EBU

OPERATING EUROVISION AND EURORADIO

TECH 3392

ADM BROADCAST PRODUCTION PROFILE

SOURCE: BTF GROUP

SPECIFICATION: Version 0.9

Geneva
October 2018

This page and others in the document are intentionally left blank to maintain pagination for two sided printing

Abstract

The Audio Definition Model (ADM) [BS.2076] is intentionally very generic to support a wide variety of different application areas. To keep the potential complexity in check and enable the interoperability of different implementations the “ADM broadcast production profile” constrains the ADM to simplify implementations and to prevent interoperability problems in the production of Next Generation Audio broadcast programmes. It aims to do so by not limiting the possibilities the ADM offers. Every constraint introduced by the profile is accompanied by a transcoding instruction, which explains how an ADM using the full feature set can be converted to an ADM which complies with this profile.

Status of this document

This document is a working draft and may be updated, replaced or made obsolete by other documents at any time without considering backwards compatibility. This document should always be cited as a work in progress. Version 1.0 of this document is expected to be published at the same time as the next version of the EBU ADM Renderer [EBU3388] to perfectly align the two. Comments and suggestions are invited and may be sent to sunna@ebu.ch.

This page and others in the document are intentionally left blank to maintain pagination for two sided printing

Contents

ADM Broadcast Production Profile

EBU Committee	*First Issued*	*Revised*	*Re-issued*
TC	2018		

Keywords: ADM, Audio Definition Model, Metadata, profile, Next Generation Audio, NGA.

1. Introduction

The term "Next Generation Audio" (NGA) consolidates all the innovative concepts which aim to deliver a better audio experience. This includes channel-based audio with an even higher speaker count ("3D audio"), scene-based audio ("Higher-Order Ambisonics"), object-based audio (audio with associated parameters), and combinations of those. It is safe to say that those concepts will provide a hugely powerful tool for developing myriads of as-yet unthought-of applications and services.

The "Audio Definition Model" (ADM) [BS.2076] is the standardised comprehensive list that describes all the metadata used in NGA. The ADM is consequentially very flexible and can allow very complex audio configurations to be described. The concept of extracting and grouping relevant sub-sets ("profiles") of this ADM metadata targeted at a particular application (or portion of the broadcast chain) is therefore a good one. It helps to keep the complexity in check and enable the interoperability of different implementations. This Tech report specifies an ADM Profile for use in broadcast production workflows. The profile constrains the ADM by enforcing additional rules on top of the [BS.2076].

1.1 Abbreviations

The following are definitions for certain terms that are used throughout the document.

ADM Audio Definition Model

BW64 Broadcast Wave 64 Format

DAW Digital Audio Workstation

EAR EBU ADM Renderer

HOA Higher-order Ambisonics

NGA Next Generation Audio

XML Extensible Markup Language

1.2 Scope

The constrained set of ADM metadata described in this specification is intended for the file-based

and live production blocks of the broadcast chain as shown in Figure 1. The rest of the simplified broadcast chain consists of a distribution block, and an emission block. These blocks will also require their own ADM profiles.

The purpose of the production blocks is to deliver tools for editing existing object-based content or creating such content from legacy audio material or other sources. The core of these production blocks may include an object-based DAW, which could be extended by several tools and workflows to import, edit, monitor and export object-based content. Widespread adoption of this profile by DAW manufacturers will effectively result in there being an interchange format for DAWs.

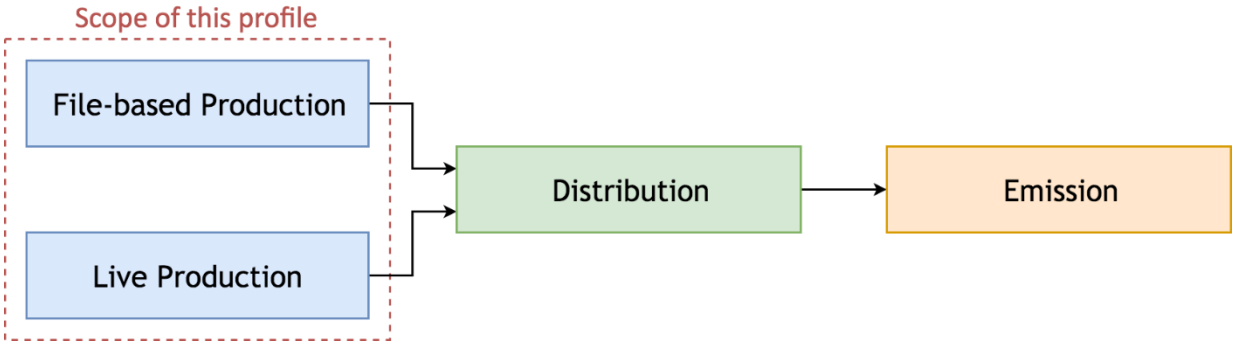

Figure 1: Simplified broadcast workflow showing the scope of this profile

1.3 Profile Levels

Some of the requirements of the profile are dependent on the used level. The profile level only sets a maximum limit to how often an element should be used but does not change which elements should be used. The choice of level should be based on the available processing power and storage of the production setup.

