Common Media Manifest Metadata

CONTENTS

REVISION HISTORY

Version	Date	Description
1.0	July 1, 2014	Initial version.
1.1	September 10, 2014	Added updateNum to Experience-Type Allowed multiple instances of SubType in Audiovisual-type Added FileDeleteManifest and MediaManifestEdit to support delivery use cases.
1.2	October 14, 2014	Schema-only update to reference Common Metadata 2.2.
1.3	February 10, 2015	Reference Common Metadata 2.2 to incorporate latest HDR metadata and Ancillary tracks Added means to encode metadata in the Inventory to reduce redundancy Updated Audiovisual to support content-less deliveries (i.e., pre-sale)

1 INTRODUCTION

This specification seeks to enhance the user experience associated with movie and television viewing, while improving supply chain efficiency. The approach taken is to define data models with unambiguous encoding definitions that define the interrelationships between media elements.

This may sound like a small thing, but it isn't. With a modest amount of data, one can move from a collection of miscellaneous video tracks, audio tracks, subtitle tracks, applications, metadata, and text objects to full and rich user experiences. Furthermore, it facilitates building those experiences with the minimum amount of duplication, fewer assets in the supply chain and the most flexibility for the user.

This model defined in this specification applies to the entire distribution workflow from studio publication through consumption by a consumer, although it may be used at any point through the lifecycle.

The model describes the interrelationship between various abstract and physical media components. It is intended to provide considerable flexibility in how content is published while maintaining the authoring intent through the content's distribution path.

Some features of this model are

- The logical relationships between assets are distinguished from their physical form and location making it possible to express a full user experience with relatively simple data.
- Late Binding is supported to minimize the number of assets in the supply chain while maximizing the ability to produce localized or user-customized products.
- Publication of adaptive streaming (e.g., DASH) components is supported to avoid duplicative encoding of stream data.
- Accessibility and language diversity is supported through deterministic default track selection
- Supplemental (value added) material is supported.
- Playback features, such as chapters, are supported

This specification is designed as a resource. Those using this specification may extend the definition with additional data element specific for their needs. Although adopting bodies may choose to constrain this specification for their needs, for interoperability all are highly encouraged to use the data elements exactly as defined.

Media Manifest is part of the Common Metadata family of specifications.

1.1 Overview

1.1.1 Scope

This specification defines the underlying metadata structure for media presented to a user without defining the presentation itself. The Media Manifest can be used in either a B2B interface, such as between as studio and a streaming site, or in a B2C interface such as found on a tablet or home media player.

This specification also includes a definition of a packaging manifest to convey information about file delivery.

This model is defined such that it can be incorporated into other specifications. As with Common Metadata and other MovieLabs documents, we expect that the model defined here will be adapted for specific applications.

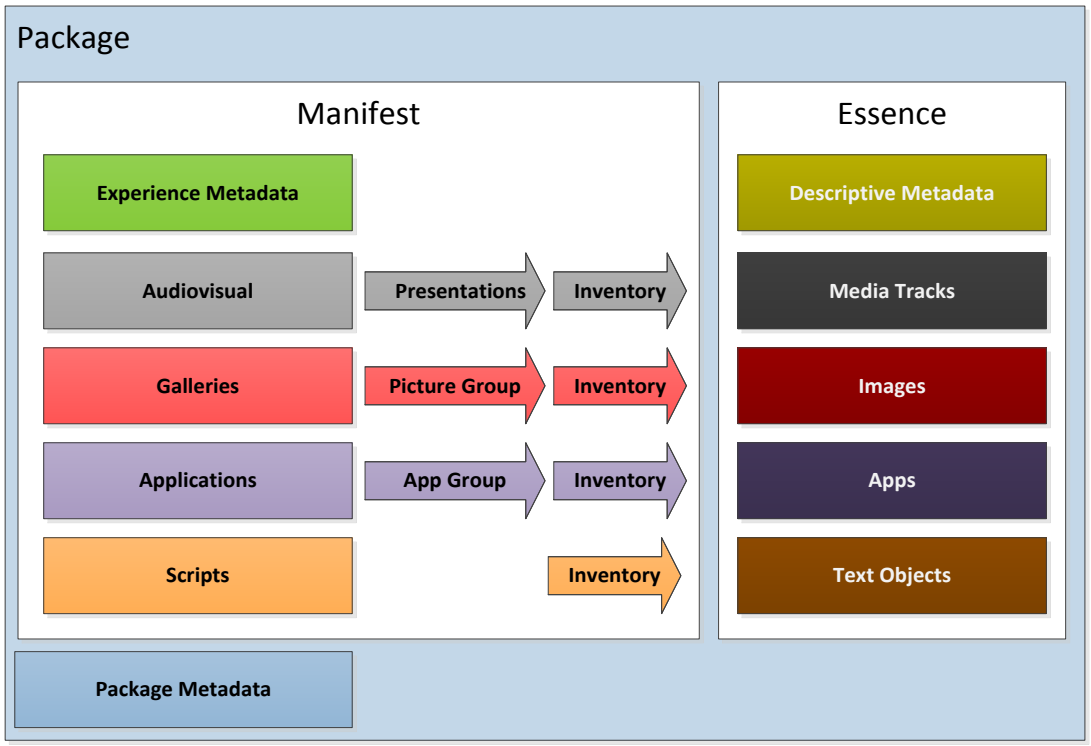
1.1.2 Media Manifest and Package

The Media Manifest architecture has the following data objects

- Compatibility – What level of functionality is required to use this document
- Inventory – Available media components. This includes metadata. Note that media components can be referenced as part of a package (container) or remotely on the Internet.
- Presentations – Set of tracks can be played concurrent; such as, related video, primary audio, commentary audio and subtitles. This includes information about track selection and chapters.
- Playable Sequence – Sequences of Presentations with entry and exit points. This is a simple composition playlist allowing video pieces to be strung together (e.g., distribution cards, followed by anti-piracy cards, followed by a main title).
- Image Grouping – Which images comprise a gallery
- Experience – Defines a collection consisting of one main title, and zero or more supplemental titles, Galleries, interactive applications and text objects (e.g., scripts).

From these components a user experience can be created. A player starts with the Experience and discovers a main title, alternate titles and picture galleries. From this the application can create the user experience including title selection, default track selection, gallery display, chapter selection, and so forth.

The following illustrates the various components specified in this document.



1.1.3 Relationship of Media Manifest to Common Metadata

Media Manifest is an extension to Common Metadata and may be used in conjunction with Common Metadata.

Common Metadata includes elements that cover typical definitions of media, particularly movies and television. Common Metadata has two parts: Basic Metadata and Digital Asset Metadata. Basic Metadata includes descriptions such as title and artists. It describes information about the work independent of encoding. Digital Asset metadata describes information about individual encoded audio, video and subtitle streams, and other media included. Ratings and Parental Control information is described.

Common Metadata is designed to provide definitions to be inserted into other metadata systems. A given metadata scheme, for example, the Entertainment Merchant's Association (EMA) may select element of the Common Metadata to be used within its definitions. EMA would then define additional metadata to cover areas not included in Common Metadata.

1.2 Document Organization

This document is organized as follows:

1. Introduction—Provides background, scope and conventions
2. Encoding

3. Media Manifest
4. Inventory
5. Presentations and Playable Sequences
6. Picture Groups and Galleries
7. Experience
8. Delivery Package
9. Annex A: Track Selection Algorithm

1.3 Document Notation and Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119]. That is:

- “MUST”, “REQUIRED” or “SHALL”, mean that the definition is an absolute requirement of the specification.
- “MUST NOT” or “SHALL NOT” means that the definition is an absolute prohibition of the specification.
- “SHOULD” or “RECOMMENDED” mean that there may be valid reasons to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- “SHOULD NOT” or “NOT RECOMMENDED” mean that there may be valid reasons when the particular behavior is acceptable, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- “MAY” or “OPTIONAL” mean the item is truly optional; however a preferred implementation may be specified for OPTIONAL features to improve interoperability.

Terms defined to have a specific meaning within this specification will be capitalized, e.g. “Track”, and should be interpreted with their general meaning if not capitalized.

Normative key words are written in all caps; e.g., “SHALL”.

1.3.1 XML Conventions

XML is used extensively in this document to describe data. It does not necessarily imply that actual data exchanged will be in XML. For example, JSON may be used equivalently. Adopting bodies should pick one encoding. XML is preferred.

This document uses tables to define XML structure. These tables may combine multiple elements and attributes in a single table. Although this does not align with schema structure, it is much more readable and hence easier to review and to implement.

Although the tables are less exact than XSD, the tables should not conflict with the schema. Such contradictions should be noted as errors and corrected.

1.3.1.1 Naming Conventions

This section describes naming conventions for Common Metadata XML attributes, element and other named entities. The conventions are as follows:

- Names use initial caps, as in InitialCaps.
- Elements begin with a capital letter, as in InitialCapitalElement.
- Attributes begin with a lowercase letter, as in initialLowercaseAttribute.
- XML structures are formatted as Courier New, such as `md:ContentID-type`
- Names of both simple and complex types are followed with "-type"

1.3.1.2 Structure of Element Table

Each section begins with an information introduction. For example, "The Bin Element describes the unique case information assigned to the notice."

This is followed by a table with the following structure.

The headings are

- Element—the name of the element.
- Attribute—the name of the attribute
- Definition—a descriptive definition. The definition may define conditions of usage or other constraints.
- Value—the format of the attribute or element. Value may be an XML type (e.g., "string") or a reference to another element description (e.g., "See Bar Element"). Annotations for limits or enumerations may be included (e.g.," int [0..100]" to indicate an XML xs:int type with an accepted range from 1 to 100 inclusively)
- Card—cardinality of the element. If blank, then it is 1. Other typical values are 0..1 (optional), 1..n and 0..n.

The first row of the table after the header is the element being defined. This is immediately followed by attributes of this element, if any. Subsequent rows are child elements and their attributes. All child elements (i.e., those that are direct descendants) are included in the table. Simple child elements may be fully defined here (e.g., "Title", " ", "Title of work", "xs:string"), or described fully elsewhere ("POC", " ", "Person to contact in case there is a problem",

"md:ContactInfo-type"). In this example, if POC was to be defined by a complex type defined as md:ContactInfo-type. Attributes immediately follow the containing element.

