPTE 410 and ARIB TR-B23, therefore the translation is omitted.

#### 5.2.3 Method for Conveying Ancillary Packet

The method of conveying ancillary packets in MXF file shall be in accordance with SMPTE 436M. For detail, refer to chapter 5, section 5.1.3 "Method for conveying closed caption data in MXF file. (Translator's notes: As section 5.1.3 is not translated, please refer to section 5.1.3.4 "Operational Guideline" for detail.)

## **Chapter 6      Program Exchange Metadata**

### **6.1.    Outline of Program Exchange Metadata**

#### **6.1.1    Program Exchange Metadata**

Program exchange metadata is used for exchanging various information related to the program when exchanging the program files itself, and it consists of information regarding the program and essences like video, audio and closed caption etc. which is conveyed in the program files. This chapter defines program exchange metadata description method and the relations between the program exchange metadata and the program file.

#### **6.1.2    Program Exchange Metadata Description Format**

Program exchange metadata, which is defined in this technical report, has two description formats as follows.

- XML description format

In this format, the program exchange metadata is described in XML, and treated as independent document file besides MXF file where program is stored. Program exchange metadata which is represented by XML is called program exchange metadata document.

- DMS-1 description format

In this format, the program exchange metadata is mapped into the DMS-1 which is defined in SMPTE 380M, and conveyed in MXF file where program is stored.

XML description format of the program exchange metadata is defined in section 6.2 to 6.4, and DMS-1 description format of the program exchange metadata is defined in section 6.5.

#### **6.1.2.1    Basic Program Exchange Metadata Description Format**

Basic program exchange metadata description format shall be XML description format which is defined in section 6.1.2, but other description format may be used. Moreover, different description formats can be used simultaneously, but each corresponding element shall match. This technical report does not define priority of unmatched corresponding element.

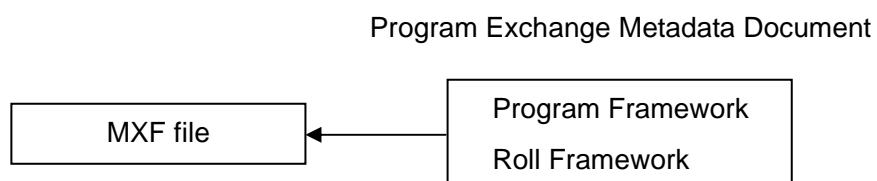
#### **6.1.3    Relations between Program Exchange Metadata Documents and MXF Files**

The relations between program exchange metadata documents which is described in XML, and MXF files which consist programs, is defined in this section. There are several cases for those

relations.

Case 1: A program consists of one MXF file

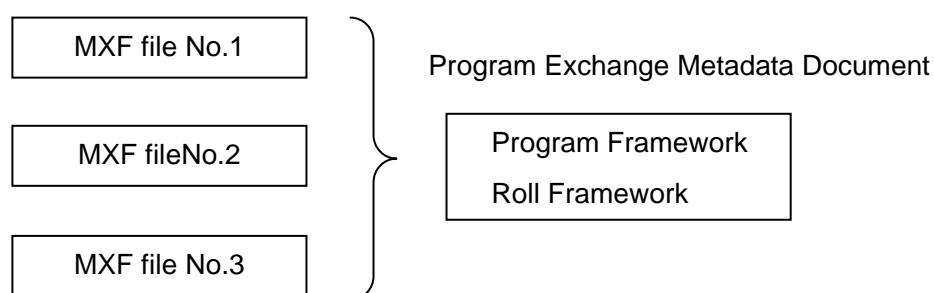
In case a program consists of one MXF file, program exchange metadata document corresponding to the MXF file shall be one file, as shown in figure 6-1. This program exchange metadata document shall consist of Program Framework Metadata which relates to program itself, and Roll Framework Metadata which relates to essences like video/audio/closed caption etc. conveyed in MXF file.



**Figure 6-1 Case a program consists of one MXF file**

Case 2: A program consists of more than one MXF file

In case a program consists of more than one MXF file, program exchange metadata document corresponding to the MXF files shall be one file, as shown in figure 6-2. This program exchange metadata document shall consist of Program Framework Metadata which relates to program itself, and Roll Framework Metadata which relates to essences like video/audio/closed caption etc. conveyed in MXF files.



**Figure 6-2 Case a program consists of more than one MXF file**

Cases other than above refer to section 6.4.4.

## 6.2. Program Exchange Metadata Encoding

### 6.2.1 Description Language

XML is used as a description language in the program exchange metadata document. Media type shall be 'text/xml; charset="UTF-16"'. XML schema recommended by W3C is used as its structural description language for program exchange metadata document.

### 6.2.2 Character Encoding

The character encoding in the program exchange metadata document is UCS (UTF-16). Character sets of UCS and the encoding methods apply to the following standards.

- ISO/IEC 10646-1:2000
- ISO/IEC 10646-1:2000/Amd.1:2002
- ISO/IEC 10646-2:2001

### 6.2.3 Name Space

Name space for the metadata descriptive scheme defined in this technical report, is shown as follows.

```
http://www.arib.or.jp/trb31/schemas/YYYY/PEM
```

Note: "YYYY" indicates the issue year of the metadata description scheme. Refer to the version management section for details.

Metadata descriptive scheme defined in this technical report include metadata descriptive scheme recommended by W3C. Program exchange metadata documents shall use appropriate name space and should declare name space.

Name space prefix shown below is used in this technical report.

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

#### 6.2.4 Data Hierarchy

Metadata of the MXF has hierarchal layered structure consist of Scheme / Framework / Set / Property. DMS-1 is one of metadata scheme which is designed according to this layered structure. According to SMPTE EG42 Annex B, the MXF metadata is classified into the following hierarchies.

- Scheme: Sets of framework, even if each element is essentially independent, it has relations in common class hierarchy and able to share the resources.
- Framework: Sets of related metadata objects (either of Set or Property) as the instance of the provided class hierarchy (entity generated based upon a certain class).
- Set: A set of properties. By collectivizing, it can provide more information including relations between properties compared to just total information of each property.
- Property: Indicates individual metadata item.
- Enumeration: Some properties are defined by values which semantics are allocated. In the property list, numerical data (language dependant), strict term definition (language dependant), or flexible term definition (Depend on language, culture, implementation or an industrial field) can be included. Under a certain limited situation, Set may be brought together in the list by a unique method.

Program exchange metadata is designed as to have layered structure based on SMPTE EG42 Annex B, concerning compatibility with DMS-1, and define two frameworks as follows.

- Program Framework

Framework consists of sets and properties which describe information regarding to program itself.

- Roll Framework

Framework consists of sets and properties which describe information regarding to video / audio / closed caption / etc..

"Roll" in this metadata means the roll described in the chapter 2.

#### 6.2.5 Version Management

Version management of the Program Exchange Metadata description scheme and metadata description is defined as follows.

(1) Version management of the Program Exchange Metadata description scheme

The version of the Program Exchange Metadata description scheme is managed by both



version attribute and namespace.

- The version consists of major version and minor version, and expressed as <major version>.<minor version> format.
- When modifying without backward compatibility, major version shall be incremented, and when modifying with backward compatibility, minor version shall be incremented.
- Version number is indicated in root attribute “version”.
- Issue year (4 characters of the dominical year) shall be included in the part of the namespace, and when using new major version, issue year shall be indicated. (In case of minor version up, issue year of the namespace shall not be changed and only minor version of the version attribute shall be incremented.)

e.g. In case of issuing new major version in 2010.

<http://www.arib.or.jp/trb31/schemas/2010/PEM>

## (2) Version management of metadata description

The version of metadata description is managed by indicating version in the MetadataVersion element.

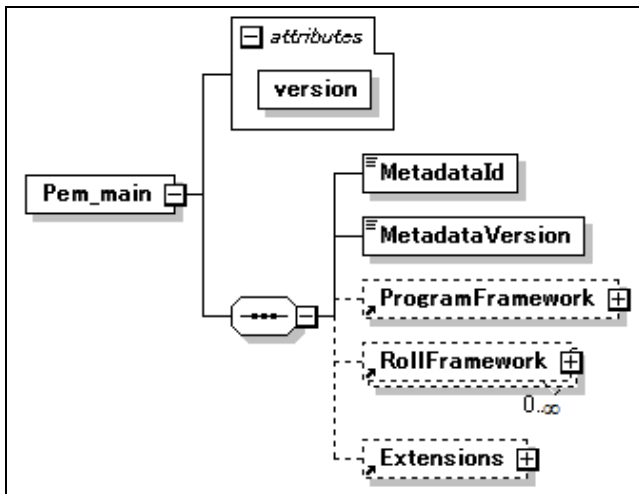
### 6.3. Program Exchange Metadata Description Format

Format, syntax and semantics of the program exchange metadata document are shown below. For syntax, when the number of times element appearance (maxOccurs/ minOccurs) is “1”, the attribute is omitted.

#### 6.3.1 Program Exchange Metadata Document

Program exchange metadata document format is shown as follows.

## ■ Diagram:



## ■ Syntax:

```

<xs:element name="Pem_main">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MetadataId" type="xs:string"/>
      <xs:element name="MetadataVersion" type="xs:string"/>
      <xs:element ref="ProgramFramework" minOccurs="0"/>
      <xs:element ref="RollFramework" maxOccurs="unbounded" minOccurs="0"/>
      <xs:element ref="Extensions" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="version" type="xs:string" use="required" fixed="1.1"/>
  </xs:complexType>
</xs:element>

```

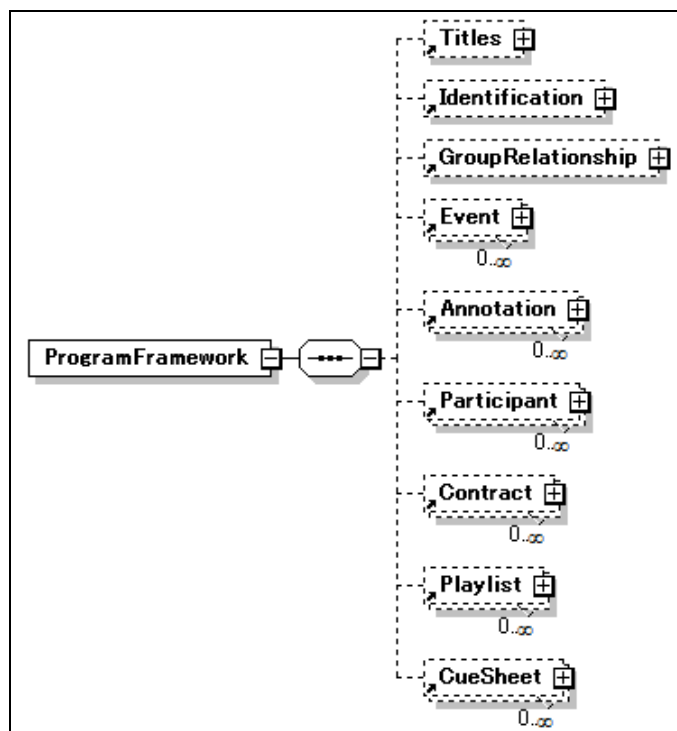
## ■ Semantics

Names	Definitions
Pem_main	It is a root element for appropriate instance document of the schema.
MetadataId	It is an ID for the program exchange metadata document.
MetadataVersion	It is a version number of metadata description.
ProgramFramework	It is a program framework.
RollFramework	It is a roll framework.
Extensions	It is an area for extension information.
@version	Version of metadata description scheme

## 6.3.2 Details of Framework

### 6.3.2.1 Program Framework

■ Diagram:



■ Syntax:

```

<xs:element name="ProgramFramework">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Titles" minOccurs="0"/>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element ref="GroupRelationship" minOccurs="0"/>
      <xs:element ref="Event" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="Participant" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="Contract" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="Playlist" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="CueSheet" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

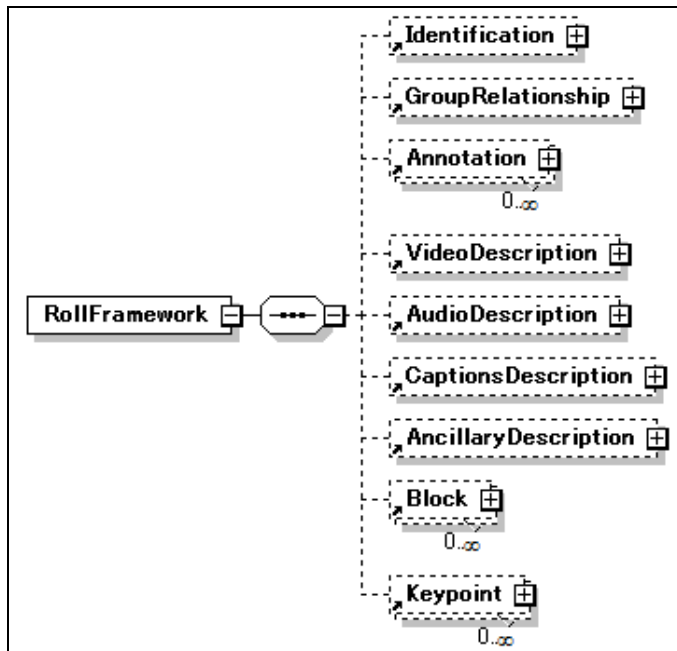
```

## ■ Semantics:

Names	Definitions
Titles	Titles (set) element
Identification	Identification (set) element (e.g. ID for a program)
GroupRelationship	GroupRelationship (set) element
Event	Event (set) element
Annotation	Annotation (set) element
Participant	Participant (set) element
Contract	Contract (set) element
Playlist	Playlist (set) element
CueSheet	CueSheet (set) element

### 6.3.2.2 Roll Framework

■ Diagram:



■ Syntax:

```

<xs:element name="RollFramework">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element ref="GroupRelationship" minOccurs="0"/>
      <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="VideoDescription" minOccurs="0"/>
      <xs:element ref="AudioDescription" minOccurs="0"/>
      <xs:element ref="CaptionsDescription" minOccurs="0"/>
      <xs:element ref="AncillaryDescription" minOccurs="0"/>
      <xs:element ref="Block" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element ref="Keypoint" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

■ Semantics:

Names	Definitions
Identification	Identification (set) element (e.g. ID for a roll)
GroupRelationship	GroupRelationship (set) element
Annotation	Annotation (set) element
VideoDescription	VideoDescription (set) element
AudioDescription	AudioDescription (set) element
CaptionsDescription	CaptionsDescription (set) element
AncillaryDescription	AncillaryDescription (set) element
Block	Block (set) element
Keypoint	Keypoint (set) element

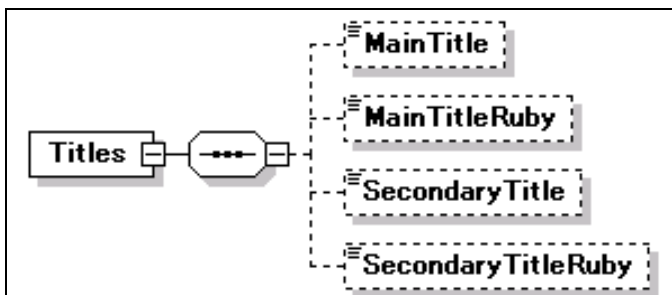
### 6.3.3 Details of Sets

Syntax and semantics of each set are shown below.

#### 6.3.3.1 Titles

Titles (set) element is an element for describing program title information.

■ Diagram:



■ Syntax:

```

<xs:element name="Titles">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MainTitle" type="xs:string" minOccurs="0"/>
      <xs:element name="MainTitleRuby" type="xs:string" minOccurs="0"/>
      <xs:element name="SecondaryTitle" type="xs:string" minOccurs="0"/>
      <xs:element name="SecondaryTitleRuby" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

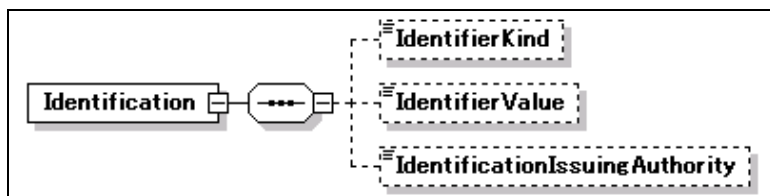
■ Semantics:

Names	Definitions
Titles	Title (set) element
MainTitle	It is a main title.
MainTitleRuby	It is a ruby for the main title.
SecondaryTitle	It is a secondary title.
SecondaryTitleRuby	It is a ruby for the secondary title.