Table 1: Level dependent inclusive maximum limits for content

Variable	Description	Level			
		1	2	3	4
audioProgrammeCount	Number of audioProgrammes in an ADM file or flow.	4	16	32	unbounded
audioContentCount	Number of audioContents in an ADM file or flow.	4	16	128	unbounded
audioObjectCount	Number of audioObjects in an ADM file or flow.	8	64	512	unbounded
concurrentAudioObjectCount	Number of audioObjects that occur at the same time instant.	4	20	128	unbounded
interactiveObjectsCount	Number of audioObjects that are interactive in an audioProgramme.	1	4	16	unbounded
complementaryGroupsCount	Number of audioObjects that contain complementaryAudioObjectsIDRefs.	4	16	64	unbounded
audioObjectNestingLevel	Number of levels of nesting of audioObjects, where the nesting level is the number of audioObjects nested. See § 2.5 for an explanation of nesting.	2	2	4	unbounded

Table 2: Level dependent inclusive maximum limits for format

Variable	Description	Level			
		1	2	3	4
audioPackFormatCount	Number of audioPackFormats in an ADM file or flow. This does not include Common Definition audioPackFormats.	16	64	256	unbounded
audioPackFormatNestingLevel	Number of levels of nesting of audioPackFormats, where the nesting level is the number of audioPackFormats nested. Common Definition audioPackFormats are not restricted by this maximum. See § 2.5 for an explanation of nesting.	1	2	4	unbounded
audioChannelFormatCount	Number of audioChannelFormats in an ADM file or flow. Same number is used for audioTrackFormat and audioStreamFormat. This does not include Common Definition audioChannelFormats.	32	128	512	unbounded
audioTrackUIDCount	Number of audioTrackUIDs in an ADM file or flow.	32	128	512	unbounded

Table 3: General level dependent inclusive maximum limits

Variable	Description	Level			
		1	2	3	4
labelLength	Number of characters in a label (e.g. audioProgrammeName).	32	32	32	unbounded
trackCount	Number of tracks in the essence.	32	64	128	unbounded

1.4 Profile Type Combinations

A subset of the typeDefinitions can be used as a restriction. This allows simpler production tools to be used with a limited set of types. The choice of type combination should be based on the capability of the production tools.

Even though every possible type combination is allowed, some combinations are more sensible than others. For example, it is not very useful if a system only supports the typeDefinition “Matrix”. It should rather be combined with the typeDefinition “DirectSpeakers“, or “HOA”. Only supporting the typeDefinition “DirectSpeakers” on the other hand is very useful for legacy channel-based systems. Current object-based productions often do not use objects only but rather extend one or more channel-beds with some objects. Object-based tools may therefore not only use the typeDefinition “Objects”, but rather also support the “DirectSpeakers” typeDefinition. Also the typeDefinitions “HOA” or “Binaural” can benefit from some additional objects, and may therefore be combined with the typeDefinition “Objects”.

A string containing the first letter of each typeDefinition is used as an abbreviation for the profile type combination. The lower-case letters should be ordered according to their typeLabel value. The profile level and type combination are independent of each other, so both require specifying.

If in doubt use the highest level (“4”) and type combination (“dmohb”).

1.5 Referencing this profile

To reference this profile the urn “urn:ebu:tech:3392:1.0:l:t” should be used, where ‘l’ is the used level, and ‘t’ is the type combination abbreviation described in § 1.4. For example, “urn:ebu:tech:3392:1.0:2:dmo” should be used to reference this profile with level 2 and the allowed typeDefinitions “DirectSpeakers”, “Matrix” and “Objects”.

2. Requirements

This version of the profile is based on “ITU-R BS.2076-1”. Unless otherwise stated in the profile the full feature set of the ADM can be used.

2.1 Referencing

A referenced ID should always be either be the ID of an ADM element which is defined in the same file or flow, or the ID of an element defined in the common definitions.

2.2 Common definitions

If a format can fully or partially be described using the common definitions [BS.2094] the ADM elements from the common definition should be used. Please note, that the recommendation [BS.2094] includes both *typeDefinitions* and *typeLabels* to indicate the types of elements. As this profile only uses *typeLabels*, use the conversion described in § 3.

2.3 ADM elements

2.3.1 audioProgramme

The number of audioProgramme elements in an ADM file or flow is restricted to a maximum of *audioProgrammeCount*. The length of all audioProgrammes in an ADM file or flow should be the same.