Accompanying the table is as much normative explanation as appropriate to fully define the element, and potentially examples for clarity. Examples and other informative descriptive text may follow. XML examples are included toward the end of the document and the referenced web sites.

1.3.2 Use of Language

This specification assumes that Devices have a parameter referred to here as System Language. The System Language is the current setting for the Device's interface language, perhaps set by the User. Users may also make independent language preference selections for audio language and for subtitle language.

Language preferences such as System Language are expressed as at least one language tag as per [RFC5646] and included in [IANA-LANG], possibly prioritized as a Language Priority List as per [RFC4647], Section 2.3. The assumed Priority List consists of at least the following language ranges:

1) The fully enumerated language tag including region, dialect or any other subtag element. For example, this would be a language tag from System Language, Audio User preference or Subtitle User preference.

2) The language tag from the first entry trimmed to the primary language tag, followed by a wildcard '*' subtag.

For example if the language is "en-GB", the Priority List will be "en-GB, en-*".

The best language match between a language preference (e.g., System Language) and one or more languages in a list (e.g., language tags in a list of audio tracks) is to be done in accordance with [RFC4647], Section 3.4 "Lookup".

1.3.3 General Notes

All required elements and attributes must be included.

When enumerations are provided in the form 'enumeration', the quotation marks ('') should not be included.

The term "Device" refers to an entity playing the interactive material specified here. It may be a standalone physical device, such as a Blu-ray player, or it might be an application running on a general purpose computer, a table, phone or as part of another device. The term 'User' refers to the person using the Device.

1.4 Normative References

[CM]	Common Metadata, www.movielabs.com/md/md
[Avail]	Avails, www.movielabs.com/md/avails
[MEC]	Media Entertainment Core, www.movielabs.com/md/mec
[RFC4646]	Philips, A, et al, *RFC 4646, Tags for Identifying Languages*, IETF, September, 2006. www.movielabs.com/md/avails
[RFC5646]	Philips, A, et al, *RFC 5646, Tags for Identifying Languages*, IETF, September, 2009. www.movielabs.com/md/mec
[IANA-LANG]	IANA Language Subtag Registry. http://www.iana.org/assignments/language-subtag-registry
[ISO639]	ISO 639-2 Registration Authority, Library of Congress. http://www.loc.gov/standards/iso639-2/
[ISO3166-1]	Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes, 2007.
[ISO3166-2]	ISO 3166-2:2007Codes for the representation of names of countries and their subdivisions -- Part 2: Country subdivision code
[ISO4217]	Currency shall be encoded using ISO 4217 Alphabetic Code. http://www.iso.org/iso/currency_codes_list-1
[ISO8601]	ISO 8601:2000 Second Edition, *Representation of dates and times, second edition*, 2000-12-15.
[TTML]	Timed Text Markup Language (TTML) 1.0, W3C Proposed Recommendation 14 September 2010, http://www.w3.org/TR/ttaf1-dfxp/

1.5 Informative References

[DASH]	ISO/IEC 23009-1:2011, "Dynamic Adaptive Streaming over HTTP"
[DMP]	DECE Media Package (DMP) Specification, Version 1.1, November 14, 2013, www.uvvuwiki.com
[DMedia]	DECE Common File Format and Media Specification, www.uvvuwiki.com
[DOI]	Digital Object Identifier (DOI), www.doi.org
[EIDR-UG]	EIDR 2.0 Registry User's Guide, eidr.org/technology/

[EIDR-ID]	EIDR ID Format, eidr.org/technology/
[ISO26324]	ISO 26324:2012, *Information and documentation -- Digital object identifier system*

1.6 Using Manifest in Other Specifications

As with all specifications in the Common Metadata family, this specification is designed to be referenced by other specifications. It can be used in whole or in part. Certain features were added to the structure to support partial use of this specification.

1.6.1 Manifest as Components

The Manifest is designed so that many individual pieces can be individually incorporated into other designs and specifications. For example, the Inventory is a general means of describing and referencing media components and can be used independently. Similarly, one may choose to use Media Manifest, but not File Manifest. This is completely within the intent of the specification.

Some independent uses are not as obvious. For example, one could use Experience, AppGroups, PictureGroups and PlayableSequences, but not use Presentations or Inventory. These are required elements at the MediaManifest level, but there are individual types defined for each of the above for inclusion in other specifications: Experience-type, AppGroup-type, PictureGroup-type, and so forth.

There are specific features designed into the elements to support this model. Most notable is heavy use of referencing objects by identifier (see following section). There are also some options embedded in the elements to support different uses. For example, the Audiovisual element (part of Experience) allows reference to playable media via PresentationID (bypassing PlayableSequence) and PlayableSequenceID. It also allows PlayableSequence to be included in-place for other implementation. It is unlikely any implementation would wish to support all three, so it is recommended that the referencing specification clarify which options are allows.

One example of use is the Common Media Package (CMP) defined in DECE. The CMP uses the top-level Media Manifest elements, such as Experience, AppGroup and PictureGroup, but uses other definitions for Presentation. The two are linked with PresentationID.

1.6.2 Reference by Identifier

To facilitate the standalone usage of major structures, they are referenced by identifier. For example, a Playable Sequence refers to a Presentation by PresentationID, and a Presentation refers to video in Inventory using VideoID.

When multiple Manifest objects are used together, the references are defined in this document. However, that does not preclude the use of external references as long as those references can be resolved.

For example, UltraViolet has its own Presentation structure. The concepts are the same, but the implementation is slightly different. So, when UltraViolet uses the Manifest, it still refers to Presentation by PresentationID, it's just referring to an UltraViolet Presentation.

1.6.3 Scope of Identifiers

There are two types of identifier in the Manifest. Local IDs and generalized identifiers.

Local identifiers are defined as md:id-type (see Section 2.4 for a list of these). These are for referencing from one part of the Manifest to another. The scope is within the Manifest. The exact scope is defined by the author, but generally the must be unique only within the scope of the manifest itself. Implementations that use parts of the Manifest specification will have to define the scope of identifiers that cross from the Manifest-defined part to the custom part.

The manifest uses a general identifier structure for identifiers that are intended as external references. The type, md:ContentIdentifier-type includes Namespace, Identifier and an optional Location. Namespace defines the identifier scheme—there is a standard definition in Common Metadata [CM]. Identifier is the identifier itself. Location is a resolvable location reference for the identifier, such as the http form of an EIDR ID [EIDR-ID] or DOI [DOI]. In the schema, there can be multiple instances of this triple to allow multiple external identifiers to be used.

General identifiers are assumed to be unique within scope of the the identifier scheme (Namespace).

1.7 Manifest Update

The Manifest is a single cohesive object. As defined in this specification, it is not possible to update one portion. In general, this is the best practice because it becomes exponentially more difficult to keep a Manifest consistent if individual sections are update.

We recommend Manifests be updated by replacement. An updated Manifest will have an updated Manifest/@updateNum that indicates that it is the latest version. It is always possible to compare updateNum to determine the more current manifest.

Under carefully controlled circumstances it is possible for a Manifest to include just changes from a previously delivered Manifest. Manifest/@updateDeliveryType must be included and be populated with a string that defines the rule set used for the update. For example, if there was a practice that allowed a localization update, updateDeliveryType could include a string such as 'LocalizationUpdateModel1'.

In general, updates are mapped using identifiers. If a new object is added, it would have a new identifier. If an object is modified the update with have the same identifier as the original. Updates must be applied in the correct order.

Experience/@version can be used uniquely identify a version of an Experience as identified by @ExperienceID. One could supply a Manifest with replacement Experience elements and the recipient could resolve what is more current. This does not support Experience removal. It also requires the recipient to follow references from Experience to other components (e.g., PlayableSequence, PictureGroup, Presentatation and Inventory) to determine what is current. I should be assumed that if one of these objects has the same ID as one delivered previously, it is an update. Again, it will be very difficult to do this correctly.

If one carefully manages delivery, one could make the assumption that an object with an ID replaces an object sent earlier with the same ID. For example, if an image is delivered with a particular ImageID, then a new Manifest has another image with the same ImageID, the first one can be replaced.

When using selected portions of this specification, it is possible to define application-specific models for updates. In general, versions must be assigned for the smallest granule that can be updated.

2 ENCODING

This section defines how the XML document elements and attributes are encoded.

2.1 Identifiers

Common Metadata identifiers are used as defined in [CM], Section 2.

The following additional identifiers are used in the Media Manifest:

- ExperienceID – Identifies and Experience
- PlayableSequenceID – identifies a Playable Sequence
- PictureGroupID – identifies a Picture Group
- PresentationID – identifies a Presentation.
- VideoTrackID, AudioTrackID, SubtitleTrackID – Identifies a video, audio or subtitle track from inventory (in its entirety)
- AncillaryTrackID – Identifies Ancillary tracks (i.e., those that enhance or supplement another track).
- ImageID – Identifies an Image (just the image)
- PictureID – Identifies a Picture (Image plus description on its use)
- GalleryID – Identifies a Gallery. Reserved for future use.

2.2 Asset References

It is necessary to reference assets at the file level (e.g., A/V files, audio files, image files) and at the track level (e.g., audio, video, subtitle, etc.).

The exact reference depends on how the assets are packaged and therefore the reference encodings will be appropriate to that packaging.

2.2.1 Location (Location-type)

Location-type provides a means of locating assets. Location-type is encoded as a URI with the following constraints:

- Local files are of the form: `file://<filename>` where `<filename>` is the name of the physical file.
- Internet files are of the form `<filelocation>` where `<filelocation>` is the fully enumerated URL for the file.

2.2.2 ContainerReference-type

This provides the means to reference containers, both locally and remotely. It can also indicate if the container is within another container (e.g., a media file within a ZIP file, or a UltraViolet DCC within a DECE Media Package).

Element	Attribute	Definition	Value	Card.
ContainerReference-type				
ContainerLocation		Location of the Container. Either a network location or a local filename.	manifest:Location-type	0..1
ParentContainer		If the container is contained within another container (e.g., a file within a ZIP file), the ParentContainer element describes the encoding container. This is recursive.	manifest:ContainerReference-type	0..1
ContainerIdentifier		Identifiers for the Container. Note that this would be an identifier, such as an EIDR ID, not a filename.	md:ContentIdentifier-type	0..n
Length		Length of Container in bytes	xs:nonNegativeInteger	0..1
Hash		Hash of Container. More than one instance can be included if using different methods.	md:Hash-type	0..n

ContainerIdentifier refers to a Container regardless of location. Assuming a player has a means of translating an identifier to a location, this is the preferred method. Namespace is either the identifier scheme as defined in [CM] or other relevant specification. Note that Container/Identifier is used for the location of the identifier, such as an EIDR in URL form [EIDR-ID], and is not the location of the Container.