### 6.3.3.2 Identification

Identification (set) element is an element for describing identification information regarding to the program or files.

■ Diagram:



■ Syntax:

```

<xs:element name="Identification">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="IdentifierKind" type="xs:string" minOccurs="0"/>
      <xs:element name="IdentifierValue" type="xs:string" minOccurs="0"/>
      <xs:element name="IdentificationIssuingAuthority" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

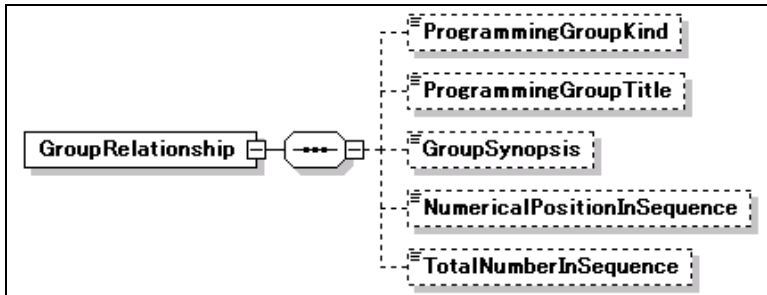
■ Semantics:

Names	Definitions
Identification	Identification (set) element
IdentifierKind	It is a kind of identifier. (or, authentication system)
IdentifierValue	It is a value for identifier.
IdentificationIssuingAuthority	It is an identification issuing authority name.

### 6.3.3.3 GroupRelationship

GroupRelationship (set) element is an element for describing group information of the program or the files.

■ Diagram:



■ Syntax:

```
<xs:element name="GroupRelationship">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProgrammingGroupKind" type="xs:string" minOccurs="0"/>
      <xs:element name="ProgrammingGroupTitle" type="xs:string" minOccurs="0"/>
      <xs:element name="GroupSynopsis" type="xs:string" minOccurs="0"/>
      <xs:element name="NumericalPositionInSequence" type="xs:unsignedInt" minOccurs="0"/>
      <xs:element name="TotalNumberInSequence" type="xs:unsignedInt" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

■ Semantics:

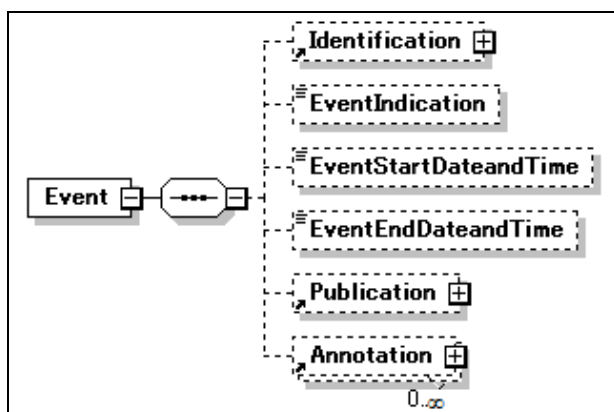
Names	Definitions
GroupRelationship	GroupRelationship (set) element
ProgrammingGroupKind	It is a kind for the group. (e.g. season, episode, series, collection of particular theme, etc.)
ProgrammingGroupTitle	It is a title of the group
GroupSynopsis	It is a synopsis of the group
NumericalPositionInSequence	It is a sequence number in the group.
TotalNumberInSequence	It is a total number of the group.



### 6.3.3.4 Event

Event (set) element is an element for defining event e.g. broadcasting date and time, or end of the broadcasting date and time etc.

■ Diagram:



■ Syntax:

```

<xs:element name="Event">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element name="EventIndication" type="xs:string" minOccurs="0"/>
      <xs:element name="EventStartDateandTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="EventEndDateandTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element ref="Publication" minOccurs="0"/>
      <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

■ Semantics:

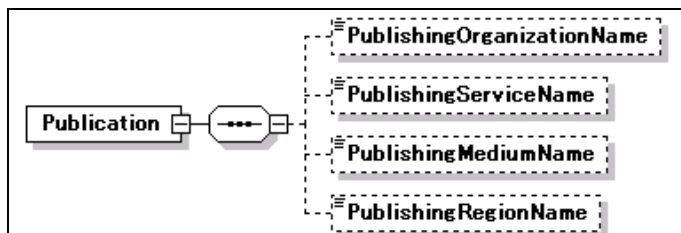
Names	Definitions
Event	Event (set) element
Identification	Identification (set) element
EventIndication	It is a kind of event. (e.g. broadcasting date and time, history, etc.)
EventStartDateandTime	It is a start date and time of the event.
EventEndDateandTime	It is an end date and time of the event.
Publication	Publication (set) element
Annotation	Annotation (set) element



### 6.3.3.5 Publication

Publication (set) element is an element for describing medium or service of the event publication.

■ Diagram:



■ Syntax:

```

<xs:element name="Publication">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PublishingOrganizationName" type="xs:string" minOccurs="0"/>
      <xs:element name="PublishingServiceName" type="xs:string" minOccurs="0"/>
      <xs:element name="PublishingMediumName" type="xs:string" minOccurs="0"/>
      <xs:element name="PublishingRegionName" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

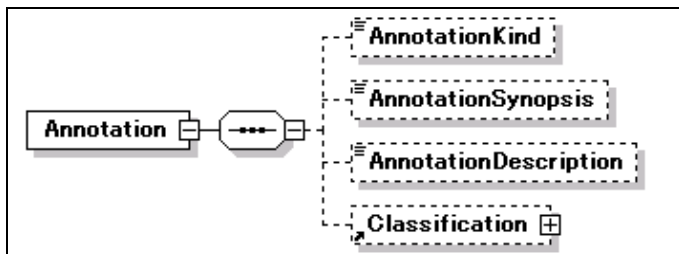
■ Semantics:

Names	Definitions
Publication	Publication (set) element
PublishingOrganizationName	It is a name of the broadcaster.
PublishingServiceName	It is a name of the service.
PublishingMediumName	It is a name of medium. (e.g. terrestrial, satellite, cable)
PublishingRegionName	It is a region that covers.

### 6.3.3.6 Annotation

Annotation (set) element is an element for describing annotation or synopsis.

■ Diagram:



■ Syntax:

```
<xs:element name="Annotation">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AnnotationKind" type="xs:string" minOccurs="0"/>
      <xs:element name="AnnotationSynopsis" type="xs:string" minOccurs="0"/>
      <xs:element name="AnnotationDescription" type="xs:string" minOccurs="0"/>
      <xs:element ref="Classification" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

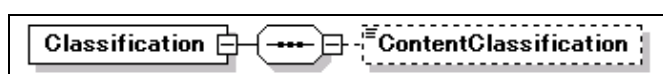
■ Semantics:

Names	Definitions
Annotation	Annotation (set) element
AnnotationKind	It is a kind of annotation.
AnnotationSynopsis	It is a synopsis.
AnnotationDescription	It is a free word description.
Classification	Classification (set) element

### 6.3.3.7 Classification

Classification (set) element is an element for describing classification of program or video/audio files.

■ Diagram:



■ Syntax:

```
<xs:element name="Classification">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ContentClassification" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

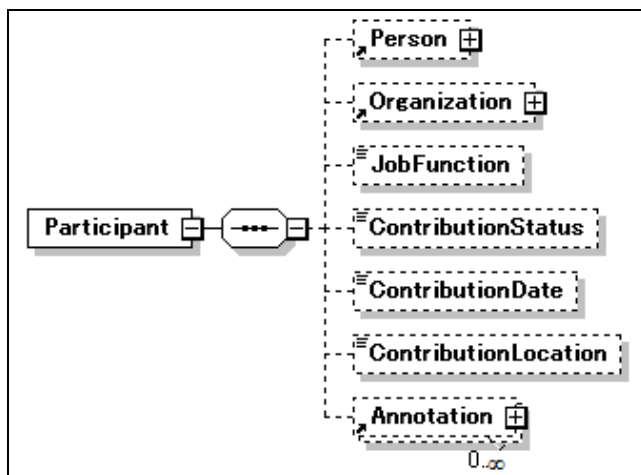
■ Semantics:

Names	Definitions
Classification	Classification (set) element
ContentClassification	Classification of the object. (e.g. Program genre)

### 6.3.3.8 Participant

Participant (set) element is an element for describing contribution status of a person or an organization.

■ Diagram:



### ■ Syntax:

```
<xs:element name="Participant">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Person" minOccurs="0"/>
      <xs:element ref="Organization" minOccurs="0"/>
      <xs:element name="JobFunction" type="xs:string" minOccurs="0"/>
      <xs:element name="ContributionStatus" type="xs:string" minOccurs="0"/>
      <xs:element name="ContributionDate" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="ContributionLocation" type="xs:string" minOccurs="0"/>
      <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

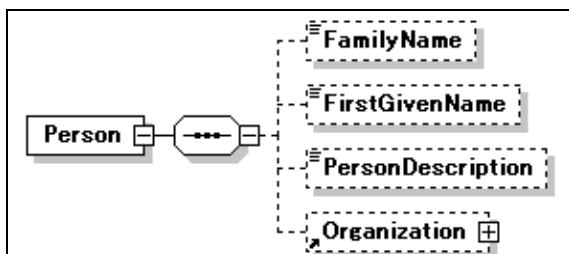
### ■ Semantics:

Names	Definitions
Participant	Participant (set) element
Person	Person (set) element
Organization	Organization (set) element
JobFunction	It is a job function of a person, an organization etc.
ContributionStatus	It is a contribution status of a person or an organization.
ContributionDate	It is a date a person or an organization contributed.
ContributionLocation	It is a place where a person or an organization contributed.
Annotation	Annotation (set) element

#### 6.3.3.9 Person

Person (set) element is an element for describing specific person.

### ■ Diagram:



■ Syntax:

```
<xs:element name="Person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="FamilyName" type="xs:string" minOccurs="0"/>
      <xs:element name="FirstGivenName" type="xs:string" minOccurs="0"/>
      <xs:element name="PersonDescription" type="xs:string" minOccurs="0"/>
      <xs:element ref="Organization" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

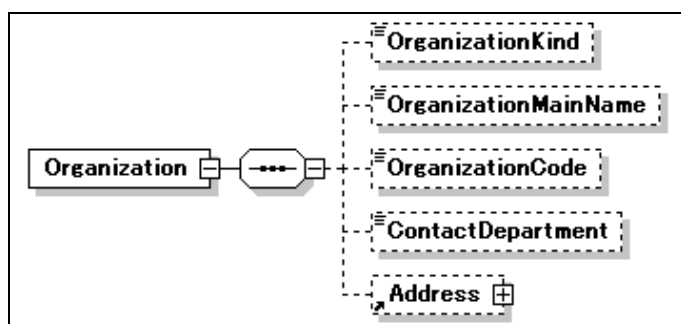
■ Semantics:

Names	Definitions
Person	Person (set) element
FamilyName	It is a family name of the person.
FirstGivenName	It is a given name of the person.
PersonDescription	It is a Free word description of the person.
Organization	Organization (set) element

### 6.3.3.10 Organization

Organization (set) element is an element for describing an organization.

■ Diagram:



### ■ Syntax:

```
<xs:element name="Organization">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="OrganizationKind" type="xs:string" minOccurs="0"/>
      <xs:element name="OrganizationMainName" type="xs:string" minOccurs="0"/>
      <xs:element name="OrganizationCode" type="xs:string" minOccurs="0"/>
      <xs:element name="ContactDepartment" type="xs:string" minOccurs="0"/>
      <xs:element ref="Address" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

### ■ Semantics:

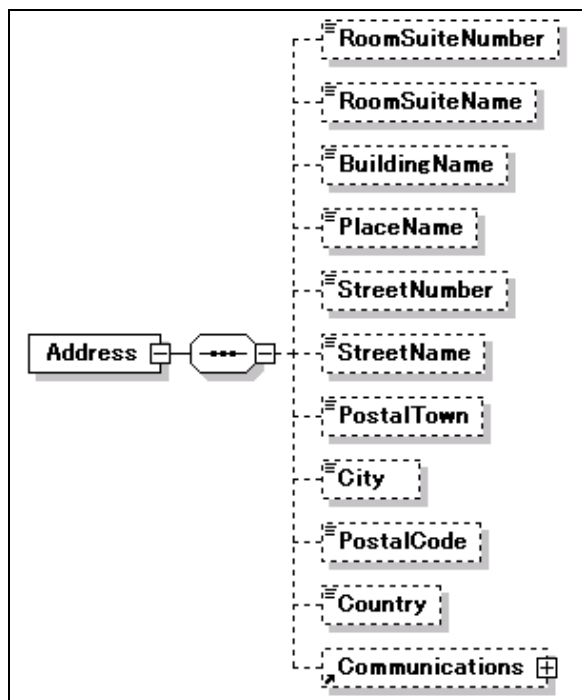
Names	Definitions
Organization	Organization (set) element
OrganizationKind	It is a kind of the organization.
OrganizationMainName	It is a main name of the organization.
OrganizationCode	It is an identification code of the organization.
ContactDepartment	It is a contact department of the organization.
Address	Address (set) element



### 6.3.3.11 Address

Address (set) element is an element for describing addresses.

■ Diagram:



■ Syntax:

```

<xs:element name="Address">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RoomSuiteNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="RoomSuiteName" type="xs:string" minOccurs="0"/>
      <xs:element name="BuildingName" type="xs:string" minOccurs="0"/>
      <xs:element name="PlaceName" type="xs:string" minOccurs="0"/>
      <xs:element name="StreetNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="StreetName" type="xs:string" minOccurs="0"/>
      <xs:element name="PostalTown" type="xs:string" minOccurs="0"/>
      <xs:element name="City" type="xs:string" minOccurs="0"/>
      <xs:element name="PostalCode" type="xs:string" minOccurs="0"/>
      <xs:element name="Country" type="xs:string" minOccurs="0"/>
      <xs:element ref="Communications" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
  
```

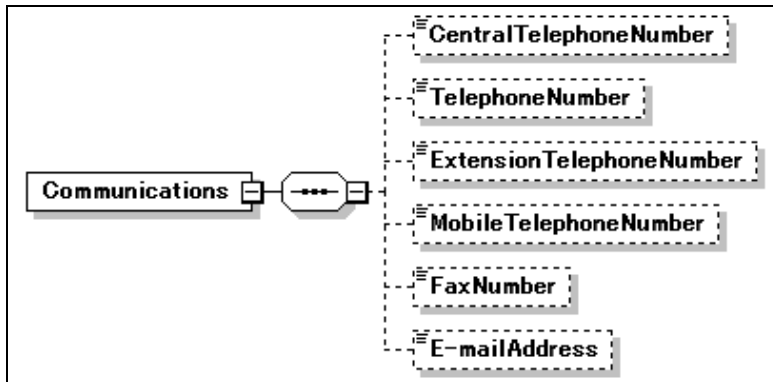
■ Semantics:

Names	Definitions
Address	Address (set) element
RoomSuiteNumber	Room number.
RoomSuiteName	Name of apartment.
BuildingName	Name of building.
PlaceName	Name of place.
StreetNumber	Street number.
StreetName	Street name.
PostalTown	Name of (postal) town.
City	Name of the city.
PostalCode	Postal code.
Country	Name of the country.
Communications	Communications (set) element

### 6.3.3.12 Communications

Communications (set) element is an element for describing means of communication.

■ Diagram:



### ■ Syntax:

```
<xs:element name="Communications">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CentralTelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="TelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="ExtensionTelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="MobileTelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="FaxNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="E-mailAddress" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

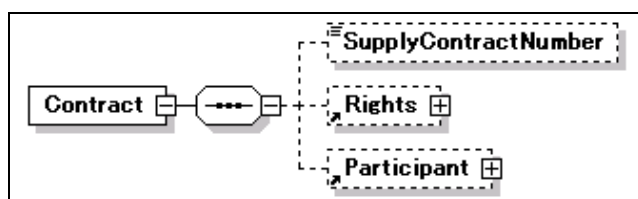
### ■ Semantics:

Names	Definitions
Communications	Communications (set) element
CentralTelephoneNumber	Central Telephone Number
TelephoneNumber	Telephone Number
ExtensionTelephoneNumber	Extension Telephone Number
MobileTelephoneNumber	Mobile Telephone Number
FaxNumber	FAX Number
E-mailAddress	E-mail Address

### 6.3.3.13 Contract

Contract element is an element for describing contract of a program or a material.