Table 4: audioProgramme attribute requirements

Attribute	Requirements	Required
audioProgrammeID	No special requirement.	Yes
audioProgrammeName	Should not exceed *labelLength* characters.	Yes
audioProgrammeLanguage	Should be set as specified in ISO 639-1 (these are two-letter codes, such as ‘en’ for English) or ISO 639-2 (these are three-letter codes, such as ‘eng’ for English).	Optional
start	Should use the decimal based format HH:MM:SS.sssss.	Optional
end	Should use the decimal based format HH:MM:SS.sssss.	Optional
maxDuckingDepth	Should not be used.[1]	–

[1] There is no common understanding how the ducking parameters within the ADM are supposed to be used. So, it is strongly recommended not to use them.

Table 5: audioProgramme sub-element requirements

Sub-element	Requirements	Quantity
audioContentIDRef	No special requirement.	1…*
loudnessMetadata	Should only be used for audioProgrammes which only contain DirectSpeakers content.[2] If present should contain the measured loudness values	0 or 1
audioProgrammeReferenceScreen	Should be present if screen-related attributes are used.	0 or 1

Table 6: loudnessMetadata attribute requirements

Attribute	Requirements	Required
loudnessMethod	Should be set to “ITU-R BS.1770-4”	Yes
loudnessRecType	Should be set to “EBU R128”	Yes
loudnessCorrectionType	Should be set to either “file-based” or “real-time”	Optional

Table 7: loudnessMetadata sub-element requirements

Sub-element	Requirements	Quantity
integratedLoudness	No special requirement.	1
loudnessRange	Should not be used for audioProgrammes with a duration under 2 minutes.	0 or 1
maxTruePeak	No special requirement.	1
maxMomentary	No special requirement.	0 or 1
maxShortTerm	Should be used for audioProgrammes with a duration under 2 minutes.	0 or 1
dialogueLoudness	Should not be used.	—

Table 8: audioProgrammeReferenceScreen attribute requirements

Attribute	Requirements	Required
aspectRatio	No special requirement.	Yes

Table 9: audioProgrammeReferenceScreen sub-element requirements

Sub-element	Requirements	Quantity
screenCentrePosition	No special requirement.	2…3
screenWidth	No special requirement.	1

[2] There is no standardised method to measure the loudness for object-based or scene-based audio.

2.3.2 audioContent

The number of audioContent elements in an ADM file or flow is restricted to a maximum of *audioContentCount*.

Every audioContent should be referenced by at least one audioProgramme.

Table 10: audioContent attribute requirements

Attribute	Requirements	Required
audioContentID	No special requirement.	Yes
audioContentName	Should not exceed *labelLength* characters.	Yes
audioContentLanguage	Should be set as specified in ISO 639-1 (these are two-letter codes, such as 'en' for English) or ISO 639-2 (these are three-letter codes, such as 'eng' for English).	Optional

Table 11: audioContent sub-element requirements

Sub-element	Requirements	Quantity
audioObjectIDRef	No special requirement.	1…*
loudnessMetadata	If present should contain the measured loudness values.	0 or 1
dialogue	No special requirement.	0 or 1

2.3.3 audioObject

The number of audioObject elements in an ADM file or flow is restricted to a maximum of *audioObjectCount*. The number of concurrent audioObject elements is restricted to a maximum of *concurrentAudioObjectCount*. The number of audioObjects that are interactive is restricted to a maximum of *interactiveObjectsCount*. The number of audioObjects, which can contain audioComplementaryIDRefs is restricted to a maximum of *complementaryGroupsCount*. The number of nesting levels of audioObjects is restricted to a maximum of *audioObjectNestingLevel*.

Every audioObject should be referenced by at least one audioContent or audioObject.

Each audioObject should either refer to one audioPackFormats or one or more audioObjects.

If an audioObject refers to an audioPackFormat it should have start and duration attributes, and contain one audioTrackUIDRef for each track in the referenced audioPackFormats, and zero references to other audioObjects.

If an audioObject refers to one or more audioObjects then is should not have any start or duration attributes, and should contain zero references to audioPackFormats or audioTrackUIDs.

Table 12: audioObject attribute requirements

Attribute	Requirements	Required
audioObjectID	No special requirement.	Yes
audioObjectName	Should not exceed *labelLength* characters.	Yes
start	Should use the decimal based format HH:MM:SS.sssss (with at least 5 d.p. of the seconds).	Optional (see above)
duration	Should use the decimal based format HH:MM:SS.sssss	Optional (see above)

	(with at least 5 d.p. of the seconds).	
dialogue	No special requirement.	Optional
importance	No special requirement.	Optional
interact	No special requirement.	Optional
disableDucking	Should not be used.[3]	—