If ContainerIdentifier is absent or insufficient for locating a container, ContainerLocation provide information. ContainerLocation is encoded as Location-type.

Length and Hash are intended for file validation, such as when the Inventory is used in conjunction with file delivery.

2.2.3 Referencing Tracks

Tracks must be referenced using TrackReference within the track definition in conjunction with an element defined using ContainerReference-type in a manner that is unambiguous and allows the player to locate that track.

The following sections instruct how to reference tracks in various scenarios.

2.2.3.1 TrackReference and TrackIdentifier

TrackReference and/or TrackIdentifier are used to identify tracks regardless of their location. In some cases, they are sufficient. In other cases they must be used in conjunction with ContainerReference.

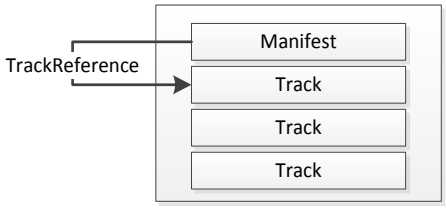
In containers that support explicit track identification, TrackReference identifies the track. For example, in an UltraViolet DCC, TrackReference is the track_ID of the track in question. When track is implicit, TrackReference can be excluded. For example, if there is only one track in the container, or there is only one track of the referencing media type then TrackReference may not be necessary. Note that specific use depends on the type of container reference.

If the track is labeled with an identifier, TrackIdentifier contains that identifier. For example, an EIDR manifestation ID can uniquely identify a track.

2.2.3.2 Referencing track collocated with referencing element

If the reference and track are in the same container, TrackReference and/or TrackIdentifier are sufficient.

The following figure illustrates in an internal track reference.

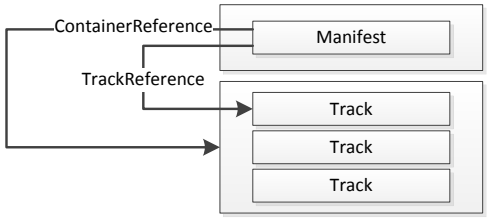


2.2.3.3 Referencing track within local Container

When the track is collocated with the Manifest, it can be referenced with a combination TrackReference/TrackIdentifier in conjunction with ContainerReference.

Local containers can be referenced with identifiers, filenames or both. To reference by identifier, use ContainerReference/ContainerIdentifier. To reference by filename use ContainerReference/ContainerLocation.

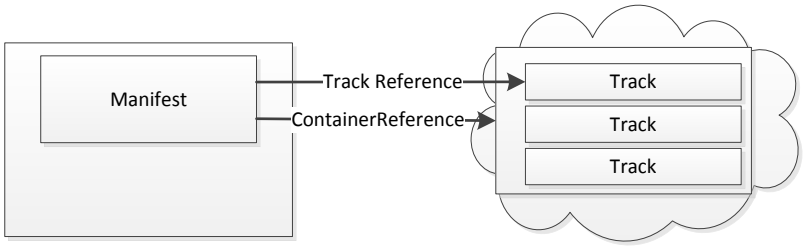
The following illustration shows ContainerReference (either Identifier or Location) referencing the container. As discussed above, TrackReference or TrackIdentifier is required to identify the Track.



In some conditions, one could avoid the ContainerReference because TrackIdentifier would be unambiguous. However, this would be a limited to containers that allow both containers and tracks to be located by track identifier.

2.2.3.4 Referencing track in online container

An online container is reference the same as a local track except ContainerReference/ContainerLocation contains a URL referencing a remote location, or ContainerReference/ContainerIdentifier identifies is resolvable to an online location.



2.2.3.5 Referencing a track in an container within another container

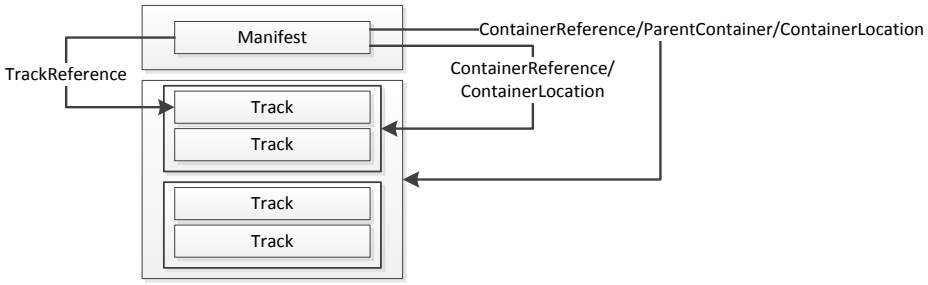
Frequently tracks will be within a container (e.g., a media file) that is itself within another container (e.g., a ZIP file).

ContainerReference is recursive allowing reference to containers within containers ad infinitum. The first reference is the innermost container with each ParentContainer referring to outer containers.

For example, if referring to an UltraViolet DCC file (media container) with a DECE Media Package (DMP ZIP file) [DMP], ContainerReference/ContainerLocation refers to the DCC file, and ContainerReference/ParentContainer/ContainerLocation refers to the DMP.

As noted above, containers can be referenced by location, identifier or both.

The following illustration shows a track within a container that is in turn within another container. As the structure is recursive, additional outer containers could be referenced as well.



Note that locations can be local or online. In the above example, the outer container would be referenced online and the inner container would be a local reference (e.g., filename) within the outer container.

2.2.3.6 Combinations

Although it's necessarily typical, local and remote references can be mixed; and internal track references and external container references can be mixed.

2.3 General Types Encoding

General Types Encoding is as per Common Metadata [CM], Section 3.

2.3.1 Timecode Encoding

Timecode references a specific time in an audio, video or subtitle track.

Element	Attribute	Definition	Value	Card.
Timecode-type				
Timecode		Timecode for referenced point in an associated track.	manifest:TimecodePattern-type	
	dropframe	Is timeframe dropframe used	xs:boolean	0..1

manifest:TimecodePattern-type is xs:string with pattern '[0-9]+\.[0-9]+'

Timecode corresponds with a constrained form of the 'offset-time' syntax (without the metric field) of the media timebase defined in [TTML], Section 10.3.1, and corresponds with the referenced video subtitle and/or audio tracks. The metric is in units of seconds.

In the case of a rounding error that doesn't result in an integer number of frames, the video and/or audio frame(s) Timecode refers to shall be the next decodable frame after the time in the media referenced by this value. For example, in a 30fps progressive video track, 0.1 = the 3rd frame. 0.101 = the 4th frame.

Note that the in some cases implementations will converted the fixed-point Timecode into floating point prior to performing calculations, potentially introducing rounding errors. Since decoding will round up, it is safest to represent non-integer timecodes with a value less than the precise frame time to ensure the correct frame will be chosen during decoding.

Encoding for dropframe is as follows:

- 'Drop' – Drop frame SMPTE timecode is used.
- 'Non-Drop' – Other timecode without drop frame
- 'EBU' – AES/EBU embedded timecome
- 'Other' – Other timecode

2.4 Simple Types

The following identifier simple types are used in Common Extras. All types are md:id-type:

- PlayableSequenceID-type
- AppGroupID-type
- PictureGroupID-type
- PresentationID-type
- VideoTrackID-type
- AudioTrackID-type
- SubtitleTrackID-type
- InteractiveTrackID-type
- AncillaryTrackID-type
- ImageID-type
- PictureID-type

3 MEDIA MANIFEST

The MediaManifest element is the top level definition of a set of Experiences.

Element	Attribute	Definition	Value	Card.
MediaManifest-type				
	ManifestID	Unique identifier for this manifest	md:id-type	0..1
	updateNum	Version of this document. Initial release should be 1. This is a value assigned by the manifest creator that should only be incremented if a new version of manifest is released. If absent, 1 is to be assumed.	xs:integer	0..1
	ExtraVersionReference	A string that describes the version of the extras.	xs:string	0..1
	updateDeliveryType	This indicates the Manifest includes just portions required for an updated. It is not a complete Manifest. The exact definition is subject to specific practices and is reference by this string.	xs:string	0..1
Compatibility		Indicates which versions of Devices can fully use this instance of Extras.	manifest:Compatibility-type	
Inventory		Inventory of audio, video, subtitle, image, and interactivity assets, regardless of where they are stored.	manifest:Inventory-type	
Presentations		Groups of tracks that are intended to be played together. Also includes information about chapters and track selection.	manifest:PresentationList-type	
PlayableSequences		Defines ordered sequences of Presentations that are intended to be played together.	manifest:PlayableSequenceList-type	0..1
PictureGroups		Collections of related images.	manifest:PictureGroupList-type	0..1

AppGroups		Collections of interactive applications that perform the same functions, but for different target systems.	manifest:AppGroupList-type	0..1
Experiences		Defines how Playable Sequences, Presentations and PictureGroups are to be presented to a User.	manifest:ExperienceList-type	
ALIDExperienceMaps		Maps ALIDs, typically from Avails, to Experience elements.	Manifest:ALIDExperienceMap List-type	0..1

3.1 Compatibility

The Manifest element provides information players can use to determine if they can play this file. The Compatibility element refers to the version of specific to which the XML document was written. Players are expected to know which versions they support.