### ■ Diagram:



■ Syntax:

```
<xs:element name="Contract">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SupplyContractNumber" type="xs:string" minOccurs="0"/>
      <xs:element ref="Rights" minOccurs="0"/>
      <xs:element ref="Participant" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

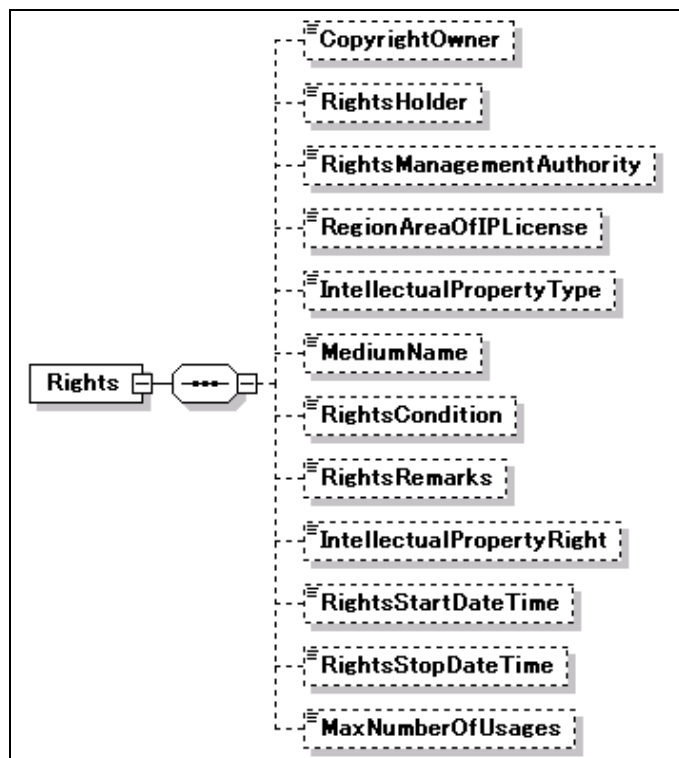
■ Semantics:

Names	Definitions
Contract	Contract (set) element
SupplyContractNumber	It is a contract number of supplier
Rights	Rights (set) element
Participant	Participant (set) element

### 6.3.3.14 Rights

Rights (set) element is an element for describing rights regarding to program and video/audio file.

■ Diagram:



■ Syntax:

```
<xs:element name="Rights">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CopyrightOwner" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsHolder" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsManagementAuthority" type="xs:string" minOccurs="0"/>
      <xs:element name="RegionAreaOfIPLicense" type="xs:string" minOccurs="0"/>
      <xs:element name="IntellectualPropertyType" type="xs:string" minOccurs="0"/>
      <xs:element name="MediumName" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsCondition" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsRemarks" type="xs:string" minOccurs="0"/>
      <xs:element name="IntellectualPropertyRight" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsStartDateTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="RightsStopDateTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="MaxNumberOfUsages" type="xs:unsignedShort" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

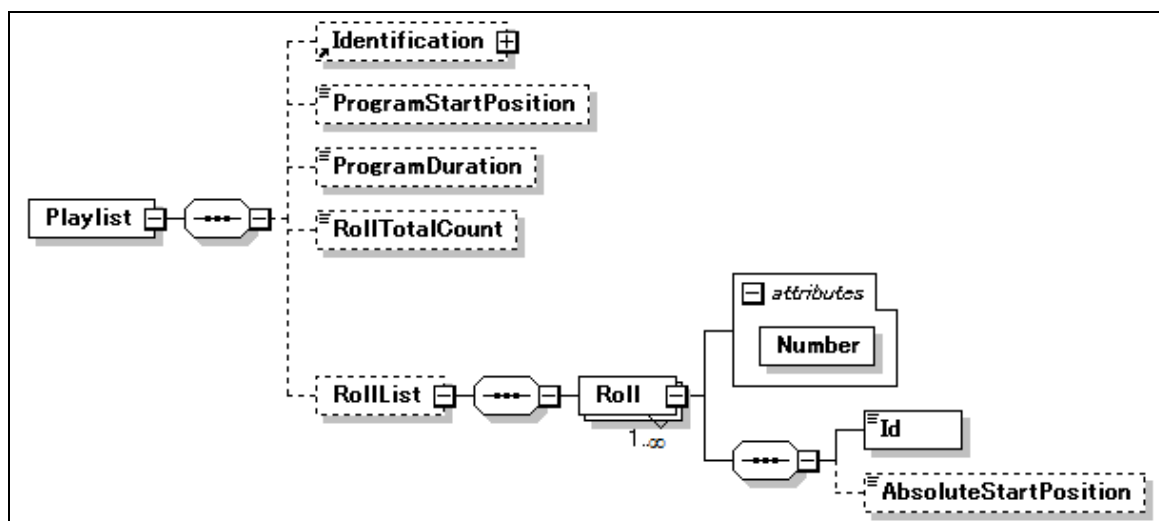
■ Semantics:

Names	Definitions
Rights	Rights (set) element
CopyrightOwner	Person or organization owns the copyright
RightsHolder	Person or organization that can exercise intellectual property rights
RightsManagementAuthority	Rights management authority name
RegionAreaOfIPLicense	Region area of intellectual property right
IntellectualPropertyType	Definition concerning intellectual property right
MediumName	Target medium for rights (e.g. terrestrial, satellite)
RightsCondition	Conditions that rights are limited
RightsRemarks	General remarks regarding rights
IntellectualPropertyRight	Free word description regarding intellectual property right
RightsStartDateTime	Date and time rights start
RightsStopDateTime	Date and time rights ends
MaxNumberOfUsages	Maximum number of usage

### 6.3.3.15 Playlist

Playlist (set) element is an element for describing absolute timing of program when constructing from multiple rolls.

■ Diagram:



### ■ Syntax:

```

<xs:element name="Playlist">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element name="ProgramStartPosition" type="TimecodeType" minOccurs="0"/>
      <xs:element name="ProgramDuration" type="TimecodeType" minOccurs="0"/>
      <xs:element name="RollTotalCount" type="xs:positiveInteger" minOccurs="0"/>
      <xs:element name="RollList" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="Roll" maxOccurs="unbounded">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Id" type="xs:string"/>
                  <xs:element name="AbsoluteStartPosition" type="TimecodeType" minOccurs="0"/>
                </xs:sequence>
                <xs:attribute name="Number" type="xs:unsignedShort" use="required"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

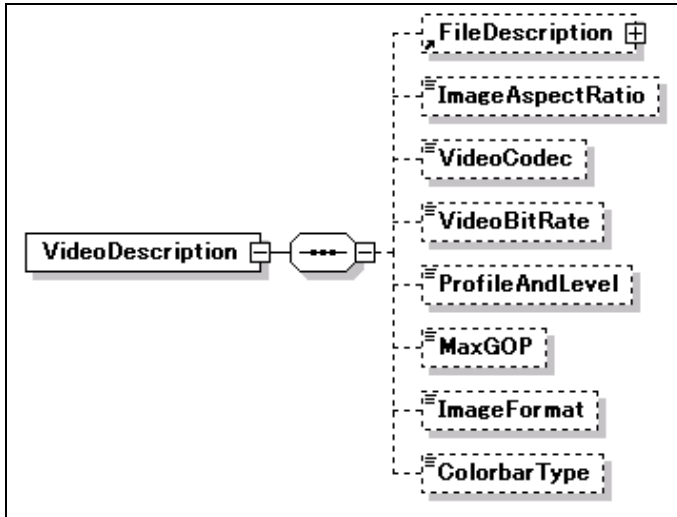
### ■ Semantics:

Names	Definitions
Playlist	Playlist (set) element
Identification	Identification (set) element
ProgramStartPosition	Program start time
ProgramDuration	Program duration
RollTotalCount	Total count of rolls
RollList	Roll list
Roll	Roll information
Id	ID number of the roll
AbsoluteStartPosition	Absolute start position of the roll
@Number	Roll number attribute

### 6.3.3.16 VideoDescription

VideoDescription (set) element is an element for describing video files and video essences.

■ Diagram:



■ Syntax:

```

<xs:element name="VideoDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="FileDescription" minOccurs="0"/>
      <xs:element name="ImageAspectRatio" type="xs:string" minOccurs="0"/>
      <xs:element name="VideoCodec" type="xs:string" minOccurs="0"/>
      <xs:element name="VideoBitRate" type="xs:string" minOccurs="0"/>
      <xs:element name="ProfileAndLevel" type="xs:string" minOccurs="0"/>
      <xs:element name="MaxGOP" type="xs:unsignedShort" default="0" minOccurs="0"/>
      <xs:element name="ImageFormat" type="xs:string" minOccurs="0"/>
      <xs:element name="ColorbarType" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
  
```



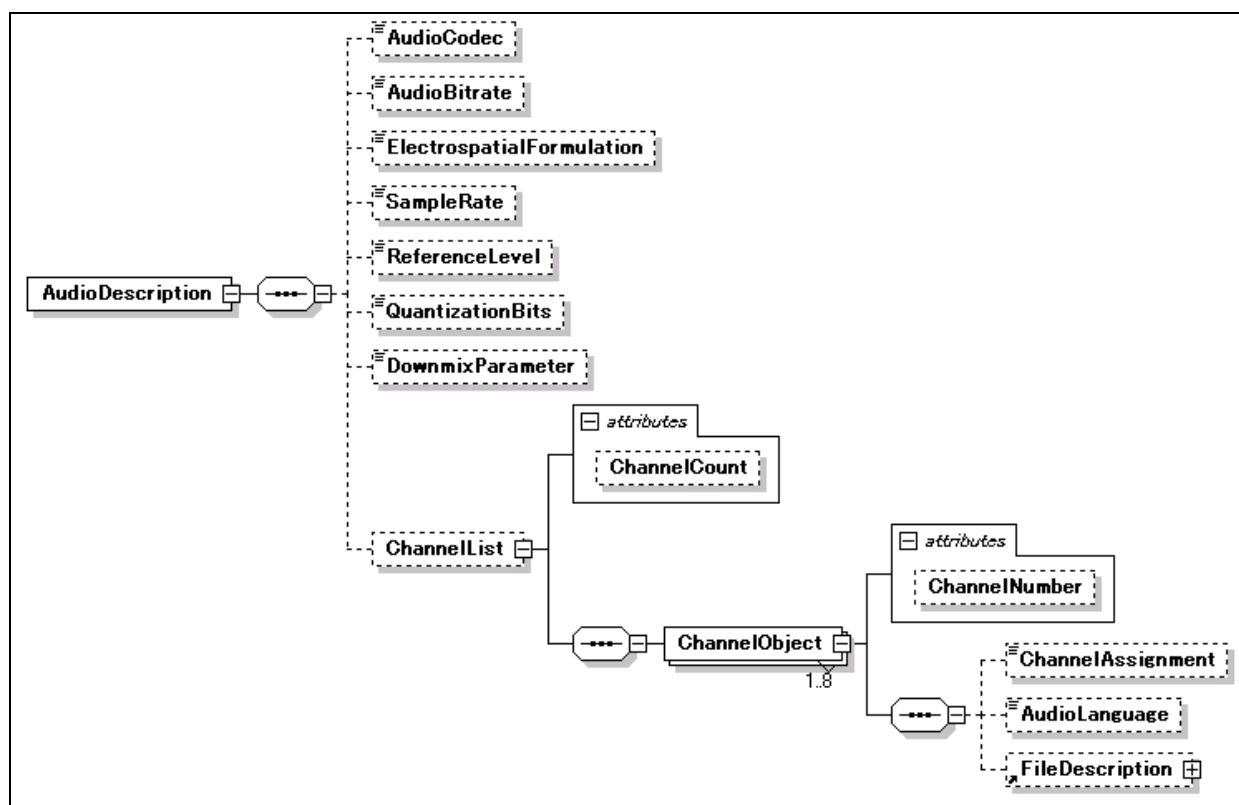
■ Semantics:

Names	Definitions
VideoDescription	VideoDescription (set) element
FileDescription	FileDescription (set) element
ImageAspectRatio	Image aspect ratio of video essence
VideoCodec	Codec used for video essence.
VideoBitRate	Bitrates for video essence.
ProfileAndLevel	Profile and level for video essence.
MaxGOP	Maximum GOP number of video essence.
ImageFormat	Imagery format of video essence.
ColorbarType	Colorbar type used in video essence.

### 6.3.3.17 AudioDescription

AudioDescription (set) element is an element for describing audio files and audio essences.

■ Diagram:



■ Syntax:

```

<xs:element name="AudioDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioCodec" type="xs:string" minOccurs="0"/>
      <xs:element name="AudioBitrate" type="xs:string" minOccurs="0"/>
      <xs:element name="ElectrospatialFormulation" type="xs:string" minOccurs="0"/>
      <xs:element name="SampleRate" type="xs:string" minOccurs="0"/>
      <xs:element name="ReferenceLevel" type="xs:string" minOccurs="0"/>
      <xs:element name="QuantizationBits" type="xs:unsignedShort" minOccurs="0"/>
      <xs:element name="DownmixParameter" type="xs:string" minOccurs="0"/>
      <xs:element name="ChannelList" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="ChannelObject" maxOccurs="8">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="ChannelAssignment" type="xs:string" minOccurs="0"/>
                  <xs:element name="AudioLanguage" type="xs:string" minOccurs="0"/>
                  <xs:element ref="FileDescription" minOccurs="0"/>
                </xs:sequence>
                <xs:attribute name="ChannelNumber" type="xs:unsignedShort"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
          <xs:attribute name="ChannelCount" type="xs:unsignedShort"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

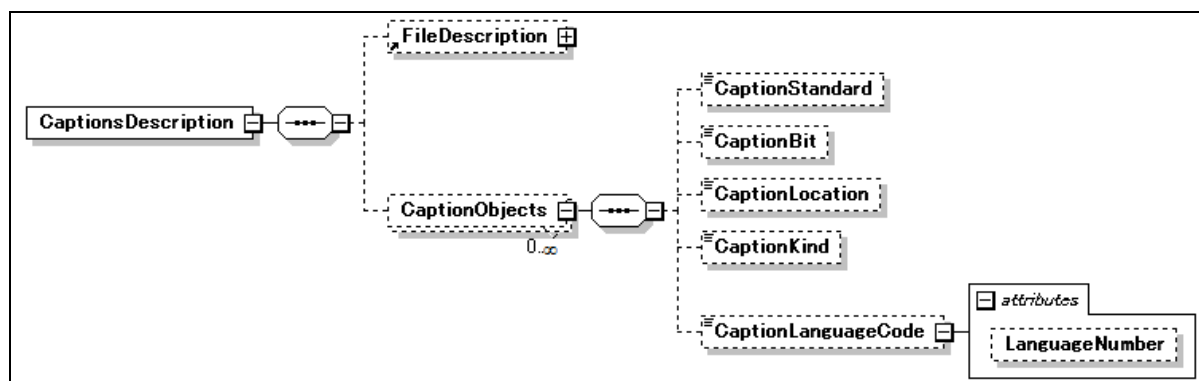
■ Semantics:

Names	Definitions
AudioDescription	AudioDescription(set)element
AudioCodec	Codec used for audio essence
AudioBitrate	Bitrates of audio essence
ElectrospatialFormulation	Audio channel division of audio essence
SampleRate	Sample rate of audio essence
ReferenceLevel	Reference level of audio essence
QuantizationBits	Quantization bits number of audio essence
DownmixParameter	Downmix parameter of audio essence
ChannelList	Channel list of each audio essence
@ChannelCount	Total count attribute of audio essence
ChannelObject	Channel object of each audio essence
@ChannelNumber	Channel number attribute of each audio channel
ChannelAssignment	Audio channel assignment
AudioLanguage	Language of the audio channel
FileDescription	FileDescription (set) element

### 6.3.3.18 CaptionsDescription

CaptionsDescription set is a set for describing closed captions.