Table 13: audioObject sub-element requirements

Sub-element	Requirements	Quantity
audioPackFormatIDRef	No special requirement.	0 or 1
audioObjectIDRef	The referenced audioObject should not — directly or indirectly — reference back to this audioObject. The nesting level of audioObjects should not exceed *audioObjectNestingLevel*.	0…* (see above)
audioComplementaryObjectIDRef	Each audioObject can only be a member of one complementary group. Only *complementaryGroupCount* audioObjects within one audioProgramme can contain one or more audioComplementaryObjectID sub-elements.	0…* (see above)
audioTrackUIDRef	If the audioObject refers to an audioPackFormat it should also refer to the corresponding audioTrackUIDs.	0…* (see above)
audioObjectInteraction	Only i*nteractiveObjectCount* audioObjects within one audioProgramme can contain this sub-element.	0 or 1

Table 14: audioObjectInteraction attribute requirements

Attribute	Requirements	Required
onOffInteract	No special requirement.	Yes
gainInteract	No special requirement.	Optional
positionInteract	No special requirement.	Optional

Table 15: audioObjectInteraction sub-element requirements

Sub-element	Requirements	Quantity
gainInteractionRange	No special requirement.	0…2
positionInteractionRange	No special requirement.	0…3

[3] There is no common understanding how the ducking parameters within the ADM are supposed to be used; so it is strongly recommended not to use them.

2.3.4 audioTrackUID

The number of audioTrackUID elements in an ADM file or flow is restricted to a maximum of *audioTrackUIDCount*.

Table 16: audioTrackUID attribute requirements

Attribute	Requirements	Required
UID	No special requirement.	Yes
sampleRate	Should be ignored if available from the audio essence.	Optional
bitDepth	Should be ignored if available from the audio essence.	Optional

Table 17: audioTrackUID sub-element requirements

Sub-element	Requirements	Quantity
audioMXFLookUp	No special requirement.	0 or 1
audioTrackFormatIDRef	No special requirement.	1
audioPackFormatIDRef	Should reference the audioPackFormat which directly references a audioChannelFormat.	1

Table 18: MXF sub-element requirements

Sub-element	Requirements	Quantity
packageUIDRef	No special requirement.	0 or 1
trackIDRef	No special requirement.	0 or 1
channelIDRef	No special requirement.	0 or 1

2.3.5 audioPackFormat

The number of audioPackFormat elements in an ADM file or flow is restricted to a maximum of *audioPackFormatCount*. This does not include common definition audioPackFormat elements.

Table 19: audioPackFormat attribute requirements

Attribute	Requirements	Required
audioPackFormatID	No special requirement.	Yes
audioPackFormatName	Should not exceed *labelLength* characters.	Yes
typeLabel	If referenced by an audioPackFormat the typeLabel should match the typeLabel of the referent audioPackFormat.	Yes
typeDefinition	Should not be present.	—
importance	No special requirement.	Optional

Table 20: audioPackFormat sub-element requirements

Sub-element	Requirements	Quantity
audioChannelFormatIDRef	The referenced audioChannelFormat should have the same typeLabel as the referent audioPackFormat.	0…*
audioPackFormatIDRef	The referenced audioPackFormat should have the same typeLabel as the referent audioPackFormat. The referenced audioPackFormat should not — directly or indirectly — reference back to this audioPackFormat. The nesting level of audioPackFormats should not exceed *audioPackFormatNestingLevel*.	0…*
absoluteDistance	No special requirement.	0 or 1

2.3.5.1 typeLabel==“0002” (Matrix)

Table 21: audioPackFormat sub-element requirements (typeLabel==“0002”)

Element	Requirements	Quantity
encodePackFormatIDRef	Must only reference IDs of typeLabel “0001” (i.e. AP_0001xxxx)	0
decodePackFormatIDRef	Must only reference IDs of typeLabel “0001” (i.e. AP_0001xxxx)	0
inputPackFormatIDRef	Must only reference IDs of typeLabel “0001” (i.e. AP_0001xxxx)	0 or 1
outputPackFormatIDRef	Must only reference IDs of typeLabel “0001” (i.e. AP_0001xxxx)	0 or 1

2.3.5.2 typeLabel==“0004” (HOA)

Table 22: audioPackFormat sub-element requirements (typeLabel==“0004”)

Sub-element	Requirements	Quantity
normalization	No special requirement.	0 or 1
nfcRefDist	No special requirement.	0 or 1
screenRef	No special requirement.	0 or 1

2.3.6 audioChannelFormat

The number of audioChannelFormat elements in an ADM file or flow is restricted to a maximum of *audioChannelFormatCount*. This does not include common definition audioChannelFormat elements.

Table 23: audioChannelFormat attribute requirements

Attribute	Requirements	Required
audioChannelFormatID	No special requirement.	Yes
audioChannelFormatName	Should not exceed *labelLength*.	Yes
typeLabel	The typeLabel should match the typeLabel of the referent audioPackFormat.	Yes
typeDefinition	Should not be used.	–

Table 24: audioChannelFormat sub-element requirements

Sub-element	Requirements	Quantity
audioBlockFormat	Should be ordered according to their rtime. For typeLabel “0001”, “0002”, “0004” and “0005”, there should only be one audioBlockFormat. For typeLabel “0003”, there should be at least 1 audioBlockFormat, and a zero duration audioBlockFormat should be placed before the corresponding audioBlockFormat with the same rtime.	1…*
frequency	Should only be used with the typeDefinition==”lowPass".	0 or 1

2.3.6.1 typeLabel==“0003” (Objects)

Each audioChannelFormat should only be referenced from one audioPackFormat.