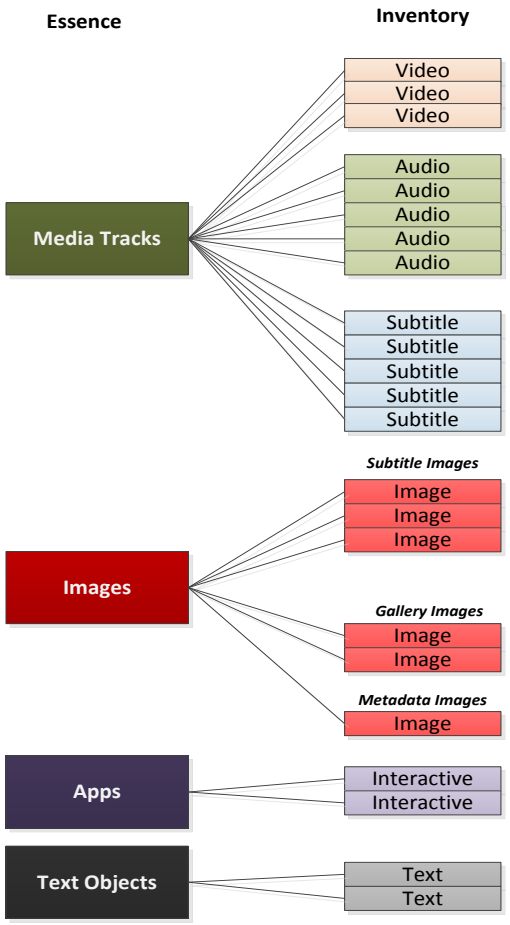
Note that this is distinct from the information in Inventory/Interactive/Encoding that describes the environment for applications.

Element	Attribute	Definition	Value	Card.
Compatibility-type				
SpecVersion		The version of this specification to which the document was written and is conformant.	xs:string	1..n
System		System for which this manifest is intended. If it is compatible with more than one system, there should be more than one instance.	xs:string	0..n

4 INVENTORY

Inventory is a list of audio, video, subtitle and image elements that comprise the Extras experience.

The following illustrates an inventory. The Inventory references essence that are stored in packages or containers; and are stored locally or online.



4.1 Inventory-type

A MediaInventory element is defined to allow the Inventory to be delivered separately.

Element	Attribute	Definition	Value	Card.
MediaInventory		Element containing an Inventory	manifest:Inventory-type	

The type is defined as follows:

Element	Attribute	Definition	Value	Card.
Inventory-type				
Audio		Description of audio asset.	manifest:InventoryAudio-type	0..n
Video		Description of video asset.	manifest:Inventory-type	0..n
Subtitle		Description of subtitle asset.	manifest:InventorySubtitle-type	0..n
Image		Description of image asset.	manifest:InventoryImage-type	0..n
Interactive		Description of Interactive asset	manifest:InventoryInteractive-type	0..n
Ancillary		Description of Ancillary asset	manifest:InventoryAncillary-type	0..n
Metadata		Basic Metadata or reference to Basic Metadata	manifest:InventoryMetadata-type	0..n

4.2 Inventory Asset Types

The inventory consists of audio, video, subtitles and images.

Each entry consists of metadata based on Common Metadata types corresponding with the type (e.g., md:DigitalAssetAudioData-type for audio), along with an unique ID and information where to find the information.

Note that the inventory identifies compete audio, video and subtitle tracks. Presentations (tracks that play together) are built from these.

4.2.1 InventoryAudio-type

Element	Attribute	Definition	Value	Card.
InventoryAudio-type			md:DigitalAssetAudioData-type (by extension)	
	AudioTrackID	Unique identifier for this asset.	manifest:AudioTrackID-type	

ContainerReference		The location of a container that holds the track. Note that containers may be within other containers.	manifest:ContainerReference-type	0..1

4.2.2 InventoryVideo-type

Element	Attribute	Definition	Value	Card.
InventoryVideo-type			md:DigitalAssetVideoData-type (by extension)	
	VideoTrackID	Unique identifier for this asset.	manifest:VideoTrackID-type	
ContainerReference		The location of a container that holds the track. Note that containers may be within other containers.	manifest:ContainerReference-type	0..1

4.2.3 InventorySubtitle-type

Element	Attribute	Definition	Value	Card.
InventorySubtitle-type			md:DigitalAssetSubtitleData-type (by extension)	
	SubtitleTrackID	Unique identifier for this asset.	manifest:SubtitleTrackID-type	
ContainerReference		The location of a container that holds the track. Note that containers may be within other containers.	manifest:ContainerReference-type	0..1

4.2.4 InventoryImage-type

Element	Attribute	Definition	Value	Card.

InventoryImage-type			md:DigitalAssetImageData-type (by extension)	
	ImageID	Unique identifier for this asset.	manifest:ImageID-type	
ContainerReference		The location of a container that holds the track. Note that containers may be within other containers.	manifest:ContainerReference-type	0..1
VideoFrameTimecode		Timecode reference to a particular frame in a Video asset	manifest:Timecode-type	0..1
	VideoTrackID	Video track from which frame is referenced.	Manifest:VideoTrackID-type	

4.2.5 InventoryInteractive-type

Interactive elements may be included. These may define appearance and behavior based on the Media Manifest. Interactive can also be associated applications such as games.

Note that this material is somewhat immature and will develop when use cases are developed.

Element	Attribute	Definition	Value	Card.
InventoryInteractive-type			md:DigitalAssetInteractiveData-type (by extension)	
	InteractiveTrackID	Identifier of this Interactive implementation.	manifest:InteractiveTrackID-type	0..1
ContainerReference		The location of a container that holds the track. Note that containers may be within other containers.	manifest:ContainerReference-type	0..1

4.2.6 InventoryAncillary-type

Element	Attribute	Definition	Value	Card.
InventoryAncillary-type			md:DigitalAssetAncillaryData-type (by extension)	

	AncillaryTrackID	Identifier of this Ancillary Track	manifest:AncillaryTrackID-type	
ContainerReference		The location of a container that holds the track. Note that containers may be within other containers.	manifest:ContainerReference-type	0..1

4.2.7 InventoryMetadata-type

InventoryMetadata-type can contain either Basic Metadata, or reference to Basic Metadata or other metadata. This allows metadata to be specified once in the Manifest and reused across Experiences.

One instance of Basic Metadata may be included in the inventory. Any number of external metadata objects can be referenced through ContainerReference.

InventoryMetadata-type also allows aliases to be created for a Basic Metadata object. These aliases defined localized subsets of Basic Metadata.

At least one instance of ContainerReference or BasicMetadata must be included.

Element	Attribute	Definition	Value	Card.
InventoryMetadata-type				
	ContentID	Content Identifier for the Basic Metadata included or referenced by this object. ContentID must match BasicMetadata/@ContentID.	md:ContentID-type	0..1
ContainerReferece		Reference to Container containing metadata. One instance for each metadata reference. If BasicMetadata is included, this shall not reference Basic Metadata.	manifest:ContainerReference-type	0..n
	type	Type of metadata referenced in ContainerReference	xs:string	
BasicMetadata		Basic Metadata.	md:BasicMetadata-type	0..1
Alias		Definition of an Alias ContentID.	Manifest:InventoryMetadataAlias-type	0..n

The type attribute references the type of metadata referenced in ContainerReference. 'common' shall be used if the referenced metadata is Common Metadata. Other values for type are not defined at this time, although they will likely be defined in the future.

4.2.7.1 InventoryMetadataAlias-type

The ContentID alias mechanism is filter for BasicMetadata that results in a new BasicMetadata instance with select LocalizedInfo instances applicable to a given Experience or Audiovisual. The intended use is to include a single Basic Metadata object that contains all localizations worldwide (i.e., LocalizedInfo for all applicable languages and territories). Alias ContentIDs are created (i.e., Alias/@ContentID) for each localized version. The Alias object would then define the languages and regions associated with that alias. For example, a Basic Metadata object might contain English, French, Spanish, Italian and German. The US version would have a unique Alias/@ContentID. Associated with that Alias/@ContentID would be LanguageIncluded instances for English, French and Spanish (i.e., equal to 'en', 'fr' and 'es'). Whenever that alias ContentID is used, LocalizedInfo instances with @language instances equal to 'en', 'fr' or 'es' would be included and @language instances equal to 'it (i.e., Italian) or 'de' (i.e., German) would be ignored. This complex type defines a subset of a BasicMetadata referred to by InventoryMetadata/@ContentID. This Alias is referred to by an Alias/@ContentID.

An Alias/@ContentID uses ContentID formatting so an alias ContentID is structurally indistinguishable from any other ContentID.

Aliases can be constrained by language, region, both or neither. The following rules are applied when using Aliased metadata:

- If there are no LocalizedPair instances, Alias/@ContentID is functionally equivalent to ContentID.
- If LocalizedPair instances exist,
  - LocalizedPair/Language and LocalizedPair/Region shall respectively correspond exactly with LocalizedInfo/@language and LocalizedInfo/Region for some instance of LocalizedInfo.
  - Instances of LocalizedPair shall appear in the same order as their corresponding LocalizedInfo instances in BasicMetadata.

Element	Attribute	Definition	Value	Card.
InventoryMetadataAlias-type				
	ContentID	Alias ContentID defined by this Alias instance.	md:ContentID-type	

LocalizedPair		Language and Region pairs defining which BasicMetadata/LocalizedInfo instances are used.	manifest:InventoryMetadataLocalizedPair-type	0..n

Within InventoryMetadataLocalizedPair-type, note that at least one instance of LanguageIncluded or RegionIncluded must be included. Otherwise, the pair could not correspond with a LocalizedInfo.

Element	Attribute	Definition	Value	Card.
InventoryMetadataLocaiizedPair-type				
LanguageIncluded		Language corresponding with LocializedInfo/@language to be included in accordance with rules above.	xs:language	0..1
RegionIncluded		Region corresponding with LocializedInfo/Region to be included in accordance with rules above.	md:Region-type	0..n

5 PRESENTATIONS AND PLAYABLE SEQUENCES

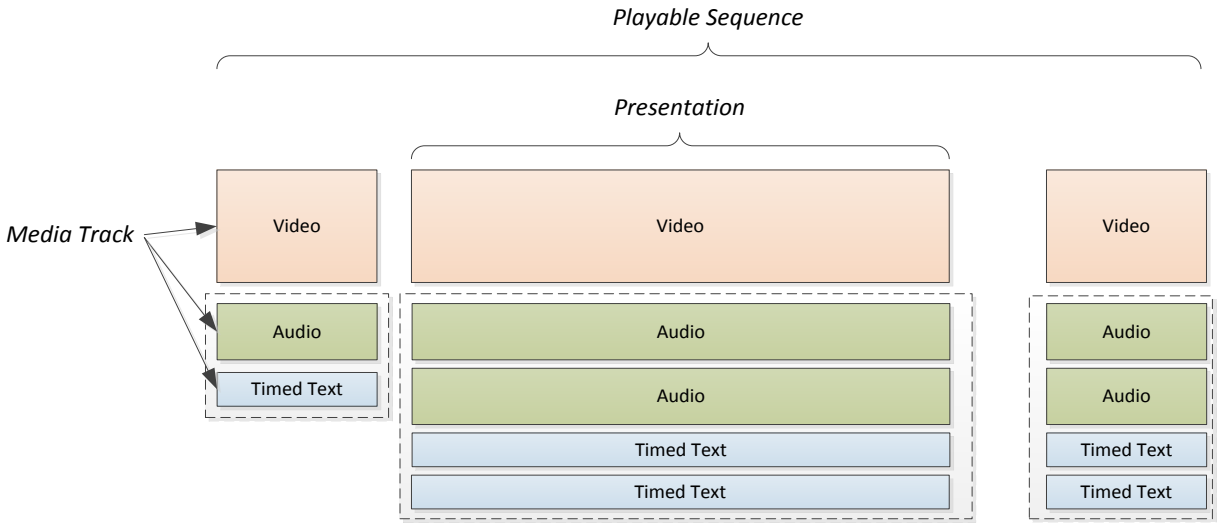
Presentations and Playable Sequences together define what audiovisual material can be played.