■ Diagram:



## ■ Syntax:

```

<xs:element name="CaptionsDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="FileDescription" minOccurs="0"/>
      <xs:element name="CaptionObjects" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="CaptionStandard" minOccurs="0"/>
            <xs:element name="CaptionBit" minOccurs="0">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="8bit"/>
                  <xs:enumeration value="10bit"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
            <xs:element name="CaptionLocation" minOccurs="0"/>
            <xs:element name="CaptionKind" minOccurs="0"/>
            <xs:element name="CaptionLanguageCode" minOccurs="0">
              <xs:complexType>
                <xs:simpleContent>
                  <xs:extension base="xs:string">
                    <xs:attribute name="LanguageNumber">
                      <xs:simpleType>
                        <xs:restriction base="xs:integer">
                          <xs:pattern value="[1-8]"/>
                        </xs:restriction>
                      </xs:simpleType>
                    </xs:attribute>
                  </xs:extension>
                </xs:simpleContent>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

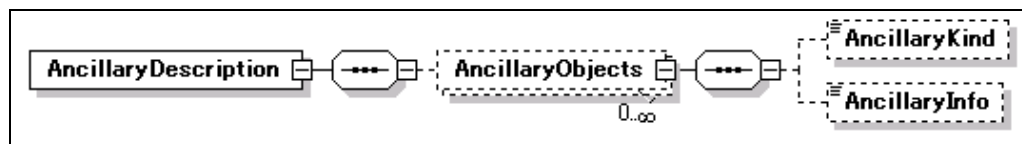
■ Semantics:

Names	Definitions
CaptionsDescription	CaptionsDescription (set) element
FileDescription	FileDescription (set ) element
CaptionObjects	List of caption information
CaptionStandard	Caption inserting standard (e.g. SMPTE 436M, SMPTE 410 etc.)
CaptionBit	Caption bit width (in case of SMPTE 436M) (e.g. 8 bit, 10 bit)
CaptionLocation	Caption location in ancillary data area (e.g. caption ancillary area No.1,2,3)
CaptionKind	Caption Kind (e.g. HD caption, SD caption, mobile caption etc.)
CaptionLanguageCode	Caption language code
@ LanguageNumber	Language count attribute from No.1 through No.8

### 6.3.3.19 AncillaryDescription

AncillaryDescription (set) element is an element for describing ancillary data.

■ Diagram:



■ Syntax:

```
<xs:element name="AncillaryDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AncillaryObjects" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="AncillaryKind" type="xs:string" minOccurs="0"/>
            <xs:element name="AncillaryInfo" type="xs:string" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

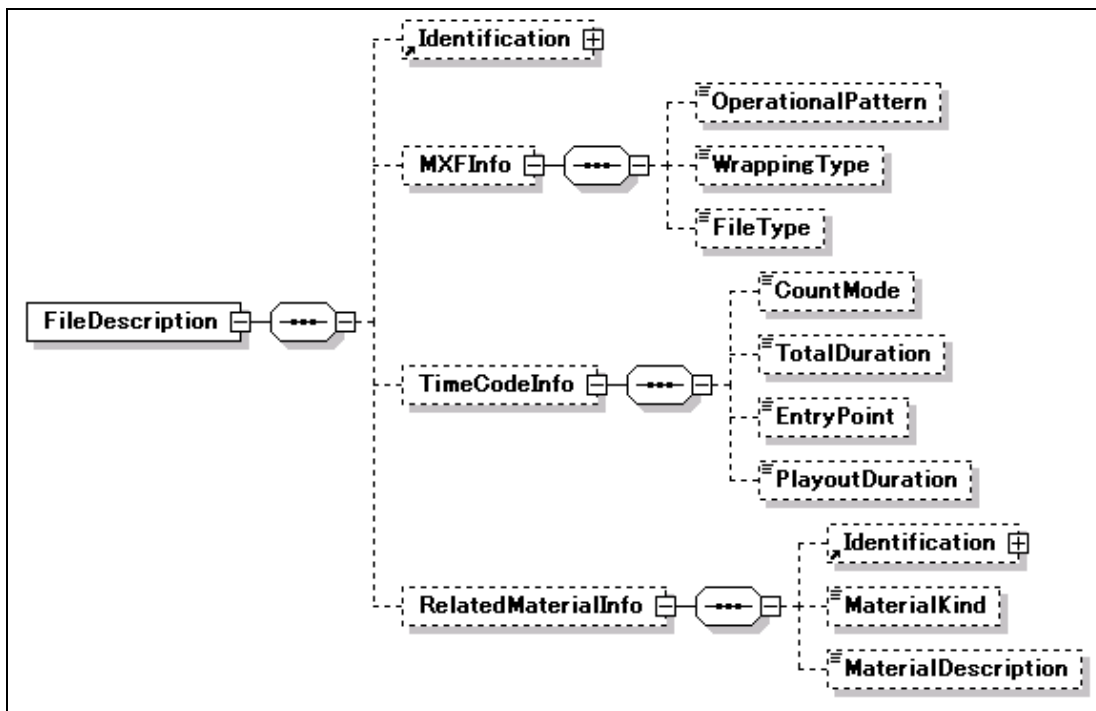
■ Semantics:

Names	Definitions
AncillaryDescription	AncillaryDescription (set) element
AncillaryObjects	List of ancillary data information
AncillaryKind	Kind of ancillary data (e.g. Inter-broadcaster control signal, Datacasting trigger signal etc.)
AncillaryInfo	Ancillary data information

### 6.3.3.20 FileDescription

FileDescription (set) element is an element for describing various information regarding files.

■ Diagram:



■ Syntax:

```

<xs:element name="FileDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element name="MXFInfo" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="OperationalPattern" type="xs:string" minOccurs="0"/>
            <xs:element name="WrappingType" type="xs:string" minOccurs="0"/>
            <xs:element name="FileType" type="xs:string" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="TimeCodeInfo" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="CountMode" type="xs:string" minOccurs="0"/>
            <xs:element name="TotalDuration" type="TimecodeType" minOccurs="0"/>
            <xs:element name="EntryPoint" minOccurs="0">
              <xs:simpleType>
                <xs:union memberTypes="xs:long TimecodeType"/>
              </xs:simpleType>
            </xs:element>
            <xs:element name="PlayoutDuration" type="TimecodeType" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="RelatedMaterialInfo" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="Identification" minOccurs="0"/>
            <xs:element name="MaterialKind" type="xs:string" minOccurs="0"/>
            <xs:element name="MaterialDescription" type="xs:string" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

## ■ Semantics:

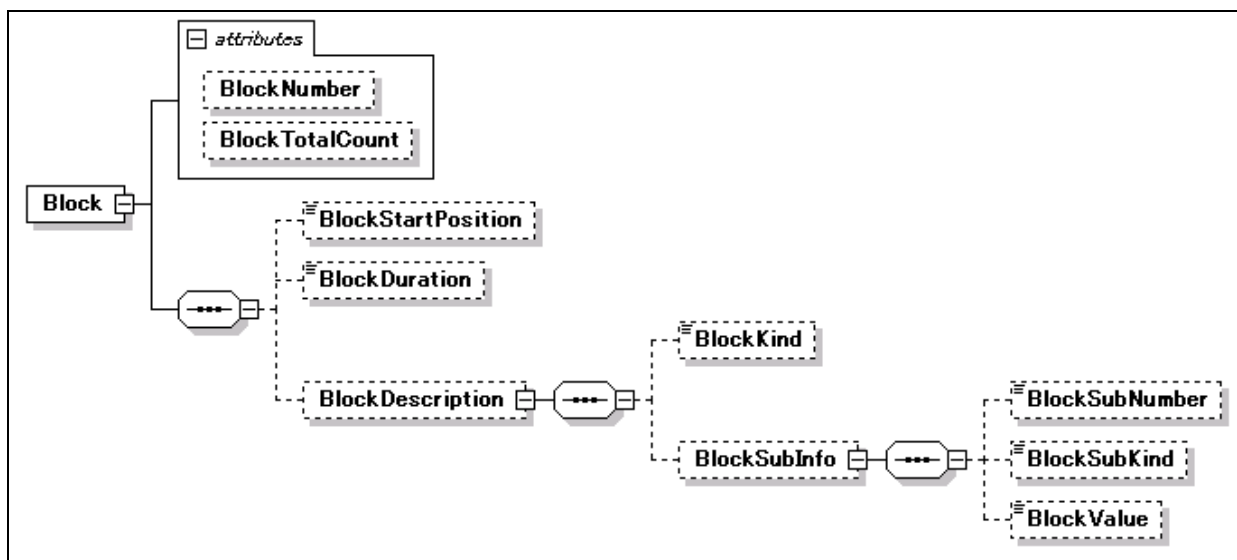
Names	Definitions
FileDescription	FileDescription (set) element
Identification	Identification (set) element (ID of the objective file)
MXFInfo	MXF information
OperationalPattern	Operational pattern
WrappingType	Wrapping type (e.g. FrameWrapping/ ClipWrapping)
FileType	File type (e.g. Video, Audio, Video+Audio, Video+Audio+Caption etc.)
TimeCodeInfo	Timecode information
CountMode	Timecode count mode (e.g. DF/NDF)
TotalDuration	Total duration
EntryPoint	Timecode entry point of the asset
PlayoutDuration	Playout duration of the asset
RelatedMaterialInfo	Information regarding related material (e.g. Original tape information etc.)
Identification	Identification element (e.g. ID of the related material)
MaterialKind	Kind of related material
MaterialDescription	Free word description of related material



### 6.3.3.21 Block

Block (set) element is an element for describing information regarding a certain consecutive video and audio essence block.

■ Diagram:



■ Syntax:

```

<xs:element name="Block">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="BlockStartPosition" type="TimecodeType" minOccurs="0"/>
      <xs:element name="BlockDuration" type="TimecodeType" minOccurs="0"/>
      <xs:element name="BlockDescription" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="BlockKind" type="xs:string" minOccurs="0"/>
            <xs:element name="BlockSubInfo" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="BlockSubNumber" type="xs:unsignedShort" minOccurs="0"/>
                  <xs:element name="BlockSubKind" type="xs:string" minOccurs="0"/>
                  <xs:element name="BlockValue" type="xs:string" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
  <xs:attribute name="BlockNumber" type="xs:unsignedShort"/>
  <xs:attribute name="BlockTotalCount" type="xs:positiveInteger"/>
</xs:complexType>
</xs:element>

```

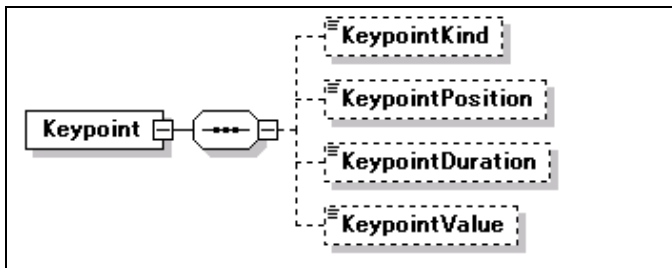
■ Semantics:

Names	Definitions
Block	Block (set) element
BlockStartPosition	Block start position
BlockDuration	Block duration
BlockDescription	Block description
BlockKind	Kind of the block
BlockSubInfo	Additional information for the block
BlockSubNumber	Block sub number (Number in block set of same kind. e.g. BlockKind = Main roll, BlockSubKind = Main roll No.1)
BlockSubKind	Secondary kind of the block (More information regarding block kind)
BlockValue	Value of the BlockSubKind
@BlockNumber	Block number attribute (Sequence number of total blocks)
@BlockTotalCount	Total block number

### 6.3.3.22 Keypoint

Keypoint (set) element is an element for describing features in a certain section of the essence.

■ Diagram:



### ■ Syntax:

```
<xs:element name="Keypoint">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="KeypointKind" type="xs:string" minOccurs="0"/>
      <xs:element name="KeypointPosition" type="TimecodeType" minOccurs="0"/>
      <xs:element name="KeypointDuration" type="TimecodeType" minOccurs="0"/>
      <xs:element name="KeypointValue" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

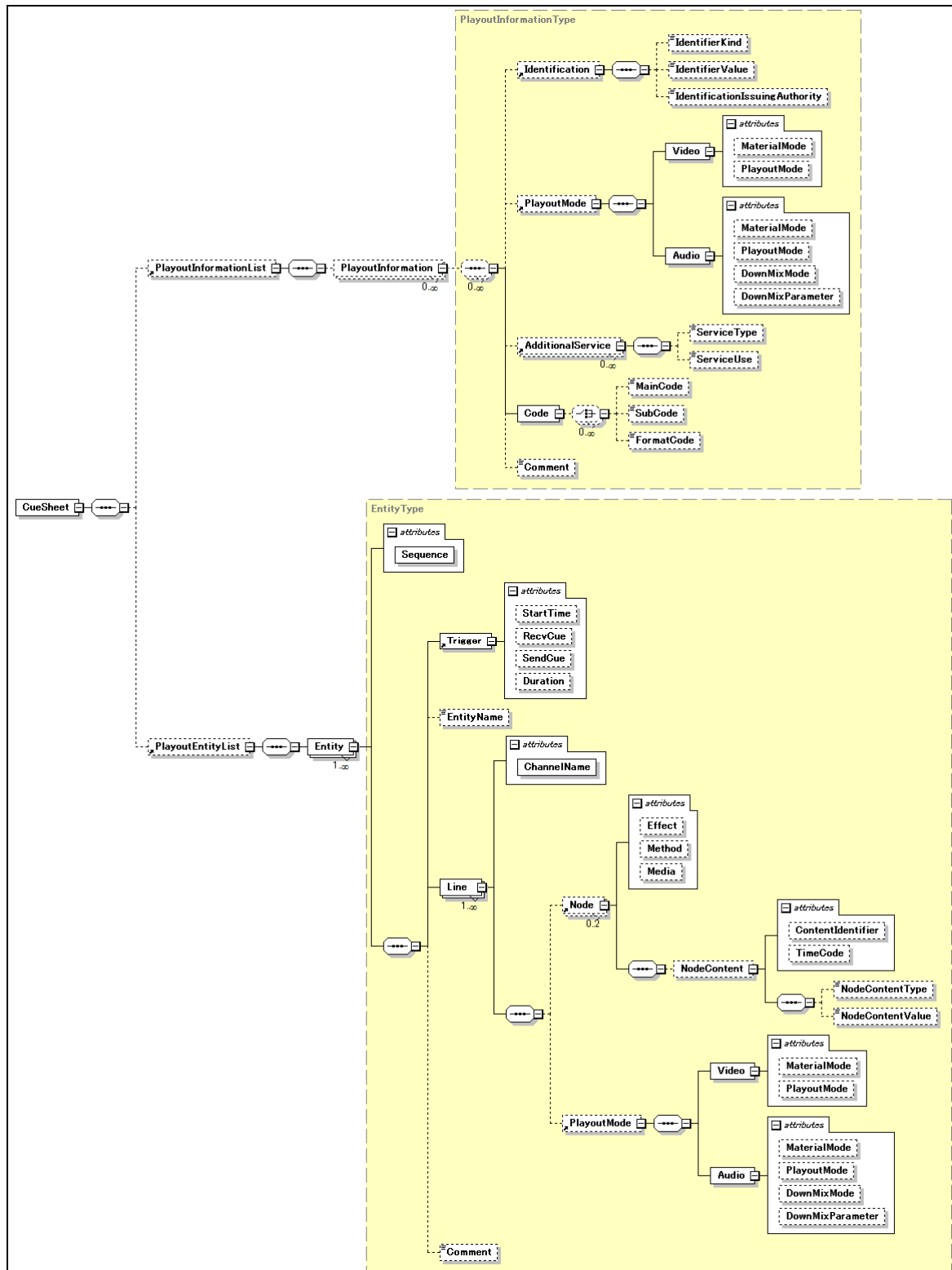
### ■ Semantics:

Names	Definitions
Keypoint	Keypoint (set) element
KeypointKind	It is a kind of keypoint (e.g. superimpose, flash effect, shot category, keyword, key video, key audio)
KeypointPosition	Keypoint timecode value of the position where event specified by KeypointKind occurred
KeypointDuration	Keypoint duration where event specified by KeypointKind occurred
KeypointValue	Description of keypoint

### 6.3.3.23 CueSheet

CueSheet (set) element is an element for describing cue sheet.