2.3.7 audioBlockFormat

Table 25: Common audioBlockFormat attribute requirements

Attribute	Requirements	Required
audioBlockFormatID	Considering the format AB_yyyyxxxx_zzzzzzzz, the zzzzzzzz part should increase strictly monotonically without skipping any values.	Yes
audioChannelFormatName	Should not exceed *labelLength* characters.	Yes
rtime	If used should be 00:00:00.00000 for the first audioBlockFormat.	Optional
duration	Should be either 00:00:00.00000 (for the initial block) or not smaller than the length of a sample of the sample rate being used. The sum of all durations should match the duration of the audioObject which references the audioChannelFormat through audioPackFormat references. The rtime + duration of an audioBlockFormat should match the rtime of the following block.	Optional

2.3.7.1 typeLabel==“0001” (DirectSpeakers)

Table 26: audioBlockFormat sub-element requirements (typeLabel==“0001”) for polar coordinates

Sub-element	Requirements	Quantity
speakerLabel	If a speaker is defined in [BS.2051] the “SP Label” in table 1 of [BS.2051] or the corresponding URI used in the common definitions should be used.	0…*
position coordinate==”azimuth”	May have the attribute screenEdgeLock with either the value “left” or “right”.	1
position coordinate==”azimuth” bound==”min”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”azimuth” bound==”max”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”elevation”	May have the attribute screenEdgeLock with either the value “top” or “bottom”.	1
position coordinate==”elevation” bound==”min”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”elevation” bound==”max”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”distance”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”distance” bound==”min”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”distance” bound==”max”	Should not have the screenEdgeLock attribute.	0 or 1

Table 27: audioBlockFormat sub-element requirements (typeLabel==“0001”) for Cartesian coordinates

Sub-element	Requirements	Quantity
speakerLabel	If a speaker is defined in [BS.2051] the “SP Label” in table 1 of [BS.2051] or the corresponding URI used in the common definitions should be used.	0…*
position coordinate==”X”	May have the attribute screenEdgeLock with either the value “left” or “right”.	1
position coordinate==”X” bound==”min”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”X” bound==”max”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”Y”	Should not have the screenEdgeLock attribute.	1
position coordinate==”Y” bound==”min”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”Y” bound==”max”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”Z”	May have the attribute screenEdgeLock with either the value “top” or “bottom”.	1
position coordinate==”Z” bound==”min”	Should not have the screenEdgeLock attribute.	0 or 1
position coordinate==”Z” bound==”max”	Should not have the screenEdgeLock attribute.	0 or 1

2.3.7.2 typeLabel==“0002” (Matrix)

Table 28: audioBlockFormat sub-element requirements (typeLabel==“0002”)

Sub-element	Requirements	Quantity
outputChannelFormatIDRef	Must only reference IDs of typeLabel “0001” (i.e. AC_0001xxxx)	1[4]
matrix	No special requirements.	1

[4] Mandatory because encodePackFormatIDRef and decodePackFormatIDRef are not allowed in this profile.

Table 29: matrix sub-element requirements

Sub-element	Requirements	Quantity
coefficient	No special requirements.	1…*

Table 30: coefficient sub-element requirements

Attribute	Requirements	Required
gain	No special requirements.	Yes
phase	Should not be used.[5]	—
delay	Should be in the range of 0 ms and 1000 ms.	No
gainVar	Should not be used.	—
phaseVar	Should not be used.	—
delayVar	Should not be used.	—

2.3.7.3 typeLabel==“0003” (Objects)

Either the polar or Cartesian coordinate system can be used for positional and extent parameters within all audioBlockFormats within a audioChannelFormat element. All the parameters within an audioBlockFormat element must be from the same coordinate system.

For the typeDefinition Objects the following table, Table 31, takes precedence over Table 25.

Table 31: audioBlockFormat attribute requirements (typeLabel==“0003”)

Attribute	Requirements	Required
rtime	Should be 00:00:00.00000 for the first audioBlockFormat.	Yes
duration	Should be either 00:00:00.00000 (for the initial block) or not smaller than the length of a sample of sample rate being used. The sum of all durations should match the duration of the audioObject which references the audioChannelFormat through audioPackFormat references. The rtime + duration of an audioBlockFormat should match the rtime of the following block.	Yes

Table 32: audioBlockFormat sub-element requirements (typeLabel==“0003”, polar)

Sub-element	Requirements	Quantity
position coordinate==”azimuth”	May have the attribute screenEdgeLock with either the value “left” or “right”.	1
position coordinate==”elevation”	May have the attribute screenEdgeLock with either the value “top” or “bottom”.	1
position coordinate==”distance”	Should not have the screenEdgeLock attribute.	0 or 1