Presentations describe audio, video and subtitles that are played together simultaneously. A Presentation is a playable unit and in many cases it plays alone.

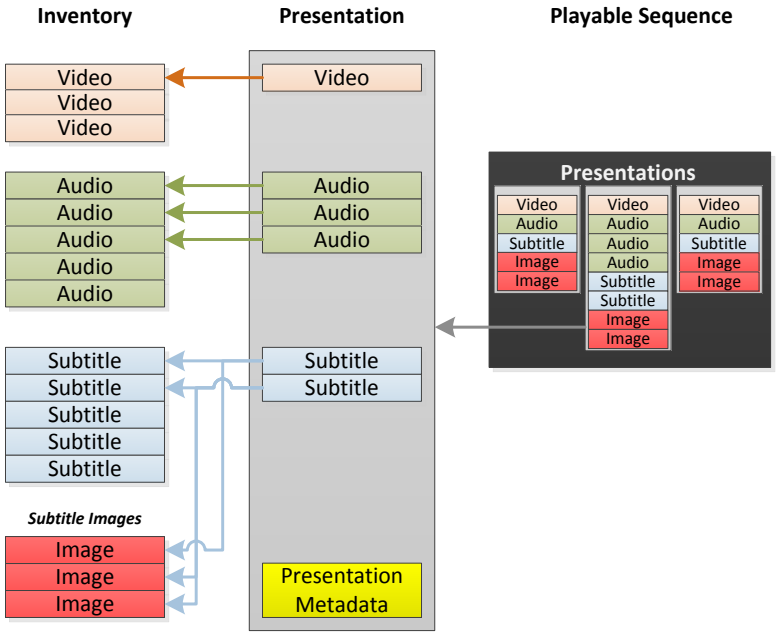
In some instances it is desirable to chain together Presentations. For example, for regional pre-roll video such as anti-piracy cards, one would play the cards before the main title. Playable Sequences indicate which Presentations are played in sequence. In summary:

- Media Track – Track containing media essence.
- Presentation – Tracks that can be played together (timed).
- Playable Sequence – Presentations that are played in sequence.

A Playable Sequence consisting of three Presentations is illustrated here.



Playable Sequences reference Presentations which reference the Inventory. This is illustrated below. Note that the Inventory references the actual media tracks.



5.1 Presentation

The Presentation element provides information which tracks are intended to be played together. They are assumed to be of the same edit. Presentations include tracks that are typically played together as part of a feature, such as video, audio and subtitle tracks. Presentations also contain alternative material, such as commentary audio tracks and ancillary tracks.

The Presentation element also provides information to assist a Device and User selecting tracks in accordance with direction from the content creator. Included are track priority and which audio and subtitle language pair is preferred based on the language preferences. These data are used in conjunction with data in Inventory.

See *Section Annex A. Track Selection Process* for information on expected interpretation of these data for the purpose of default track selection.

5.1.1 PresentationList-type

A Presentation-List contains a set of Presentations.

Element	Attribute	Definition	Value	Card.
PresentationList-type				
Presentation		A Presentation	manifest:Presentation-type	1..n

5.1.2 Presentation-type

A Presentation contains references to all tracks that can play together; that is, all tracks that have been encoded such that a player can play them simultaneously.

Within the Presentation, tracks can be further subdivided into tracks that should be played together. Tracks may appear in more than one TrackMetadata instance. For example, it is likely the video track will appear in TrackMetadata with primary audio and normal subtitles, and also in TrackMetadata for commentary audio/subtitles. See TrackMetadata-type for more detail.

LanguagePair and Chapters apply across all TrackMetadata

Element	Attribute	Definition	Value	Card.
Presentation-type				
	PresentationID	Presentation identifier	manifest:PresentationID-type	
TrackMetadata		Set of tracks that can be played together.	manifest:TrackMetadata-type	1..n
LanguagePair		Defines which audio language and subtitle language are paired with a System Language. Each instance SHALL have a SystemLanguage element. With a unique language.	manifest:ContainerLanguage Pair-type	0..n
Chapters		Chapter stop definitions	manifest:ChapterList-type	0..n

5.1.3 TrackMetadata-type

TrackMetadata-type defines tracks that are associated with each other for the purposes of playback. They must be synchronized, the same length (within some tolerance) and play in meaningful combinations. This allows a player to determine which tracks should be played together. Within an element of this type, any audio track is associated with any video track and any subtitle track; and any subtitle track is associated with any video track and any audio track.

For example, all video, audio and subtitle track relating to the main program, regardless of CODEC and language would be in the same element. However, commentary audio and subtitle tracks would be in a separate element. A TrackMetadata instance would not include both a ‘primary’ audio track and a ‘commentary’ subtitles track that are not intended to be played together. A Device would know from this structure which subtitle track to play with a commentary audio track.

TrackMetadata includes chapter information and LanguagePair (for default track selection). Chapters apply to all tracks in the group.

Also in included in TrackMetadata is additional information to assist with default track selection. Specifically, LanguagePair elements include information about which tracks language combinations the author recommends for a given a System Language.

At least one instance of TrackMetadata SHALL have TrackSelectionNumber='0'.

Element	Attribute	Definition	Value	Card.
TrackMetadata-type				
TrackSelectionNumber		A Track Selection Number assigned to the group of tracks that belong to the same type, such as normal or commentary tracks.	xs:nonNegativeInteger	
VideoTrackReference		Track Reference to a Video track in Inventory.	manifest:VideoTrackReference-type	0..n
AudioTrackReference		Track Reference to an Audio track in Inventory.	manifest:AudioTrackReference-type	0..n
SubtitleTrackReference		Track Reference to a Subtitle track in Inventory.	manifest:SubtitleTrackReference-type	0..n
AncillaryTrackReference		Track Reference to an Ancillary track in the Inventory	manifest:AncillaryTrackRefence	0..n

It is strongly recommended that at most one video track can be included in a Presentation. This is a simplifying assumption. Alternate video must be in a separate Presentation. Exceptions must be carefully evaluated. Normatively, only one video track is allowed unless a specific exception is given in a referencing specification.

Within AudioTrackReference and SubtitleTrackReference, the priority attribute is the relative priority of the track. A smaller number is a higher priority, with `'1'` being the highest priority.

Within a TrackMetadata-type instance, each VideoTrackReference/priority child SHALL be unique.

Within a TrackMetadata-type instance, each AudioTrackReference/priority child SHALL be unique.

Within a TrackMetadata-type instance, each SubtitleTrackReference/priority child SHALL be unique.

Each TrackSelectionNumber represents a selection of tracks that belong to the same type. For example, primary audio tracks and normal subtitle tracks are associated with TrackSelectionNumber ='0', director's commentary audio tracks and subtitle tracks are associated with TrackSelectionNumber ='1', and so on.

Audio tracks of type ‘primary’ and subtitle tracks of Type ‘normal’ SHALL be associated with TrackSelectionNumber=’0’.

AudioTrackReference and SubtitleTrackReference elements, lists the track priority order for all video, audio and subtitle tracks associated with the TrackSelectionNumber. All tracks within TrackMetadata of a particular type (i.e., audio, video, subtitle) associated with a lower TrackSelectionNumber are higher priority than all tracks of that same type associated with a higher TrackSelectionNumber.

The priority attribute can be used to specify priority order amongst equivalent tracks. For example, given multiple AudioTrackReference instances that reference primary English tracks with different CODECs, the preferred order of these tracks would be indicated by the priority attributes, with the most preferred track having priority=’1’. If there are multiple instances of SubtitleTrackReference elements for equivalent tracks with different Track/FormatTypes (Text or Image), authors can specify which FormatType has higher priority using the priority attribute. Within a TrackMetadata instance, Priority is unique across all audio tracks and is unique across all subtitle tracks.

Note that CFF currently only allows one video track, so it is not meaningful to have more than one VideoTrackReference (i.e., a cardinality of 1). The schema allows multiple instances to support future growth.

5.1.3.1 AudioTrackReference-type, VideoTrackReference-type and SubtitleTrackReference-type

AudioTrackReference-type, VideoTrackReference-type and SubtitleTrackReference-type allows audio, video and subtitle tracks, respectively, to be both referenced and grouped into Adaptation Sets for adaptive streaming.

There are two forms of these elements. In the first form, a single track is referenced. In this form, there is exactly one track identifier element.

The other form is for adaptive streaming, and the TrackReference refers to an adaptation set. In this case, a TrackID instance is included for each Representation in the adaptation set. Note that one must examine the metadata in the Inventory to determine which Representation has the desired characteristics (e.g., bitrate). See Section 5.1.6 for more information on Adaptive Streaming.

In both forms, the priority may be included for use in the default track selection process.