■ Diagram:



■ Syntax:

```

<xs:element name="CueSheet">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="PayoutInformationList" minOccurs="0"/>
      <xs:element ref="PayoutEntityList" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PayoutInformationList">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PayoutInformation" type="PayoutInformationType" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PayoutEntityList">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Entity" type="EntityType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="PayoutMode">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Video">
        <xs:complexType>
          <xs:attribute name="MaterialMode" type="xs:string"/>
          <xs:attribute name="PayoutMode" type="xs:string"/>
        </xs:complexType>
      </xs:element>
      <xs:element name="Audio">
        <xs:complexType>
          <xs:attribute name="MaterialMode" type="xs:string"/>
          <xs:attribute name="PayoutMode" type="xs:string"/>
          <xs:attribute name="DownMixMode" type="xs:string"/>
          <xs:attribute name="DownMixParameter" type="xs:string"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:complexType name="PayoutInformationType">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="Identification" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

```

```

<xs:element ref="PayoutMode" minOccurs="0"/>
<xs:element ref="AdditionalService" minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="Code">
  <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="MainCode" type="xs:string" minOccurs="0"/>
      <xs:element name="SubCode" type="xs:string" minOccurs="0"/>
      <xs:element name="FormatCode" type="xs:string" minOccurs="0"/>
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:element name="Comment" type="xs:string" minOccurs="0"/>
</xs:sequence>
</xs:complexType>
<xs:element name="AdditionalService">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ServiceType" minOccurs="0">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="Caption"/>
            <xs:enumeration value="Data"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="ServiceUse" type="xs:boolean" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Trigger">
  <xs:complexType>
    <xs:attribute name="StartTime" type="TimecodeType"/>
    <xs:attribute name="RecvCue" type="xs:string"/>
    <xs:attribute name="SendCue" type="xs:string"/>
    <xs:attribute name="Duration" type="TimecodeType"/>
  </xs:complexType>
</xs:element>
<xs:complexType name="EntityType">
  <xs:sequence>
    <xs:element ref="Trigger"/>
    <xs:element name="EntityName" type="xs:string" minOccurs="0"/>
    <xs:element name="Line" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="Node" minOccurs="0" maxOccurs="2"/>
          <xs:element ref="PayoutMode" minOccurs="0"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>

```

```

    <xs:attribute name="ChannelName" type="xs:string" use="required"/>
  </xs:complexType>
</xs:element>
  <xs:element name="Comment" type="xs:string" minOccurs="0"/>
</xs:sequence>
  <xs:attribute name="Sequence" type="xs:integer" use="required"/>
</xs:complexType>
<xs:element name="Node">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="NodeContent" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="NodeContentType" minOccurs="0">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="ContentName"/>
                  <xs:enumeration value="Roll"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
            <xs:element name="NodeContentValue" type="xs:string" minOccurs="0"/>
          </xs:sequence>
          <xs:attribute name="ContentIdentifier" type="xs:string"/>
          <xs:attribute name="TimeCode" type="TimecodeType"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="Effect" type="xs:string"/>
    <xs:attribute name="Method" type="xs:string"/>
    <xs:attribute name="Media" type="xs:string"/>
  </xs:complexType>
</xs:element>

```

#### ■ Semantics:

Names	Definitions
CueSheet	CueSheet (set) element
PlayoutInformationList	List of playout information
PlayoutInformation	Playout information
PlayoutEntityList	List of playout entity
Identification	Identification (set) element
PlayoutMode	Playout mode information of video and audio
Video	Video material aspect ratio and broadcast aspect ratio
@MaterialMode	Attribute for material aspect ratio
@PlayoutMode	Attribute for broadcast aspect ratio

Audio	Material audio mode and broadcast audio mode
@MaterialMode	Attribute for material audio mode
@PlayoutMode	Attribute for broadcast audio mode
@DownMixMode	Attribute for downmix conversion mode
@DownMixParameter	Attribute for downmix parameter
Code	Information regarding program main code, program sub code and format number
MainCode	Program main code
SubCode	Program sub code
FormatCode	Format number
Comment	Comment, annotation
AdditionalService	Information regarding closed caption and datacasting
ServiceType	Indicates whether closed caption or datacasting
ServiceUse	Indicates whether closed caption or datacasting is in use
Entity	Each lap of the cue sheet
@Sequence	Attribute for representing sequence number of each lap
Trigger	Trigger of the lap
@StartTime	Attribute for lap start time (fixed time, undefined time)
@RecvCue	Attribute for receiving inter-broadcaster control signal
@SendCue	Attribute for sending inter-broadcaster control signal
@Duration	Attribute for duration
EntityName	Name of lap
Line	Description about output feed
@ChannelName	Attribute for name of output feed (e.g. Local feed, network feed, overlap feed etc.)
Node	Description of video and audio in the lap
@Effect	Attribute for representing node effect (e.g. fade-in, cutout, etc.)
@Method	Attribute for source of the node (e.g. Network feed, studio lives, VCR, etc.)
@Media	Attribute for describing essence of node (e.g. video, audio)
NodeContent	Information regarding video and audio content of the lap
@ContentIdentifier	Attribute for content ID
@TimeCode	Attribute for content start timecode
NodeContentType	Contentname or roll number
NodeContentValue	Indicates value of content name or roll number



## 6.4. Operational Guidelines of Program Exchange Metadata

### 6.4.1 File Format of Program Exchange Metadata Document

Metadata shall be conveyed in one metadata document (XML document with XML declaration at the first line, main document starts with <Pem\_main> and end with </Pem\_main> after second line.)

XML declaration should be described as follows.

```
<?xml version="1.0" encoding="UTF-16"?>
```

### 6.4.2 Extension of Program Exchange Metadata Document

In case of extending schema of program exchange metadata locally, add below

“Pem\_main/Extensions/xs:any” element in the schema.

### 6.4.3 Operational Guideline of Blocks

Program exchange metadata is possible to describe information about specific section which is time wise consecutive video/audio essence called “blocks”.

Figure 6-3 shows the concept of relation between program, roll and block.

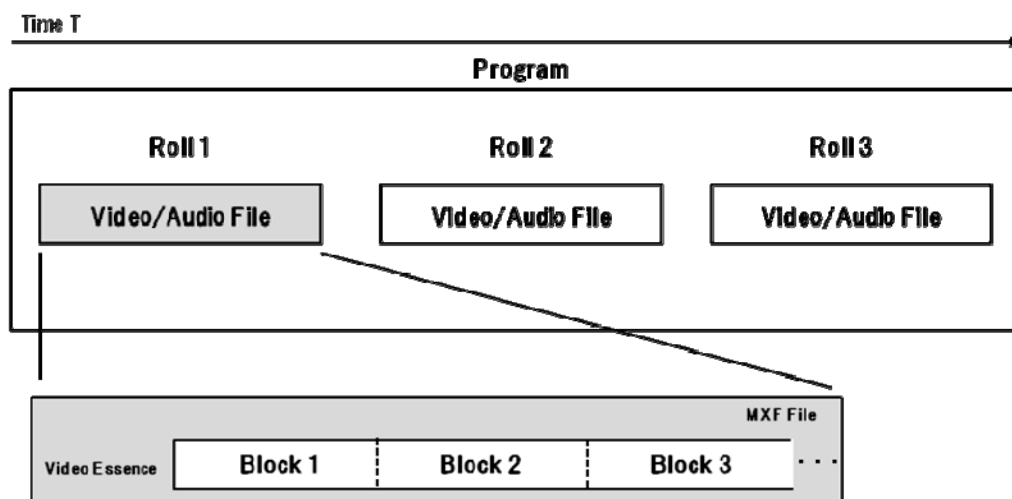


Figure 6-3 Relation of program, roll and block

“Roll” is a concept that physically time wise splitting of a program before making MXF file. “Block” is a concept that time wise partitioning of an essence after MXF file is made.

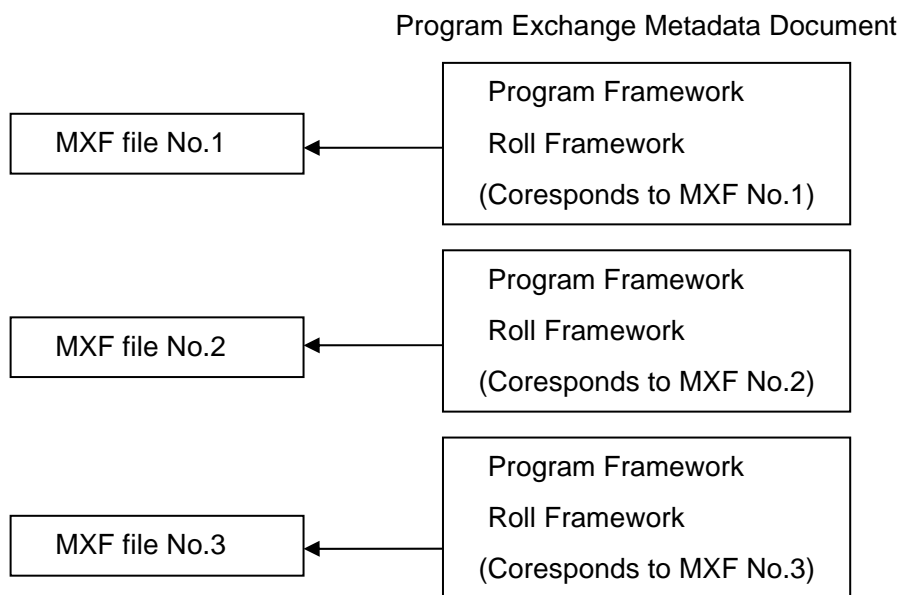
When describing a certain time wise essence section in a “Roll”, use “Blockelement”.

#### 6.4.4 Division of Program Exchange Metadata

In case program consists of more than one MXF file, program exchange metadata document corresponding to the MXF files shall be one file as defined in section 6.1.3, but as shown in figure 6-4, dividing of program exchange metadata document to correspond for multiple MXF files may be tolerated. Operational guidelines for dividing the program exchange metadata document to correspond for multiple MXF files are shown as follows.

Case 1: Case program framework and roll framework are merged into a same program exchange metadata document

In case a program consists of more than one MXF file and multiple program exchange metadata document corresponding to the each MXF files, each program exchange metadata document shall consists of program framework and roll framework which correspond essence in each MXF file. Moreover, program framework metadata in each program exchange metadata document shall contain same metadata element.

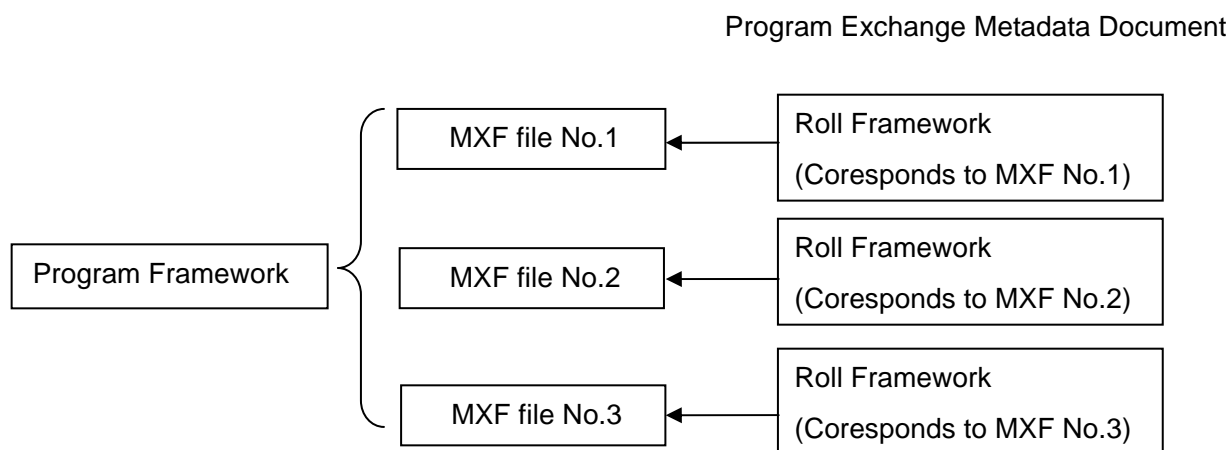


**Figure 6-4 Case program framework and roll framework are merged into a same program exchange metadata document**

Case 2: Case program framework and roll framework are in separate program exchange metadata document

In case program exchange metadata document which describes whole program, and program exchange metadata document which describe each MXF file are separated as shown in figure 6-5,

program exchange metadata document which describes whole program shall consists of program framework only. Program exchange metadata documents which describe each MXF file shall consist of roll framework only.



**Figure 6-5 Case program framework and roll framework are in separate program exchange metadata document**

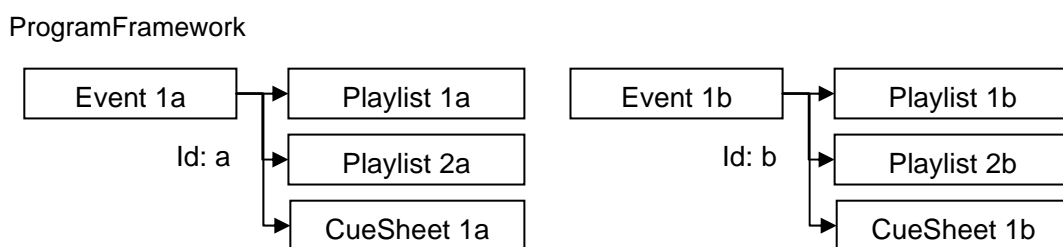
#### 6.4.5 Operational Guidelines for Identification Element and MXFInfo Element

When MXF file in each roll is single OP-1a file, VideoDescription element, AudioDescription element, or “Identification”, “MXFInfo” which is under CaptionsDescription element, same MXF file ID value or file type can be written.

In this case, to avoid unmatched of “Identification” and “MXFInfo” value which is under each Description element, it is desirable not to describe “Identification” and “MXFInfo” for audio and caption.

#### 6.4.6 Operational Guidelines for Identification Set

By using same ID value for Identification set which is used in Event set, Playlist set and CueSheet set, Event set in ProgramFramework element which can be describe more than once, can link with PlayList and CueSheet. Example of linkage using Identification set is shown in Figure 6-6.



**Figure 6-6 Example of linkage using Identification set**

### 6.5. Mapping of Program Exchange Metadata Document to DMS-1

In this section, Method for mapping program exchange metadata to DMS-1, which is defined in SMPTE 380M, is defined.

When mapping the program exchange metadata to DMS-1, only program framework should be mapped. Because there is no corresponding element so that roll framework should not be mapped. For the mapping of program framework to DMS-1, elements should be mapped to set and element which has same name. When a program consists of multiple MXF files, same program exchange metadata should be embedded. When MXF file is OP-Atom format, program exchange metadata should be embedded in video file.

Use of metadata element in DMS-1, which this technical report does not define, is not prohibited. Moreover, apparatus which uses program exchange metadata in DMS-1 is desirable to skip metadata element that does not defined in this technical report. However, the apparatus should save or record as-is.

### 6.6. Embedding Method of Program Exchange Metadata Document into MXF File

There are several method to embed the program exchange metadata document into MXF file, for example, creating new descriptive metadata scheme other than DMS-1, or using generic stream which is defined in SMPTE 410, but in this technical report, it shall be out of the scope.

## Chapter 7      Transfer of Distribution Package

The distribution package in the chapter 2 is transferable by a physical medium or through a network. This chapter defines structures of the distribution package and operations in a distribution package transfer.

### 7.1.    Package Information Document

The package information document specifies assets in a distribution package. The package information document has a list of reference to all assets in the distribution package. Each asset in the list can have a unique ID, with which the asset itself can be identified in the distribution package.

#### 7.1.1      Description Language

XML is used as a description language in the program information document, where XML schemas and XML data types are used as its structures and data types.

Table 7-1 has the namespace in URI, which is used for the package information document, where the MIME type of the package information document is “text/xml”

**Table 7-1 Namespace in XML**

Namespace prefix	URI
pki	http://www.arib.or.jp/trb31/schemas/YYYY/PKI
xs	http://www.w3.org/2001/XMLSchema

Note: “YYYY” indicates the issue year of the description scheme. Refer to the version management section for details.