[5] The necessary signal processing for a phase shift is ambiguous.

width	No special requirement.	0 or 1
height	No special requirement.	0 or 1
depth	No special requirement.	0 or 1

Table 33: audioBlockFormat sub-element requirements (typeLabel==“0003”, Cartesian)

Sub-element	Requirements	Quantity
position coordinate==”X”	May have the attribute screenEdgeLock with either the value “left” or “right”.	1
position coordinate==”Y”	Should not have the screenEdgeLock attribute.	1
position coordinate==”Z”	May have the attribute screenEdgeLock with either the value “top” or “bottom”.	0 or 1
width	No special requirement.	0 or 1
height	No special requirement.	0 or 1
depth	No special requirement.	0 or 1

Table 34: audioBlockFormat sub-element requirements (typeLabel==“0003”, common)

Sub-element	Requirements	Quantity
cartesian	No special requirement.	0 or 1
gain	No special requirement.	0 or 1
diffuse	No special requirement.	0 or 1
channelLock	No special requirement.	0 or 1
objectDivergence	No special requirement.	0 or 1
jumpPosition	The interpolationLength should not be bigger than the duration of the audioBlockFormat. If not set for the first audioBlockFormat the default value for jumpPosition is 1 and for interpolationLength 0.0. For all the other audioBlockFormats the default value for jumpPosition is 0.	0 or 1
zoneExclusion	No special requirement.	0 or 1
screenRef	No special requirement.	0 or 1
importance	Should not be used or set to 10.	0 or 1

2.3.7.4 typeLabel==“0004” (HOA)

Table 35: audioBlockFormat sub-element requirements (typeLabel==“0004”)

Sub-element	Requirements	Quantity
equation	Should not be used.	–
Order	Each audioBlockFormat within a single audioPackFormat should have a unique order/degree pair.	1
Degree	Each audioBlockFormat within a single audioPackFormat should have a unique order/degree pair.	1
normalization	Should not be used.	–
nfcRefDist	Should not be used.	–
screenRef	Should not be used.	–

2.3.7.5 typeLabel==“0005” (Binaural)

No special requirement.

2.3.8 audioStreamFormat

The number of audioStreamFormat elements in an ADM file or flow is restricted to a maximum of *audioStreamFormatCount*. This does not include common definition audioStreamFormat elements.

Table 36: audioStreamFormat attribute requirements

Attribute	Requirements	Required
audioStreamFormatID	No special requirement.	Yes
audioStreamFormatName	Should not exceed *labelLength* characters.	Yes
formatLabel	Should be set to “0001”.	Yes
formatDefinition	Should not be used.	–

Table 37: audioStreamFormat sub-element requirements

Sub-element	Requirements	Quantity
audioChannelFormatIDRef	No special requirement.	1
audioPackFormatIDRef	Should not be used.	0
audioTrackFormatIDRef	Should not be used. [6]	0

[6] This profile ignores that this reference is mandatory in BS.2076-1, as it is an obvious mistake.

2.3.9 audioTrackFormat

The number of audioTrackFormat elements in an ADM file or flow is restricted to a maximum of *audioTrackFormatCount*. This does not include common definition audioTrackFormat elements.

Table 38: audioTrackFormat attribute requirements

Attribute	Requirements	Required
audioTrackFormatID	The zz part considering the format AT_xxxxyyyy_zz should be set to 0x01.	Yes
audioTrackFormatName	Should not exceed *labelLength* characters.	Yes
formatLabel	Should be set to “0001”.	Yes
formatDefinition	Should not be used.	—

Table 39: audioTrackFormat sub-element requirements

Sub-element	Requirements	Quantity
audioStreamFormatIDRef	No special requirement.	1

2.3.10 audioFormatExtended

Table 40: audioFormatExtended attribute requirements

Attribute	Requirements	Required
version	Should be set to “ITU-R_BS.2076-1”	Yes

Table 41: audioFormatExtended sub-element requirements

Sub-element	Requirements	Quantity
audioProgramme	No special requirement.	1…*
audioContent	No special requirement.	1…*
audioObject	No special requirement.	1…*
audioTrackUID	No special requirement.	1…*
audioPackFormat	No special requirement.	0…*
audioChannelFormat	No special requirement.	0…*
audioStreamFormat	No special requirement.	0…*
audioTrackFormat	No special requirement.	0…*

2.4 File-based specific

The requirements listed in this section only apply to file-based systems, such as the BW64 file format.

The number of tracks in the audio file should be limited to *trackCount*.

For BW64 files the audioMXFLookUp sub-element (see Table 15) should not be present. For MXF files it should be present.

2.5 Nesting of elements

Both audioObject and audioPackFormat elements can be nested, in other words they can reference other audioObject and audioPackFormat elements respectively. In the common definitions [BS.2094], the HOA audioPackFormats use nesting to group the different HOA orders.