Element	Attribute	Definition	Value	Card.
AudioTrackReference-type				

	priority	Relative priority of this track. Used in default track selection.	xs:positiveInteger	0..1
AudioTrackID		The AudioTrackID references audio tracks in the Inventory.	manifest:AudioTrackID-type	1..n
AdaptationSetID		If included, identifies the Adaptation Set to which the Tracks belong. Reserved for future use.	md:id-type	0..1

Element	**Attribute**	**Definition**	**Value**	**Card.**
VideoTrackReference-type				
	priority	Relative priority of this track. Used in default track selection.	xs:positiveInteger	0..1
VideoTrackID		The VideoTrackID references video tracks in the Inventory.	manifest:VideoTrackID-type	1..n
AdaptationSetID		If included, identifies the Adaptation Set to which the Tracks belong. Reserved for future use.	md:id-type	0..1

Element	**Attribute**	**Definition**	**Value**	**Card.**
SubtitleTrackReference-type				
	priority	Relative priority of this track. Used in default track selection.	xs:positiveInteger	0..1
SubtitleTrackID		The SubtitleTrackID references subtitle tracks in the Inventory.	manifest:SubtitleTrackID-type	1..n
AdaptationSetID		If included, identifies the Adaptation Set to which the Tracks belong. Reserved for future use.	md:id-type	0..1

Element	**Attribute**	**Definition**	**Value**	**Card.**

AncillaryTrackReference-type				
	priority	Reserved for use specific to the type of Ancillary track used	xs:positiveInteger	0..1
AncillaryTrackID		The AncillaryTrackID references ancillary tracks in the Inventory.	manifest:AncillaryTrackID-type	1..n
AdaptationSetID		If included, identifies the Adaptation Set to which the Tracks belong. Reserved for future use.	md:id-type	0..1

5.1.4 ContainerLanguagePair-type

ContainerLanguagePair-type allows the author to specify audio and subtitle track pairs based on a User's System Language.

A User preference for System Language does not always imply audio and subtitle tracks of the same language. For example, in some cases the best choice for a Japanese viewer would be Japanese language audio and no subtitle. In other cases, the best choice would be an English audio track and a Japanese subtitle.

Presentation/AudioReference and Presentation/SubtitleReference refer to a subset of tracks in Inventory/Audio and Inventory/Subtitle respectively. ContainerLanguagePair-type further constrains the track list by selecting tracks by language. That is, LanguagePair refers only to audio tracks where Inventory/Audio/Language equals AudioLanguage and to subtitle tracks where Inventory/Subtitle/Language equals SubtitleLanguage.

Element	**Attribute**	**Definition**	**Value**	**Card.**
ContainerLanguagePair-type				
SystemLanguage		The language scope for which the Language Pair applies. For example, if this element is 'en-US' then the Language Pair element applies to English spoken in the United States.	xs:language	
AudioLanguage		Author recommended audio language for given SystemLanguage	xs:language	

SubtitleLanguage		Authore recommended subtitle language for given SystemLanguage	xs:language	

Within the set of LanguagePair elements, each LanguagePair element SHALL have a unique value in SystemLanguage.

5.1.5 Chapter Metadata

An A/V stream may be divided into chapters. The assumption is that a use may skip between chapters or jump to a chapter based on a list.

In an A/V stream, chapters are referenced to video and are timed or referenced to a specific video frame. Audio chapters are time referenced to the beginning of the audio.

It is best practice to encode audio and video to allow jumps to chapter starts.

Chapter start times are assumed against a/v stream of the same edit. A video with parts added or removed, such as a director's cut will have different chapter start times than the theatrical cut. This generally applies to Supplemental Material.

The challenge in defining chapters is referencing the correct frame. Depending on how the video is encoded, the time reference can be different.

Chapter metadata identifies the locations within a track where chapters begin. Each chapter has a numerical index and an entry point that defines where the chapter starts.

Note that Chapters are defined against a Presentation rather than PlayableSequence. Although this is less general, it greatly simplifies the Chapter implementation.

Element	Attribute	Definition	Value	Card.
ChapterList-type				
Chapter		Chapter entry point descriptor	manifest:Chapter-type	1..n

Chapter elements in ChapterList-type SHALL be in chapter order.

Element	Attribute	Definition	Value	Card.
Chapter-type				
	index	Chapter index.	xs:integer	
EntryTimecode		Entry point for chapter start.	manfiest:Timecode-type	
DisplayLabel		Displayable text on a per-language basis for the chapter	xs:string	0..n
	language	Language of DisplayLabel. Must be included in all DisplayLabel elements if more than one DisplayLabel element is included. Matching is in accordance with Section 4.1.5.1 Use of Language	xs:language	0..1
ImageID		Reference to a chapter image.	manifest:ImageID-type	0..n
	language	Language of image referenced by ImageID. Must be included in all ImageID elements if more than one ImageID element is included and there is language-specific text in the image (i.e., burned in text).	xs:language	0..1

The index attribute is a number starting with 0 and increasing monotonically for each subsequent chapter.

EntryTimecode corresponds with a constrained form of the 'offset-time' syntax (without the metric field) of the media timebase defined in [TTML], Section 10.3.1, and corresponds with the beginning of the chapter in the video and/or audio tracks for which the chapters are identified. The metric is in units of seconds.

In the case of a rounding error that doesn't result in an integer number of frames, the video and/or audio frame(s) EntryTimecode refers to shall be the next decodable frame after the time in the media referenced by this value. For example, in a 30fps progressive video track, 0.1 = the 3rd frame. 0.101 = the 4th frame.

5.1.6 Presentations and Adaptive Streaming

For Adaptive Streaming, TrackReference allows multiple TrackIDs. Each TrackID is reflects a Representation (defined here). The specific information on the Representation is found in the Inventory.

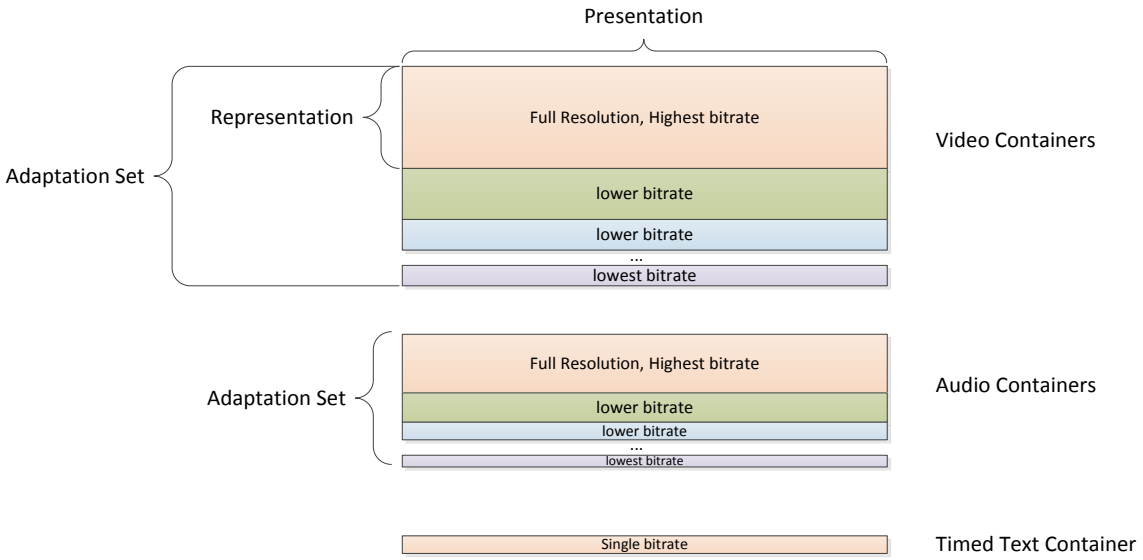
The Presentation structure supports Adaptive Streaming as follows. This was developed with DASH [DASH] as the primary model, although other models should work.

In Adaptive Streaming, some combination of player and server logic switches between tracks with different bitrates to not exceed available bandwidth. Adaptive Streaming requires a collection of tracks that are essentially the same except for encoding parameters, with the most essential parameter being bitrate.

The following terminology is used

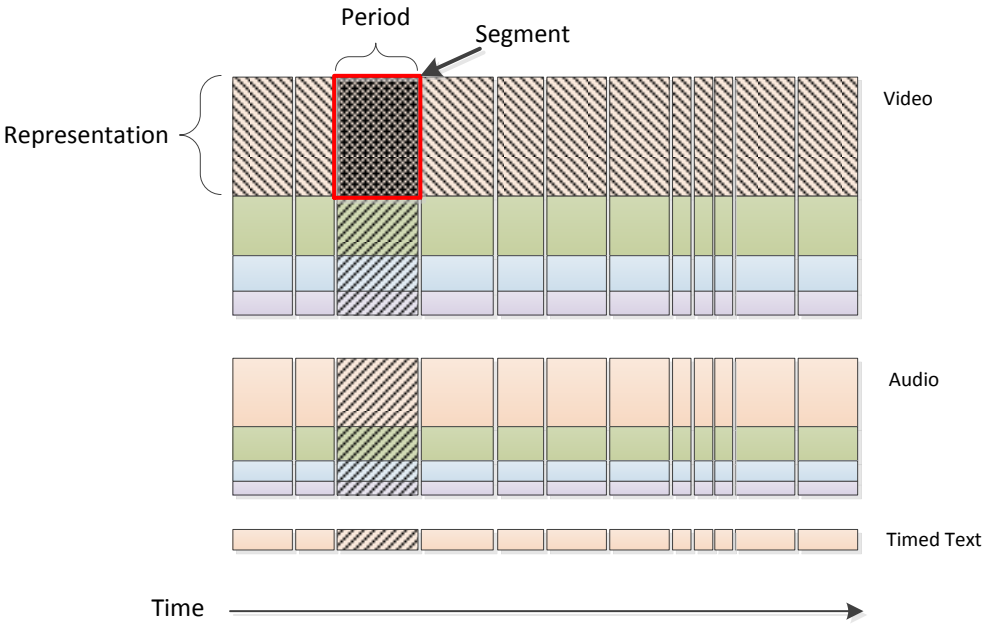
- Representation is a single encoding of a track. There is a bitrate associated with this track.
- An Adaptation Set is a collection of Representations. Each Representation encodes the same content, but with different encoding parameters, particularly bitrate.

Following is an illustration of a Presentation with one video track, one audio track and one subtitle track. The video and audio tracks have multiple Representations



The model supports adaptively streaming video, audio and subtitles. Typically only video has multiple Representations, less frequently audio will have multiple Representations and only rarely will subtitles/timed text have multiple Representations.

Also implicit in DASH is the concept of time Periods. Representations are divided into segments corresponding to the same Periods. It is important that the tracks be constructed such that the segments align. Segment definition is outside the scope of this document. The tracks above are illustrated with segments. The red box pointed to by “Segment” is a single video segment (one time Period for one Representation).