#### 7.1.2      Character Encoding

The character encoding in the package information document is UCS (Unicode 2.0) and its scheme is UTF-8.

#### 7.1.3      Filename

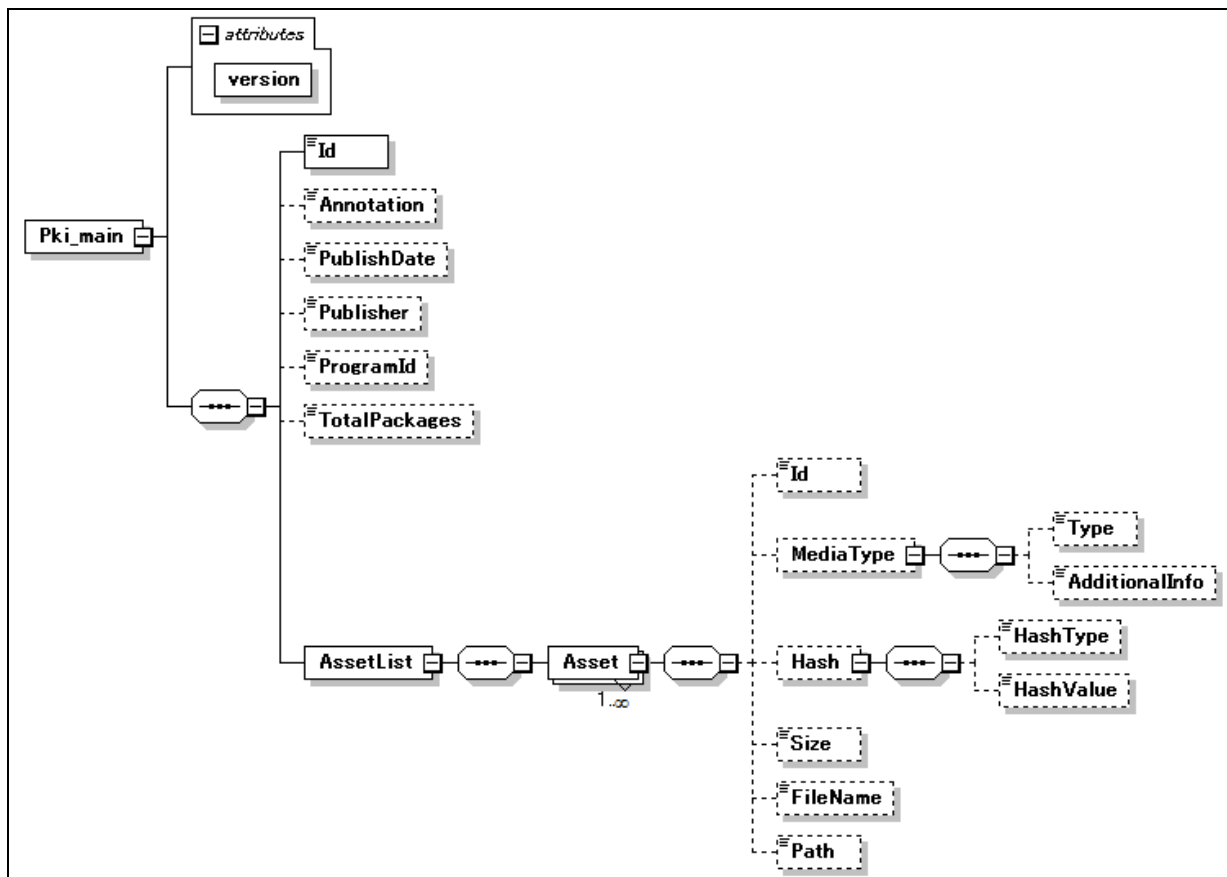
The filename of the package information document is packageinfo.xml, where single-byte and lower-case letters are used.

#### 7.1.4      Structure of Package Information Document

The package information document is described in XML and the root element is Pki\_main.

### 7.1.4.1 The Overview of the Package Information Document Structure

■ Diagram:



The following are semantics in the package information document.

■ Semantics

Names	Definitions	Type
Id	It is a unique identifier given to the package information document. The data type is xs:string	Required
Annotation	It is a free format text as an annotation for the distribution package. The data type is xs:string	Optional
PublishDate	It is a date and time when the package information document was published. The data type is xs:dateTime.	Optional
Publisher	It is a free format text as descriptive information about a creator (e.g. a company or a person) of the package information document. The data type is xs:string.	Optional
ProgramId	It is a unique identifier of the program to which the distribution package that contains the package information document	Optional

	belongs. The data type is xs:string.	
TotalPackages	It is the total number of distribution packages that have the same ProgramId. The data type is xs:positiveInteger.	Optional
AssetList	It is a list of the Asset elements.	Required
Asset	It is information of an individual asset element contained in the distribution package. Its structure is described in the following page.	Required
@version	Version of the package information document description scheme	Required

#### 7.1.4.2 Asset Element

The package information document can have a table of information about every asset that the distribution package contains. Every asset information is described in the asset element, of which detail data type can be referred in the XML Schema in Appendix. The semantics of elements in the asset are following:

## ■ Semantics

Names	Definitions	Type
Id	It is a unique identifier given to the asset. The data type is xs:string.	Optional
MediaType	It is a type of the asset	Optional
Type	It is a type of the asset (e.g. video, audio, caption, video/audio, video/audio/caption, metadata). The data type is xs:string.	Optional
AdditionalInfo	It is an additional information of the asset. The data type is xs:string.	Optional
Hash	It is a message digest of the asset, which is used to verify fixity of the asset.	Optional
HashType	It is a type of the hash function used for message digest of the asset. The data type is xs:string.	Optional
HashValue	It is a hash value generated by the hash function. The data type is xs:hexBinary.	Optional
Size	It is the size of the asset in byte. The data type is xs:positiveInteger	Optional
FileName	It is a filename of the asset. The data type is xs:string.	Optional
Path	It is path information of the asset file, which is described as URL defined by RFC173. The semantic and format depend on the medium to use. The data type is xs:anyURI.	Optional

### 7.1.5 Operations of the Package Information Document

#### 7.1.5.1 Operation of ProgramId Element

According to the transfer method, a single program to transfer may be divided to multiple distribution packages.

Pki\_main/ProgramId element is used to identify which program the distribution package belongs to. So that, the distribution packages that belong to the same program shall have the same value in Pki\_main/ProgramId. It is recommended for Pki\_main/TotalPackages to have the total number of the distribution packages if the package information document has Pki\_main/ProgramId in order to declare the program is composed of multiple distribution packages. If the package information document has no Pki\_main/ProgramId, it means the program is composed of a single distribution package.

#### 7.1.5.2 Operation of MediaType Element

Pki\_main/AssetList/Asset/MediaType is used to describe the type of essence of the file to be specified as the asset.



Examples:

The case that the type of an essence of file is video and audio:

Pki\_main/AssetList/Asset/MediaType/Type="Video+Audio"

The case that the type of an essence of file is caption:

Pki\_main/AssetList/Asset/MediaType/Type="Caption"

### 7.1.5.3 Dividing an Asset

As in the chapter 2, an asset in a distribution package will be a video file, an audio file, a caption file and so on.

In the case the size of a file or asset is bigger than the capacity of the storage medium where the file is stored, the file may be divided to multiple parts to make its size fit to the storage capacity. If the file is divided for this purpose and the file type will be other than the types described in the chapter 4 to 6, Pki\_main/AssetList/Asset/MediaType has the description that indicates the file is divided (e.g. Pki\_main/AssetList/Asset/MediaType/Type="Chunk").

In addition, Pki\_main/AssetList/Asset/MediaType/AdditionalInfo can have additional information about the number of divided files or a method of file divide.

Note that the method of file divide is out of scope of this document.

### 7.1.6 Version Management

Version management of the Package Information Document description scheme is defined as follows.

- The version consists of major version and minor version, and expressed as <major version>.<minor version> format.
- When modifying without backward compatibility, major version shall be incremented, and when modifying with backward compatibility, minor version shall be incremented.
- Version number is indicated in root attribute "version".
- Issue year (4 characters of the dominical year) shall be included in the part of the namespace, and when using new major version, issue year shall be indicated. (In case of minor version up, issue year of the namespace shall not be changed and only minor version of the version attribute shall be incremented.)

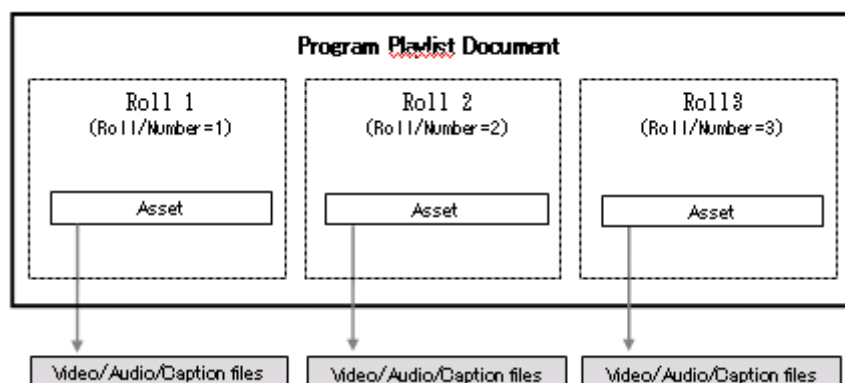
E.g. In case of issuing new major version in 2010.

<http://www.arib.or.jp/trb31/schemas/2010/PKI>

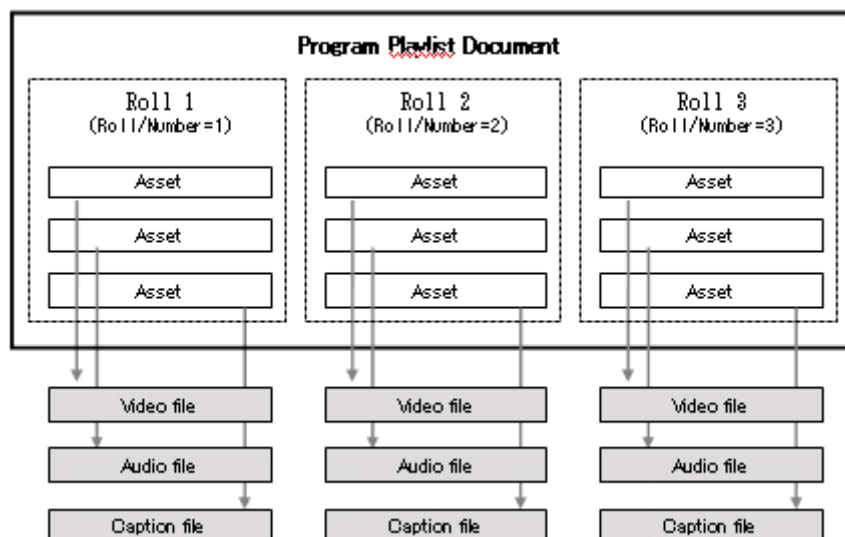
## 7.2. Program Playlist Document

As in the chapter 2, a program is composed of one or more rolls. Each roll is composed of one or more assets.

The program playlist document is the list of program rolls in play order. The relation between the program playlist document and file instances is shown in Figure 7-1, Figure 7-2.



**Figure 7-1** An example of the relation between the program playlist document and the file instances in Op-1a case.



**Figure 7-2** An example of the relation between the program playlist document and the file instances in Op-Atom case

The program playlist document can have multiple assets in a single roll to play them concurrently. Each asset can have an entry point and duration, which are described later, to specify the segment to play.



### 7.2.1 Description Language

The program playlist document is written in XML, where the XML schema is used to describe its structure and data type.

Table 7-2 has the namespace that is used in the program playlist document, where MIME type is “text/xml”.

**Table 7-2 XML Namespace**

Namespace prefix	URI
Ppl	http://www.arib.or.jp/trb31/schemas/YYYY/PPL
Xs	http://www.w3.org/2001/XMLSchema

Note: “YYYY” indicates the issue year of the description scheme. Refer to the version management section for details.

### 7.2.2 Character Encoding

The character encoding in the program playlist document is UCS (Unicode 2.0) and its scheme is UTF-8.

### 7.2.3 Filename

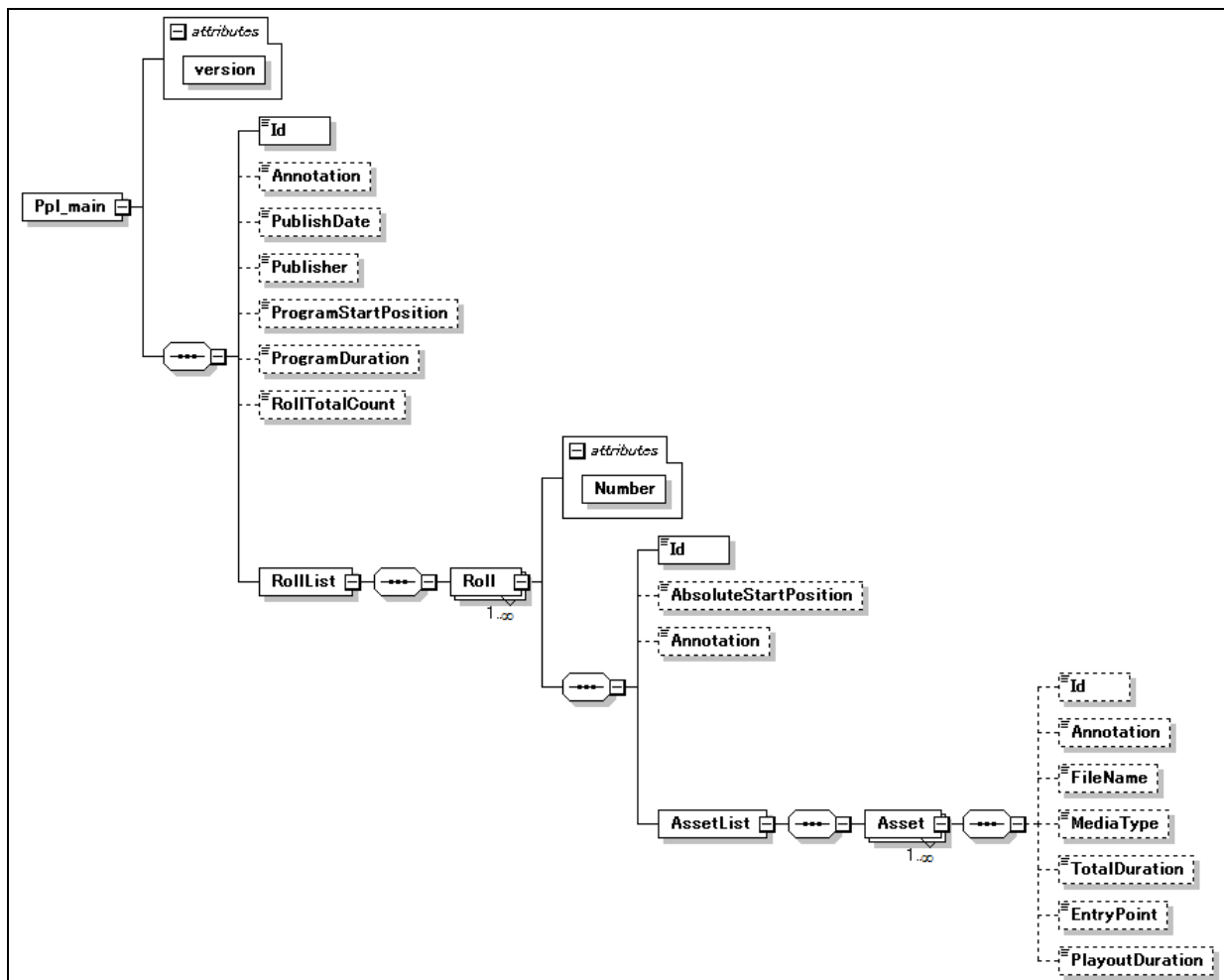
The filename of the program playlist document is programplaylist.xml, where single-byte and lower-case letters are used.

### 7.2.4 Structure of the Program Playlist Document

The program playlist document is described in XML and the root element is Ppl\_main.