The number of nesting levels refers to the number of generations of element that reference each other. So, two nesting levels is a parent element referencing one or more child elements. Three nesting levels would have the child elements having grandchild elements referenced from them. If the number nesting levels is restricted to one, then audioObjects can't reference any other audioObjects (likewise for audioPackFormats). The diagram in Figure 2 shows the simplest structure for three level nesting of audioObject elements. In this example audioObject 21 would then reference an audioPackFormat and some audioTrackUIDs; audioObject 1 would be referenced from an audioContent element.

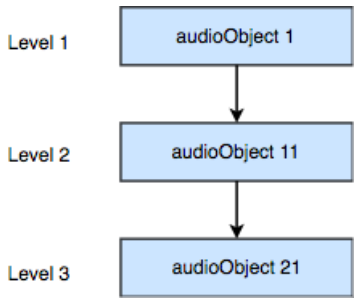

Figure 2: Simplest three level nesting.

More complex trees can be generated, that are considered to be three levels, such as having multiple grandchildren as shown in Figure 3. A more complex example is shown in Figure 4 where audioObject 22 has three parents, one of which is at level 1 and the other two at level 2.

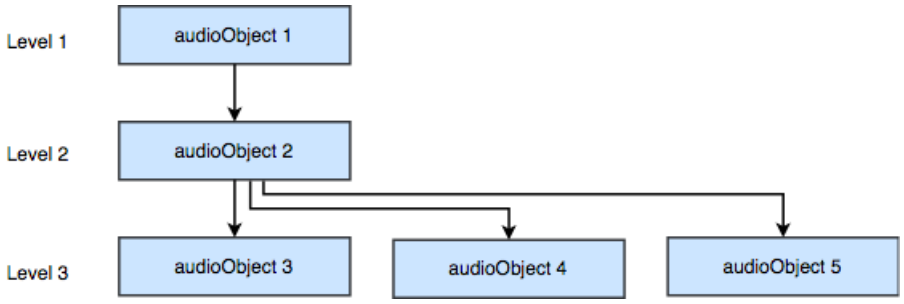

Figure 3: Three grandchildren.

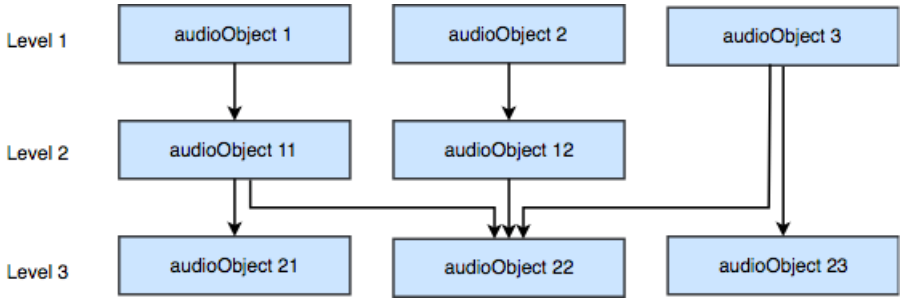

Figure 4: More complex three level structure

Not all structures are valid, as elements that reference themselves are not allowed as shown in Figure 5. Any reference path that produce a loop are also invalid such as the one shown in Figure 6.

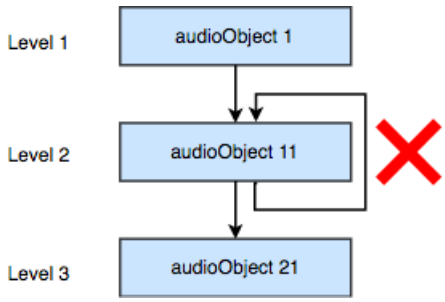

Figure 5: Invalid self-referencing element.

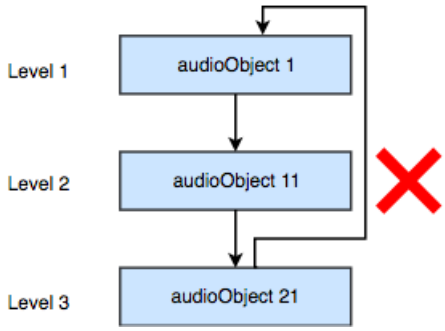

Figure 6: Invalid looped reference.

3. Transcoding instructions

If an ADM file that is not constrained by this profile needs to be converted to a profile conformant file the instructions described in this section can be used.

3.1 audioProgramme

If there is no audioProgramme element present, then a new one should be generated. Any audioContent elements present should be referenced from this new audioProgramme element.

A missing start attribute should be set to 00:00:00.00000. A missing end attribute should be set to start plus the duration of the file.

3.2 audioContent

If there is no audioContent element present, then a new one should be generated. It should be referenced from the audioProgramme element and reference all the audioObjects present.