5.1.7 Ancillary Track Model

Ancillary Tracks are tracks that are not in themselves playable, but provide additional information for other tracks. All Ancillary tracks are bound to a specific track audio, video, subtitle track in the same Presentation. Although no use case is currently identified for the use of Ancillary Tracks for images and interactive, its use is not prohibited.

Players that do not know how to interpret an Ancillary Track are expected to ignore that Ancillary Track.

Ancillary Track Metadata is defined in [CM], particularly Section 5.2.12.

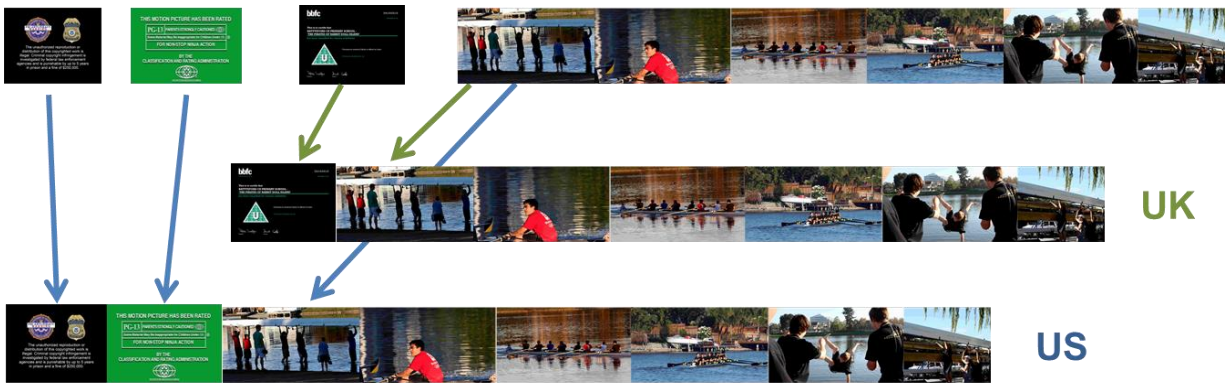
5.2 Playable Sequences

The Presentation is the minimum unit of playback from the perspective of a User. A Playable Sequence references one or more Presentations in sequence.

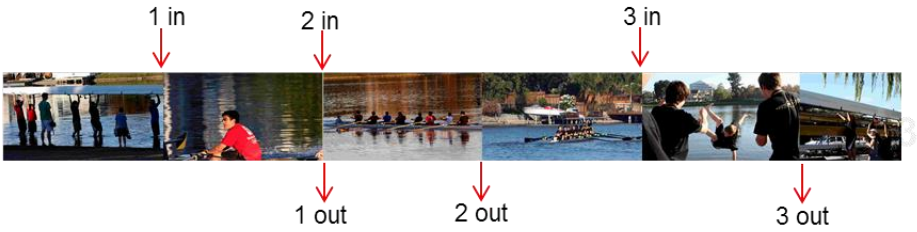
5.2.1 Playable Sequences Use Cases

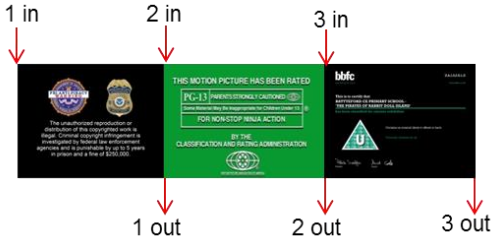
Playable Sequences supports pre-roll or post-roll material unique to a region. In this case, the main title is a Presentation and each pre- and post-roll segment is its own Presentation. For each region, there is a Playable Sequence consistent of the pre-roll Presentation for that region and the main title Presentation.

The following picture illustrates how four distinct Presentations are chained into Playable Sequences suitable for UK and US playback.



Playable Sequence allows the specification of entry and exit timecodes within a Presentation. Usage examples include extracting specific clips from a main title or to extract specific sections for pre-roll or post-roll. These are illustrated below. 'In' and 'Out' are encoded as EntryTimecode and ExitTimecode respectively.





This use case allows some period of black and silence between clips. That is, clips need not be played with the first frame of one clip immediately following the last frame of the previous clip.

Another use case supports support edits, such as a Theatrical Cut and a Director's Cut. In this case, clips would be played such that clips are played with no gaps (i.e., video plays frame-to-frame and audio plays continuously). This is referred to as seamless chaining. If the seamless attribute is 'true' for given clip it is intended to be played immediately following the previous clip.

5.2.2 Chapters, Timelines and Playable Sequences

When Presentations are sequenced using Playable Sequences, they must construct a complete chapter list correctly. Chapter times in each Presentation must remain at the same point in that Presentation. The full chapter list is constructed from the chapter list in each Presentation in sequence.

A Playable Sequence is a concatenation of Presentations that each has their own timeline. There is no defined concept of a timeline for the end result. It is at the discretion of the player whether it wishes to provide an end-to-end timeline. For example, if a player displayed a progress timeline, it would likely add the total time of all clips in the playable sequence and add any inter-clip delays to create a total time. As playback proceeded through the clips, the current time would be the accumulated time through all clips to that point. One variation is to only provide timelines for individual clips or, if applicable, a collection of clips played with seamless chaining.

5.2.3 PlayableSequenceList-type

A Playable Sequence List contains multiple PlayableSequence elements.

Element	Attribute	Definition	Value	Card.
PlayableSequenceList-type				
PlayableSequence		A Playable Sequence. The order of PlayableSequence elements defines the order of playback.	manifest:PlayableSequence-type	1..n

5.2.3.1 PlayableSequence-type

The Playable Sequence allows Presentations and images to be chained together into a final viewable video.

This can be used both in the modes where the last frame of one clip continues without hesitation into the first frame of the next clip, and where there may be a period of blank screen and silence between clips. That latter is assumed across all implementations. The former might not be supported in some environments. Individual systems must define their own specific constraints; however, changes in resolution, frame rate, encryption keys and other decoding parameters might occur between clips.

The order of playback is determined by the sequence attributes in Clip and ImageClip. Clips and Image Clips can be interspersed. The sequence attribute indicates the order of playback. Note that sequence is a signed integer, so playback could start with clips or images with a negative value sequence.

Instances of Clips SHALL be listed in ascending order of sequence. Instances of ImageClips SHALL be listed in ascending order of sequence.

The sequence element SHALL be unique across all instances. sequences in a PlayableSequence SHALL have contiguous values. Within a Playable Sequence, there SHALL be at least one instance of with sequence attribute is either missing or zero ('0'). This is considered the primary clip. Track selection is performed on the primary clip.

The seamless attribute is meaningless for the first clip and SHOULD NOT be specified. If specified, it shall be ignored.

Element	Attribute	Definition	Value	Card.
PlayableSequence-type				
	PlayableSequence ID	Identifier used to refer to the PlayableSequence. Must be unique within an Extras definition.	manifest:PlayableSequenceID-type	
Clip		Clip of audiovisual playback material.	manifest:AudiovisualClipRef-type	1..n
ImageClip		An image to be played for a period of time (duration).	manifest:ImageClipRef-type	0..n
ReferenceID		Identifiers that identify the PlayableSequence. Additional information can be obtained using these Identifiers. We strongly suggest that at least one instance be an EIDR ID.	md:ContentIdentifer-type	0..n

5.2.3.2 AudiovisualClipRef-type

A Clip is a subset of audio, video, subtitles or some combination.

The following defines an audiovisual clip.

Element	Attribute	Definition	Value	Card.
AudiovisualClipRef-type				
	sequence	Indicates sequence of this clip relative to other clips in the same grouping. Order is from lowest value to highest.	xs:integer	0..1

	seamless	If 'true', this clip is to be played immediately following previous clip (i.e., frame to frame).	xs:boolean	0..1
PresentationID		Identifier for Presentation that may contain combinations of audio, video and subtitles.	manifest:PresentationID-type	
EntryTimecode		Beginning timecode for clip. If absent, start is the beginning of the track.	manifest:Timecode-type	0..1
ExitTimecode		Timecode for end of clip. If absent, end is the end of the track.	manifest:Timecode-type	0..1

5.2.3.3 AudioClipRef-type

A Clip is a subset of audio. This is useful for features such as audio loops.

The following defines an audio-only clip.

Element	**Attribute**	**Definition**	**Value**	**Card.**
AudioClipRef-type				
	sequence	Indicates sequence of this clip relative to other clips in the same grouping. Order is from lowest value to highest.	xs:integer	0..1
	seamless	If 'true', this clip is to be played immediately following previous clip.	xs:boolean	0..1
AudioTrackID		Identifier for audio track.	manifest:AudioTrackID-type	
EntryTimecode		Beginning timecode for clip. If absent, start is the beginning of the track.	manifest:Timecode-type	0..1
ExitTimecode		Timecode for end of clip. If absent, end is the end of the track.	manifest:Timecode-type	0..1

5.2.3.4 ImageClipRef-type

An Image Clip is an image reference and duration, such that the duration indicates how long the referenced image should be played. It is to be played equivalently to a Presentation of length duration that shows a single image for the duration.

The following defines an Image Clip.

Element	Attribute	Definition	Value	Card.
ImageClipRef-type				
	sequence	Indicates sequence of this clip relative to other clips in the same grouping. Order is from lowest value to highest.	xs:integer	0..1
	seamless	If 'true', this clip is to be played with immediately following previous clip (i.e., frame to frame).	xs:boolean	0..1
ImageID		Identifier for Image in the Inventory.	manifest:ImageID-type	
Duration		How long the image is to be displayed.	xs:duration	0..1

5.2.4 Playable Sequence constraints to support default track selection

Annex A. defines Default Track Selection. This algorithm assumes that the entire video has the same tracks. This is not necessarily the case in a Playable Sequence where the individual Presentations do not have the same tracks. The section constrains Playable Sequences to support track selection.

There are two primary use cases supported by these constraints. The first is a work that is divided into sections. A common example is a movie that can show with or without deleted scenes. In this case, all Presentations must have the same combination of tracks.

The second use case involves material that appears prior to the main title or following the main title. Content commonly in this category includes anti-piracy notices and distributor logos. These do not typically have alternate tracks.