### 7.2.4.1 The Overview of the Program Playlist Document Structure

#### ■ Diagram



#### ■ Semantics

Names	Definitions	Type
Id	It is a unique identifier given to the program playlist document. The data type is xs:string	Required
Annotation	It is a free format text as an annotation for the program playlist document. The data type is xs:string	Optional
PublishDate	It is a date and time when the program playlist document was published. The data type is xs:dateTime.	Optional
Publisher	It is a free format text as a descriptive information about a creator (e.g. a company or a person) of the program playlist document. The data type is xs:string.	Optional
ProgramStartPosition	It is the start time of the program in time code value.	Optional,
ProgramDuration	It is the length of the program in time code value.	Optional
RollTotalCount	It is the total number of rolls. The data type is	Optional

	xs:positiveInteger	
RollList	It is the list of roll elements.	Required
Roll	It is the element to describe information about the roll. Its structure will be described below.	Required
@version	Version of the program playlist document description scheme	Required

#### 7.2.4.2 Roll Element

A roll is composed of one or more assets. The roll element describes information of the asset that belongs to the same roll. The semantics of elements in the roll element are the following:

##### Semantics

Names	Definitions	Type
Id	It is a unique identifier given to the roll. The data type is xs:string	Required
AbsoluteStartPosition	It is the initial time code where the roll is placed.	Optional
Annotation	It is a free format text as an annotation for the roll. The data type is xs:string	Optional
@Number	It is an attribute to have a roll number.	Required
AssetList	It is the list of assets in the roll.	Required
Asset	It is an element to have asset information. Its structure will be described below.	Required

#### 7.2.4.3 Asset Element

The asset element has information of each asset in the roll. The following are the semantics of the asset elements:

##### Semantics

Names	Definitions	Type
Id	It is a unique identifier given to the asset. The data type is xs:string	Optional
Annotation	It is a free format text as an annotation for the asset. The data type is xs:string	Optional
FileName	It is a filename of the asset. The data type is xs:string.	Optional
MediaType	It is a type of the asset (e.g. video, audio, closed caption, video/audio, video/audio/closed caption). The data type is xs:string.	Optional
TotalDuration	It is a length of the asset in time code value.	Optional
EntryPoint	It is a start position in the asset to play. It is given as a length from the beginning of the asset in time code value or the number of frames.	Optional
PlayoutDuration	It is a length to play from the EntryPoint in time code value.	Optional

## 7.2.5 Operation of the Program Playlist Document

### 7.2.5.1 Specifying a file in Asset Element

A program playlist document can specify the assets that are contained in the same roll with Ids of the asset or filenames.

The assets that are in the same roll will be basically independent files such as a video file, an audio file or a closed caption file. It is not assumed that a file spans to multiple rolls but only an exception is a closed caption file that will span over multiple rolls for an operation reason. In such the exception case, the rolls that share the single closed caption file should have the same closed caption file ID or filename in Id or Filename element.

### 7.2.5.2 Identifying an Asset

In the program playlist document, the value of the Id element of an asset depends on the type of asset file; video, audio or closed caption. It is recommended to refer Table 7-3 in setting the value of the Id element.

**Table 7-3 Asset type file and Id value<sup>1</sup>**

Asset file type	Value of Id
Video file	UMID of the file
Audio file	UMID of the file
Video/Audio file	UMID of the file
Video/Audio/Closed Caption file	UMID of the file
Caption file	An unique value of the file (*)

\*The Asset Number in the Program Management Information of ARIB STD-B36 is recommended.

### 7.2.5.3 Operation of MediaType Element

Ppl\_main/RollList/Roll/AssetList/Asset/MediaType has the type of essence of the file that is specified as an asset.

Examples

In case that the essence of file is video and audio:

Ppl\_main/RollList/Roll/AssetList/Asset/MediaType="Video+Audio"

In case that the essence of file is caption:

Ppl\_main/RollList/Roll/AssetList/Asset/MediaType="Closed Caption"

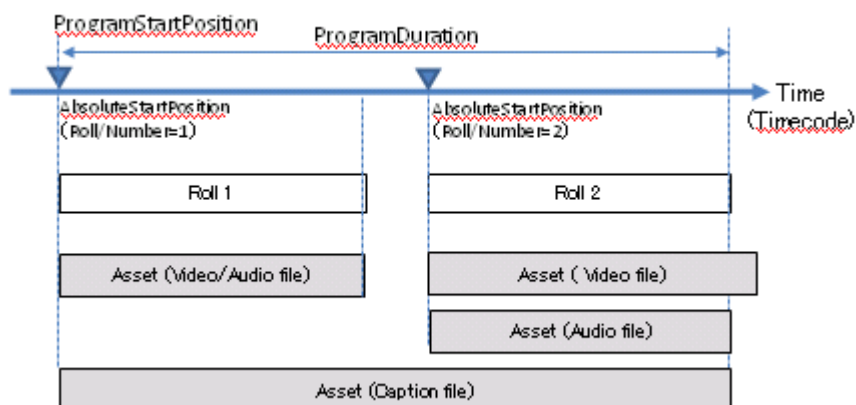
#### 7.2.5.4 Timing parameters in Roll

The program playlist document can have the time information of a program, a roll or each asset in a roll, which is referred in reconstructing a program.

- The time information of a program

A time information as a program timeline will be given with ProgramStartPosition and ProgramDuration.

The start position of a program is given with ProgramStartPosition in time code value (Timecode) and the length is given with ProgramDuration. An example of the relation between rolls and assets in a program timeline is shown in Figure 7-3.



**Figure 7-3 An example of rolls and assets in a program timeline**

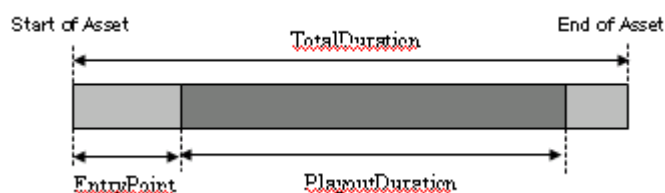
- Time information of roll

A roll can be placed at an absolute time position in a program. A roll or rolls that consist of a program can have the start position that is given with Roll/AbsoluteStartPosition in time code value

- Time information of asset

Asset can have three time elements for playout timing; TotalDuration, EntryPoint and PlayoutDuration. Figure 7-4 has the timing parameters of an asset.





**Figure 7-4 Asset timing parameters**

TotalDuration has the full length of asset. The duration of an asset in a roll cannot be less than 1 second.

EntryPoint has the start time to playout the asset.

All assets in the same roll shall be played at the position specified by EntryPoint synchronously.

However, a caption file will be an exception because it may span over rolls as described in 7.2.5.1

PlayoutDuration has the length to play from EntryPoint. If the program playlist document doesn't have EntryPoint and PlayoutDuration, the asset is played from the beginning of the asset and finishes at the end.

The total length of the roll to play shall be equal to the PlayoutDuration of the asset that has a video essence. If there is no video essence, the shortest TotalDuration in the assets will be used as the length to play.

### 7.2.6 Version Management

Version management of the Program Playlist Document description scheme is defined as follows.

- The version consists of major version and minor version, and expressed as <major version>.<minor version> format.
- When modifying without backward compatibility, major version shall be incremented, and when modifying with backward compatibility, minor version shall be incremented.
- Version number is indicated in root attribute "version".
- Issue year (4 characters of the dominical year) shall be included in the part of the namespace, and when using new major version, issue year shall be indicated. (In case of minor version up, issue year of the namespace shall not be changed and only minor version of the version attribute shall be incremented.)

E.g. In case of issuing new major version in 2010.

<http://www.arib.or.jp/trb31/schemas/2010/PPL>

### **7.3. Distribution Package Transfer**

The distribution package is able to transfer by a physical medium or through a network. Here describes the methods to transfer the distribution package.

#### **7.3.1 Transfer by a Physical Medium (recording medium)**

##### **7.3.1.1 Package Information Document**

It is recommended that a recording medium has the package information document defined in this document when the distribution package is transferred by a physical medium. However the use of the package information document may not be enforced if a proprietary document or data can work as a replacement of the package information document.

##### **7.3.1.1.1 Location of the Document on a Recording Medium**

It is recommended to store the package information document in the same directory on a file system where the associated distribution package is stored.

##### **7.3.1.1.2 Path Element**

It is recommended to have Pki\_main/Assetlist/Asset/Path element in the package information document in order to specify an absolute or relative path of the package information document on the recording medium. The syntax of the path element follows the operations below:

- URL Scheme

The path element uses “file” as a URL scheme, where hostname segment is NULL. Therefore, all URL value begins with “file:///”.

- URL Path

The path element is composed of the following 8bit ASCII codes:、

a-z/A-Z/0-9/“-” (dash)/ “\_” (underscore)/“.” (period)

The path can be divided to segments with “/” (slash). The length of path shall be less or equal to 1024 characters. The length of segment shall be less or equal to 128. The total number of segments should be less or equal to 10.

### **7.3.1.2 Program Playlist Document**

A recording medium can have the program playlist document when a program is divided to rolls. However the use of the program playlist document may not be enforced if a proprietary document or data can work as a replacement of the program playlist document.

#### **7.3.1.2.1 Location of the Document on a Recording Medium**

It is recommended to store the program playlist document in the same directory on a file system where the associated distribution package is stored.

### **7.3.2 Transfer through a Network**

A online data transfer through a point to point network needs a lot of definitions such as a transfer protocol, coding or transfer method of control information, information about a file to transfer, a mechanism of transfer or security.

If this document defines those things, both a sender and a receiver can share the basic method of file transfer. However, it might results in inflexibility in the use of the latest IP technology for improving operations or functions in file transfer then might become less beneficial.

For the reason above, this document focuses just on the minimum function requirements that will be mandatory. Any other function requirement will not be defined but may be suggested as a recommendation.

As an example of the protocol to negotiate parameters in network transfer between systems, MDP (Media Dispatch Protocol) as a SMPTE standard (SMPTE 2032) is available, of which details are in Appendix 4.

#### **7.3.2.1 Package Information Document**

It is recommended that the package information document is transferred with an associated distribution package when a distribution package is transfer through a network.

#### **7.3.2.2 Program Playlist Document**

It is possible for the program playlist document to be transferred with an associated distribution package when a program is divided to rolls.

## **Appendix 1      User Requirement**

NO TRANSLATION

## Appendix 2 Program Exchange Metadata XML Schema

```

<?xml version="1.0" encoding="utf-16"?>
<xs:schema xmlns="http://www.arib.or.jp/trb31/schemas/2010/PEM"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.arib.or.jp/trb31/schemas/2010/PEM"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <!-- ##### root ##### -->
  <xs:element name="Pem_main">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="MetadataId" type="xs:string"/>
        <xs:element name="MetadataVersion" type="xs:string"/>
        <xs:element ref="ProgramFramework" minOccurs="0"/>
        <xs:element ref="RollFramework" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Extensions" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" use="required" fixed="1.1"/>
    </xs:complexType>
  </xs:element>
  <!-- ##### Framework ##### -->
  <!-- Program Framework-->
  <xs:element name="ProgramFramework">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Titles" minOccurs="0"/>
        <xs:element ref="Identification" minOccurs="0"/>
        <xs:element ref="GroupRelationship" minOccurs="0"/>
        <xs:element ref="Event" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Participant" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Contract" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="Playlist" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="CueSheet" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <!-- Roll Framework -->
  <xs:element name="RollFramework">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Identification" minOccurs="0"/>
        <xs:element ref="GroupRelationship" minOccurs="0"/>
        <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="VideoDescription" minOccurs="0"/>
        <xs:element ref="AudioDescription" minOccurs="0"/>
        <xs:element ref="CaptionsDescription" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

```

    <xs:element ref="AncillaryDescription" minOccurs="0"/>
    <xs:element ref="Block" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="Keypoint" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
<!-- Extensions -->
<xs:element name="Extensions">
  <xs:complexType>
    <xs:sequence>
      <xs:any maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- ##### Sets ##### -->
<!-- Titles-->
<xs:element name="Titles">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MainTitle" type="xs:string" minOccurs="0"/>
      <xs:element name="MainTitleRuby" type="xs:string" minOccurs="0"/>
      <xs:element name="SecondaryTitle" type="xs:string" minOccurs="0"/>
      <xs:element name="SecondaryTitleRuby" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Identification-->
<xs:element name="Identification">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="IdentifierKind" type="xs:string" minOccurs="0"/>
      <xs:element name="IdentifierValue" type="xs:string" minOccurs="0"/>
      <xs:element name="IdentificationIssuingAuthority" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Group Relationship -->
<xs:element name="GroupRelationship">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ProgrammingGroupKind" type="xs:string" minOccurs="0"/>
      <xs:element name="ProgrammingGroupTitle" type="xs:string" minOccurs="0"/>
      <xs:element name="GroupSynopsis" type="xs:string" minOccurs="0"/>
      <xs:element name="NumericalPositionInSequence" type="xs:unsignedInt" minOccurs="0"/>
      <xs:element name="TotalNumberInSequence" type="xs:unsignedInt" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

```

```

</xs:element>
<!-- Event-->
<xs:element name="Event">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element name="EventIndication" type="xs:string" minOccurs="0"/>
      <xs:element name="EventStartDateandTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="EventEndDateandTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element ref="Publication" minOccurs="0"/>
      <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Publication-->
<xs:element name="Publication">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PublishingOrganizationName" type="xs:string" minOccurs="0"/>
      <xs:element name="PublishingServiceName" type="xs:string" minOccurs="0"/>
      <xs:element name="PublishingMediumName" type="xs:string" minOccurs="0"/>
      <xs:element name="PublishingRegionName" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Annotation-->
<xs:element name="Annotation">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AnnotationKind" type="xs:string" minOccurs="0"/>
      <xs:element name="AnnotationSynopsis" type="xs:string" minOccurs="0"/>
      <xs:element name="AnnotationDescription" type="xs:string" minOccurs="0"/>
      <xs:element ref="Classification" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Classification-->
<xs:element name="Classification">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ContentClassification" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Participant -->
<xs:element name="Participant">
  <xs:complexType>

```