3.3 audioObject

If an audioObject refers to both — audioPackFormats *and* audioObjects — another audioObject should be added which groups the referenced audioPackFormats.

A missing start attribute should be set to 00:00:00.00000. A missing duration attribute should be set to the duration of the file minus the start of the audioObject.

3.4 audioTrackUID

No instructions necessary.

3.5 audioPackFormat

If an audioPackFormat only contains the typeDefinition attribute the corresponding typeLabel should be added and the typeDefinition removed. If both are present the typeDefinition should be removed.

If an audioPackFormat refers to both audioPackFormats *and* audioChannelFormats then another audioPackFormat should be added which groups the referenced audioChannelFormats.

3.6 audioChannelFormat

If an audioChannelFormat only contains the typeDefinition attribute the corresponding typeLabel should be added and the typeDefinition removed. If both are present the typeDefinition should be removed.

3.7 audioBlockFormat

If an audioChannelFormat of typeLabel "0003" (Objects) is present and it contains only one audioBlockFormat, then an initial zero-duration audioBlockFormat should be inserted before it with the same parameters.

3.8 audioStreamFormat

If an audioStreamFormat only contains the formatDefinition attribute the corresponding formatLabel should be added and the formatDefinition removed. If both are present the formatDefinition should be removed.

If the formatLabel is not "0001" (PCM), and therefore the audio is not PCM, then a suitable conversion/decode should be carried out on the audio data to convert it to PCM samples.

3.9 audioTrackFormat

If an audioTrackFormat only contains the formatDefinition attribute the corresponding formatLabel should be added and the formatDefinition removed. If both are present the formatDefinition should be removed.

If the audioStreamFormatIDRef is missing the audioStreamFormatID of the audioStreamFormat which refers to it should be added.

If the formatLabel is not "0001" (PCM), and therefore the audio is not PCM, then a suitable conversion/decode should be carried out on the audio data to convert it to PCM samples.

3.10 Converting between Type Combinations

If the ADM file being read is of a higher type combination than the desired type combination, then conversion of some elements may be required. These conversions are assuming that the desired approach of the converter is to retain all of the audio content. Converters ought to give warnings of the conversions being made to inform the user, and thus allow them to intervene should they desire.

These conversions will generally be lossy, where quality will be compromised. Therefore, they are considered to be bare minimum approaches to conversions while retaining the audio signals. If more sophisticated conversions are available, then they can be used. As most type combinations will contain the DirectSpeakers type (typeLabel "0001"), then any unsupported types should be converted to DirectSpeakers:

- Objects to DirectSpeakers: Render (using EAR [EBU3388]) to an appropriate common definition DirectSpeakers audioPackFormat layout.
- HOA to DirectSpeakers: Render (using EAR [EBU3388]) to an appropriate common definition DirectSpeakers audioPackFormat layout.
- Matrix to DirectSpeakers: Apply the matrix defined to convert to the audioPackFormat defined in the outputAudioPackFormat.
- Binaural to DirectSpeakers: Convert directly to the 'stereo' common definition audioPackFormat (AP_00010002). Please note that this is a process which does not preserve spatial imaging.

The choice of target audioPackFormat layout depends on the number of tracks that are available in the target file and the nature of the source layout, but usually the greater number of channels (therefore, generally the spatial resolution) in the audioPackFormat the better.

4. References

[BS.2076] ITU-R Recommendation BS.2076-1: Audio Definition Model.
https://www.itu.int/dms_pubrec/itu-r/rec/bs/R-REC-BS.2076-1-201706-I!!PDF-E.pdf

[BS.2051] ITU-R Recommendation BS.2051-1: Advanced sound system for programme production
https://www.itu.int/dms_pubrec/itu-r/rec/bs/R-REC-BS.2051-1-201706-I!!PDF-E.pdf

[BS.2094] ITU-R Recommendation BS.2094-1: Common definitions for the Audio Definition Model
https://www.itu.int/dms_pubrec/itu-r/rec/bs/R-REC-BS.2094-1-201706-I!!PDF-E.pdf

[BS.2088] ITU-R Recommendation BS.2088-0: Long-form file format for the international exchange of audio programme materials with metadata.
https://www.itu.int/dms_pubrec/itu-r/rec/bs/R-REC-BS.2088-0-201510-I!!PDF-E.pdf

[BS.2388] ITU-R Recommendation BS.2388-3: Usage Guidelines for the Audio Definition Model and Multichannel Audio Files
https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-BS.2388-3-2018-PDF-E.pdf

[EBUTR42] EBU TR 042: Example of an end-to-end OBA broadcast architecture and workflow
https://tech.ebu.ch/files/live/sites/tech/files/shared/techreports/tr042.pdf

[EBU3388] EBU TECH 3388: ADM Renderer for use in NGA Broadcasting
https://tech.ebu.ch/files/live/sites/tech/files/shared/tech/tech3388.pdf

Note: All links are current at time of publishing this document.