```

<xs:sequence>
  <xs:element ref="Person" minOccurs="0"/>
  <xs:element ref="Organization" minOccurs="0"/>
  <xs:element name="JobFunction" type="xs:string" minOccurs="0"/>
  <xs:element name="ContributionStatus" type="xs:string" minOccurs="0"/>
  <xs:element name="ContributionDate" type="xs:dateTime" minOccurs="0"/>
  <xs:element name="ContributionLocation" type="xs:string" minOccurs="0"/>
  <xs:element ref="Annotation" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<!-- Person -->
<xs:element name="Person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="FamilyName" type="xs:string" minOccurs="0"/>
      <xs:element name="FirstGivenName" type="xs:string" minOccurs="0"/>
      <xs:element name="PersonDescription" type="xs:string" minOccurs="0"/>
      <xs:element ref="Organization" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Organisation -->
<xs:element name="Organization">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="OrganizationKind" type="xs:string" minOccurs="0"/>
      <xs:element name="OrganizationMainName" type="xs:string" minOccurs="0"/>
      <xs:element name="OrganizationCode" type="xs:string" minOccurs="0"/>
      <xs:element name="ContactDepartment" type="xs:string" minOccurs="0"/>
      <xs:element ref="Address" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Address-->
<xs:element name="Address">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="RoomSuiteNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="RoomSuiteName" type="xs:string" minOccurs="0"/>
      <xs:element name="BuildingName" type="xs:string" minOccurs="0"/>
      <xs:element name="PlaceName" type="xs:string" minOccurs="0"/>
      <xs:element name="StreetNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="StreetName" type="xs:string" minOccurs="0"/>
      <xs:element name="PostalTown" type="xs:string" minOccurs="0"/>
      <xs:element name="City" type="xs:string" minOccurs="0"/>
      <xs:element name="PostalCode" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```



```

    <xs:element name="Country" type="xs:string" minOccurs="0"/>
    <xs:element ref="Communications" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
<!-- Communications -->
<xs:element name="Communications">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CentralTelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="TelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="ExtensionTelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="MobileTelephoneNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="FaxNumber" type="xs:string" minOccurs="0"/>
      <xs:element name="E-mailAddress" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Contract -->
<xs:element name="Contract">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="SupplyContractNumber" type="xs:string" minOccurs="0"/>
      <xs:element ref="Rights" minOccurs="0"/>
      <xs:element ref="Participant" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Rights -->
<xs:element name="Rights">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="CopyrightOwner" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsHolder" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsManagementAuthority" type="xs:string" minOccurs="0"/>
      <xs:element name="RegionAreaOfIPLicense" type="xs:string" minOccurs="0"/>
      <xs:element name="IntellectualPropertyType" type="xs:string" minOccurs="0"/>
      <xs:element name="MediumName" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsCondition" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsRemarks" type="xs:string" minOccurs="0"/>
      <xs:element name="IntellectualPropertyRight" type="xs:string" minOccurs="0"/>
      <xs:element name="RightsStartDateTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="RightsStopDateTime" type="xs:dateTime" minOccurs="0"/>
      <xs:element name="MaxNumberOfUsages" type="xs:unsignedShort" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

<!-- Video Description-->
<xs:element name="VideoDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="FileDescription" minOccurs="0"/>
      <xs:element name="ImageAspectRatio" type="xs:string" minOccurs="0"/>
      <xs:element name="VideoCodec" type="xs:string" minOccurs="0"/>
      <xs:element name="VideoBitRate" type="xs:string" minOccurs="0"/>
      <xs:element name="ProfileAndLevel" type="xs:string" minOccurs="0"/>
      <xs:element name="MaxGOP" type="xs:unsignedShort" default="0" minOccurs="0"/>
      <xs:element name="ImageFormat" type="xs:string" minOccurs="0"/>
      <xs:element name="ColorbarType" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- AudioDescription-->
<xs:element name="AudioDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AudioCodec" type="xs:string" minOccurs="0"/>
      <xs:element name="AudioBitrate" type="xs:string" minOccurs="0"/>
      <xs:element name="ElectrospatialFormulation" type="xs:string" minOccurs="0"/>
      <xs:element name="SampleRate" type="xs:string" minOccurs="0"/>
      <xs:element name="ReferenceLevel" type="xs:string" minOccurs="0"/>
      <xs:element name="QuantizationBits" type="xs:unsignedShort" minOccurs="0"/>
      <xs:element name="DownmixParameter" type="xs:string" minOccurs="0"/>
      <xs:element name="ChannelList" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="ChannelObject" maxOccurs="8">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="ChannelAssignment" type="xs:string" minOccurs="0"/>
                  <xs:element name="AudioLanguage" type="xs:string" minOccurs="0"/>
                  <xs:element ref="FileDescription" minOccurs="0"/>
                </xs:sequence>
                <xs:attribute name="ChannelNumber" type="xs:unsignedShort"/>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
          <xs:attribute name="ChannelCount" type="xs:unsignedShort"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Captions Description-->

```

```

<xs:element name="CaptionsDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="FileDescription" minOccurs="0"/>
      <xs:element name="CaptionObjects" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="CaptionStandard" minOccurs="0"/>
            <xs:element name="CaptionBit" minOccurs="0">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="8bit"/>
                  <xs:enumeration value="10bit"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
            <xs:element name="CaptionLocation" minOccurs="0"/>
            <xs:element name="CaptionKind" minOccurs="0"/>
            <xs:element name="CaptionLanguageCode" minOccurs="0">
              <xs:complexType>
                <xs:simpleContent>
                  <xs:extension base="xs:string">
                    <xs:attribute name="LanguageNumber">
                      <xs:simpleType>
                        <xs:restriction base="xs:integer">
                          <xs:pattern value="[1-8]"/>
                        </xs:restriction>
                      </xs:simpleType>
                    </xs:attribute>
                  </xs:extension>
                </xs:simpleContent>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- AncillaryDescription -->
<xs:element name="AncillaryDescription">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AncillaryObjects" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="AncillaryKind" type="xs:string" minOccurs="0"/>

```

```

        <xs:element name="AncillaryInfo" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<!-- FileDescription -->
<xs:element name="FileDescription">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="Identification" minOccurs="0"/>
            <xs:element name="MXFInfo" minOccurs="0">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="OperationalPattern" type="xs:string" minOccurs="0"/>
                        <xs:element name="WrappingType" type="xs:string" minOccurs="0"/>
                        <xs:element name="FileType" type="xs:string" minOccurs="0"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="TimeCodeInfo" minOccurs="0">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="CountMode" type="xs:string" minOccurs="0"/>
                        <xs:element name="TotalDuration" type="TimecodeType" minOccurs="0"/>
                        <xs:element name="EntryPoint" minOccurs="0">
                            <xs:simpleType>
                                <xs:union memberTypes="xs:long TimecodeType"/>
                            </xs:simpleType>
                        </xs:element>
                        <xs:element name="PlayoutDuration" type="TimecodeType" minOccurs="0"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="RelatedMaterialInfo" minOccurs="0">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="Identification" minOccurs="0"/>
                        <xs:element name="MaterialKind" type="xs:string" minOccurs="0"/>
                        <xs:element name="MaterialDescription" type="xs:string" minOccurs="0"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

```

```

<!-- Block -->
<xs:element name="Block">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="BlockStartPosition" type="TimecodeType" minOccurs="0"/>
      <xs:element name="BlockDuration" type="TimecodeType" minOccurs="0"/>
      <xs:element name="BlockDescription" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="BlockKind" type="xs:string" minOccurs="0"/>
            <xs:element name="BlockSubInfo" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="BlockSubNumber" type="xs:unsignedShort" minOccurs="0"/>
                  <xs:element name="BlockSubKind" type="xs:string" minOccurs="0"/>
                  <xs:element name="BlockValue" type="xs:string" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="BlockNumber" type="xs:unsignedShort"/>
    <xs:attribute name="BlockTotalCount" type="xs:positiveInteger"/>
  </xs:complexType>
</xs:element>
<!-- Key Point -->
<xs:element name="Keypoint">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="KeypointKind" type="xs:string" minOccurs="0"/>
      <xs:element name="KeypointPosition" type="TimecodeType" minOccurs="0"/>
      <xs:element name="KeypointDuration" type="TimecodeType" minOccurs="0"/>
      <xs:element name="KeypointValue" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- Program Playlist -->
<xs:element name="Playlist">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Identification" minOccurs="0"/>
      <xs:element name="ProgramStartPosition" type="TimecodeType" minOccurs="0"/>
      <xs:element name="ProgramDuration" type="TimecodeType" minOccurs="0"/>
      <xs:element name="RollTotalCount" type="xs:positiveInteger" minOccurs="0"/>
      <xs:element name="RollList" minOccurs="0">

```

```

<xs:complexType>
  <xs:sequence>
    <xs:element name="Roll" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="Id" type="xs:string"/>
          <xs:element name="AbsoluteStartPosition" type="TimecodeType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="Number" type="xs:unsignedShort" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<!-- CueSheet -->
<xs:element name="CueSheet">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="PlayoutInformationList" minOccurs="0"/>
      <xs:element ref="PlayoutEntityList" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- PlayoutInformationList -->
<xs:element name="PlayoutInformationList">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="PlayoutInformation" type="PlayoutInformationType" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- ProgressList -->
<xs:element name="PlayoutEntityList">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Entity" type="EntityType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<!-- PlayoutMode-->
<xs:element name="PlayoutMode">
  <xs:complexType>
    <xs:sequence>

```

```

<xs:element name="Video">
  <xs:complexType>
    <xs:attribute name="MaterialMode" type="xs:string"/>
    <xs:attribute name="PayoutMode" type="xs:string"/>
  </xs:complexType>
</xs:element>
<xs:element name="Audio">
  <xs:complexType>
    <xs:attribute name="MaterialMode" type="xs:string"/>
    <xs:attribute name="PayoutMode" type="xs:string"/>
    <xs:attribute name="DownMixMode" type="xs:string"/>
    <xs:attribute name="DownMixParameter" type="xs:string"/>
  </xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<!-- PlayoutInformation -->
<xs:complexType name="PlayoutInformationType">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="Identification" minOccurs="0"/>
    <xs:element ref="PlayoutMode" minOccurs="0"/>
    <xs:element ref="AdditionalService" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Code">
      <xs:complexType>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="MainCode" type="xs:string" minOccurs="0"/>
          <xs:element name="SubCode" type="xs:string" minOccurs="0"/>
          <xs:element name="FormatCode" type="xs:string" minOccurs="0"/>
        </xs:choice>
      </xs:complexType>
    </xs:element>
    <xs:element name="Comment" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<!-- AdditionalService -->
<xs:element name="AdditionalService">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="ServiceType" minOccurs="0">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="Caption"/>
            <xs:enumeration value="Data"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>

```

```

    <xs:element name="ServiceUse" type="xs:boolean" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
<!-- Trigger -->
<xs:element name="Trigger">
  <xs:complexType>
    <xs:attribute name="StartTime" type="TimecodeType"/>
    <xs:attribute name="RecvCue" type="xs:string"/>
    <xs:attribute name="SendCue" type="xs:string"/>
    <xs:attribute name="Duration" type="TimecodeType"/>
  </xs:complexType>
</xs:element>
<!-- EntityType -->
<xs:complexType name="EntityType">
  <xs:sequence>
    <xs:element ref="Trigger"/>
    <xs:element name="EntityName" type="xs:string" minOccurs="0"/>
    <xs:element name="Line" maxOccurs="unbounded">
      <xs:complexType>
        <xs:sequence>
          <xs:element ref="Node" minOccurs="0" maxOccurs="2"/>
          <xs:element ref="PlayoutMode" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="ChannelName" type="xs:string" use="required"/>
      </xs:complexType>
    </xs:element>
    <xs:element name="Comment" type="xs:string" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="Sequence" type="xs:integer" use="required"/>
</xs:complexType>
<!-- NodeType -->
<xs:element name="Node">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="NodeContent" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="NodeContentType" minOccurs="0">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:enumeration value="ContentName"/>
                  <xs:enumeration value="Roll"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
            <xs:element name="NodeContentValue" type="xs:string" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```



```

    </xs:sequence>
    <xs:attribute name="ContentIdentifier" type="xs:string"/>
    <xs:attribute name="TimeCode" type="TimecodeType"/>
  </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="Effect" type="xs:string"/>
<xs:attribute name="Method" type="xs:string"/>
<xs:attribute name="Media" type="xs:string"/>
</xs:complexType>
</xs:element>
<!-- ##### DATA TYPES ##### -->
<!-- TimecodeType -->
<xs:simpleType name="TimecodeType">
  <xs:annotation>
    <xs:documentation>Timecode string(00:00:00:00)</xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:pattern value="[0-9]{2}:[0-9]{2}:[0-9]{2}:[0-9]{2}"/>
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```

## Appendix 3 Package Information / Program Playlist Document XML Schema

### 1 Package Information Document XML Schema

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns="http://www.arib.or.jp/trb31/schemas/2010/PKI"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.arib.or.jp/trb31/schemas/2010/PKI" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="Pki_main">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Id" type="xs:string"/>
        <xs:element name="Annotation" type="xs:string" minOccurs="0"/>
        <xs:element name="PublishDate" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="Publisher" type="xs:string" minOccurs="0"/>
        <xs:element name="ProgramId" type="xs:string" minOccurs="0"/>
        <xs:element name="TotalPackages" type="xs:positiveInteger" minOccurs="0"/>
        <xs:element name="AssetList">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Asset" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Id" type="xs:string" minOccurs="0"/>
                    <xs:element name="MediaType" minOccurs="0">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="Type" type="xs:string" minOccurs="0"/>
                          <xs:element name="AdditionalInfo" type="xs:string" minOccurs="0"/>
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                    <xs:element name="Hash" minOccurs="0">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="HashType" type="xs:string" minOccurs="0"/>
                          <xs:element name="HashValue" type="xs:hexBinary" minOccurs="0"/>
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                    <xs:element name="Size" type="xs:positiveInteger" minOccurs="0"/>
                    <xs:element name="FileName" type="xs:string" minOccurs="0"/>
                    <xs:element name="Path" type="xs:anyURI" minOccurs="0"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

```

    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="version" type="xs:string" use="required" fixed="1.1"/>
</xs:complexType>
</xs:element>
</xs:schema>

```

## 2 Program Playlist Document XML Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.arib.or.jp/trb31/schemas/2010/PPL"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.arib.or.jp/trb31/schemas/2010/PPL"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <!-- ProgramPlaylist -->
  <xs:element name="Ppl_main">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Id" type="xs:string"/>
        <xs:element name="Annotation" type="xs:string" minOccurs="0"/>
        <xs:element name="PublishDate" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="Publisher" type="xs:string" minOccurs="0"/>
        <xs:element name="ProgramStartPosition" type="TimecodeType" minOccurs="0"/>
        <xs:element name="ProgramDuration" type="TimecodeType" minOccurs="0"/>
        <xs:element name="RollTotalCount" type="xs:positiveInteger" minOccurs="0"/>
        <xs:element name="RollList">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Roll" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Id" type="xs:string"/>
                    <xs:element name="AbsoluteStartPosition" type="TimecodeType" minOccurs="0"/>
                    <xs:element name="Annotation" type="xs:string" minOccurs="0"/>
                    <xs:element name="AssetList">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="Asset" maxOccurs="unbounded">
                            <xs:complexType>
                              <xs:sequence>
                                <xs:element name="Id" type="xs:string" minOccurs="0"/>
                                <xs:element name="Annotation" type="xs:string" minOccurs="0"/>
                                <xs:element name="FileName" type="xs:string" minOccurs="0"/>
                                <xs:element name="MediaType" type="xs:string" minOccurs="0"/>

```

```

        <xs:element name="TotalDuration" type="TimecodeType" minOccurs="0"/>
        <xs:element name="EntryPoint" minOccurs="0">
            <xs:simpleType>
                <xs:union memberTypes="xs:long TimecodeType"/>
            </xs:simpleType>
        </xs:element>
        <xs:element name="PlayoutDuration" type="TimecodeType" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="Number" type="xs:unsignedShort" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="version" type="xs:string" use="required" fixed="1.1"/>
</xs:complexType>
</xs:element>
<!--TimecodeType-->
<xs:simpleType name="TimecodeType">
    <xs:annotation>
        <xs:documentation>Timecode string(00:00:00:00)</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
        <xs:pattern value="[0-9]{2}:[0-9]{2}:[0-9]{2}:[0-9]{2}"/>
    </xs:restriction>
</xs:simpleType>
</xs:schema>

```

## **Appendix 4    Summary of Media Dispatch Protocol**

### **NO TRANSLATION**



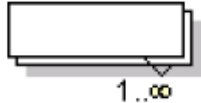
This appendix describes summary of SMPTE 2032 - Media Dispatch Protocol, therefore the translation is omitted.

## Appendix 5 XML Diagram





Notation of diagram which is used in this technical report to indicate XML structures are shown below.

### 1 Element Symbol

Optional element is shown by dashed-line, mandatory element is shown by solid-line and element which appears more than one times is shown by double-line. Element suffix shows number of appearance.


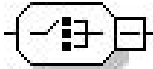
		
Optional Element	Mandatory Element	Mandatory Element (More than 1)
Min. occurrence = 0	Min. occurrence = 1	Min. occurrence = 1
Max. occurrence = 1	Max. occurrence = 1	Max. occurrence = unbounded

Element content model is shown by left side or right side of element box.

			
Simple Content	Complex Content	Complex Content with Child Element	Content without Child Content

### 2 Model Symbol

Model symbols are shown below.

	Sequence of element. Element appears in sequence defined by schema.
	Selectable element. One of the child elements is selectable.







---

TECHNICAL METHODS FOR FILE-BASED PROGRAM EXCHANGE

ARIB TECHNICAL REPORT

ARIB TR-B31 VERSION 1.1-E1  
(March 28, 2011)

---

This Document is based on the ARIB Technical Report of “Technical  
Methods for File-Based Program Exchange” in Japanese edition and  
Translated into English in May, 2011

Published by

Association of Radio Industries and Businesses

11F, Nittochi Building,  
1-4-1 Kasumigaseki, Chiyoda-ku, Tokyo 100-0013, Japan

TEL 81-3-5510-8590  
FAX 81-3-3592-1103

Printed in Japan  
All rights reserved

---