One Presentation with the full complement of tracks, typically the main feature, is used for track selection. Other Presentations must either match that Presentation's audio track or have a single audio track; and must match that Presentation's subtitle track or have no subtitles

Track Selection is based on the 'primary clip' as defined in Section 5.2.3.1. Track selection for clips other than the primary clip is as follows

- If the clip has the same tracks as the primary clip, the same tracks are chosen as were chosen for the primary clip.
- Otherwise

- That clip must have zero or one audio tracks. If a track is present, it is always selected. Note that this track would usually be a music and effect track.
- That clip must have zero or one subtitle tracks. If a track is present, it is always selected. The most useful case is an language-neutral image subtitle track.

6 PICTURE GROUPS AND GALLERIES

Images may be provided with a main title, or as supplements to supplemental audiovisual material.

Images are grouped and sequenced. Basic models allow for a single sequence (slide show). More advanced models allow more complex navigation paths.

The Gallery, part of a Experience, defines how a Picture Group is displayed.

6.1 Picture Group

The top level definition for Picture Groups is PictureGroupList-type. It contains one or more Picture Groups.

Within PictureGroupList-type, PictureGroup elements may appear in any order.

Element	Attribute	Definition	Value	Card.
PictureGroupList-type				
PictureGroup		A list of Picture Groups.	manifest:PictureGroup-type	1..n

6.2 Picture Group Type

A Picture Group consists of one or more references to pictures. These pictures may be annotated.

Image annotation includes

- Localized captions to be displayed with images
- Languages of text in images (for localization)
- Intended sequence for playback in a gallery.
- Localized alternate text for accessibility.

Element	Attribute	Definition	Value	Card.
PictureGroup-type				
	PictureGroupID	Identifier for the Picture Group. Must be unique within an Extras element.	manifest:PictureGroupID-type	
Picture		An individual picture within the PictureGroup.	manifest:Picture-type	1..n

6.2.1 Picture-Type

Picture-Type describes an individual picture, including how it relates to other pictures when sequenced within a Gallery.

Note that an image (ImageID) may be referenced by more than one Picture. The picture might have different annotation and be used in a different context. Hence, a unique PictureID is required.

Element	Attribute	Definition	Value	Card.
Picture-type				
PictureID		Identifier for this Picture.	manifest:PictureID-type	
ImageID		Reference to the image for the Picture.	manifest:ImageID-type	
LanguageInImage		If there is any text visible in the image, this element identifies this language. Anticipated use is to determine when alternate text is required.	xs:language	0..1
AlternateText		Alternate text to be used for accessibility.	xs:string	0..n
	Language	Language of AlternateText	xs:language	0..1
Caption		Caption for the image.	xs:string	0..n
	Language	Language of Caption	xs:language	0..1
Sequence		Order of this Picture in Picture Group.	xs:nonNegativeInteger	0..1

It is assumed that accessible applications will perform text-to-speech on Caption if no AlternateText is provided. If provided, AlternateText is intended to replace Caption with any caption information plus a description of the contents of the image.

If there is only one Picture in the PictureGroup, Sequence should be omitted. If there is more than one Picture in the Picture Group then either

- Pictures are unordered: no Picture may have a Sequence
- Pictures are ordered: all Pictures must have a unique Sequence. The first Picture must have a Sequence of 0.

7 INTERACTIVE APPLICATIONS

This section defines data describing interactive applications. The term "app" and "application" are used interchangeably, both in the schema and in prose.

An Application Group (App Group) is a collection of applications with different implementations that differ primarily in the environment in which they run. For example, a Navigation Application for a TV series might have native implementations for Android, Windows, Mac and iOS; and HTML5 and declarative XML definitions. There might be distinct implementations for specific system versions. These would all have similar behavior, allowing a user to navigate the series' seasons, episodes and extras material. All these implementation would be in the same Application Group. The Experience references the Application Group to specify the functionality and the player or the player environment chooses the most appropriate application for the circumstances.

The information for selecting the appropriate application is included deep in the Inventory. However, to facilitate easier access and to allow a TrackGroup to stand alone, this information is replicated for each application.

7.1.1 AppsGroupList-type

Element	Attribute	Definition	Value	Card.
AppsGroupList-type				
AppGroups		A list of App Groups.	manifest:AppsGroup-type	0..n

7.1.2 AppsGroup-type

AppsGroupType defines which interactive applications perform the same function, but are implemented for different platforms. For example, a declarative XML extras experience, an Adobe Flash application and an HTML5-based application could each be listed.

Element	Attribute	Definition	Value	Card.
AppGroup-type				
	AppGroupID	Unique identifier for this App Group.	manifest:AppGroupID-type	
InteractiveTrackReference		Reference to an Interactive track (i.e., application)	manifest:InteractiveTrackReference-type	0..n

Within InteractiveTrackReference instances the priority attribute is the relative priority of the application implementation. A smaller number is a higher priority, with '1' being the highest priority.

Within an AppGroup-type instance, each InteractiveTrackReference/priority child SHALL be unique.

InteractiveTrackReference elements, lists the track priority order for all applications within the group.

The priority attribute can be used to specify priority order amongst equivalent apps. For example, given XML, Adobe Flash and HTML5, one could specify that HTML5 is the preferred application by giving it the highest priority (i.e., priority='1').

7.1.2.1 InteractiveTrackReference-type

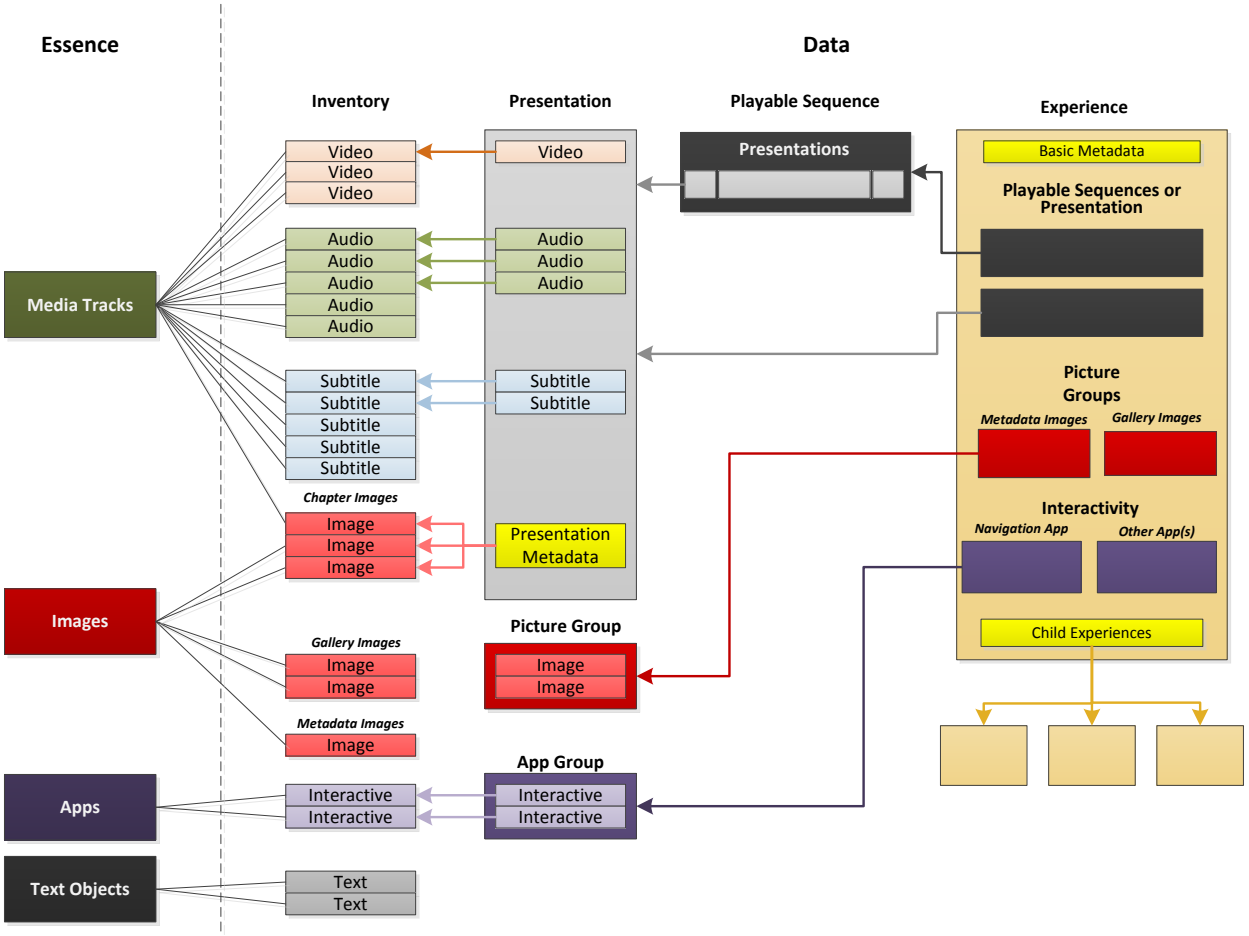
InteractiveTrackReference-type allows applications that perform the same function to be grouped. An instance of InteractiveTrackReference-type exists for each application that performs the same function. Selection of the appropriate track requires following the InteractiveTrackID to the metadata in Inventory.

Element	Attribute	Definition	Value	Card.
InteractiveTrackReference-type				
	priority	Priority of this implementation relative to other implementations in the TrackGroup.	xs:positiveInteger	0..1
InteractiveTrackID		The InteractiveTrackID references an interactive track in the Inventory.	manifest:InteractiveTrackID-type	
Compatibility		Application encoding information, used to select appropriate Application implementation. As there maybe multiple target platforms for a given implementation, multiple instances can be included.	md:DigitalAssetInteractiveEncoding-type	1..n

8 EXPERIENCE

The Experience is the top-level element for determining what can be offered to a user. It defines which other elements comprise a user experience.

The following illustrates how Experiences related to Programs, Playable Sequences, Presentations, Picture Groups, App Groups, and Text (text not yet implemented).



An Experience consists of pointers to the following objects

- Presentations and Playable Sequences – audiovisual material
- Picture Groups – metadata images (cover art) and galleries
- Applications – applications used to navigate the experience and other applications such as games

There is metadata at the Experience level and also metadata at the each object referenced by the Experience.

The Experience contains applicability information about language and region so targeted Experiences can be created.

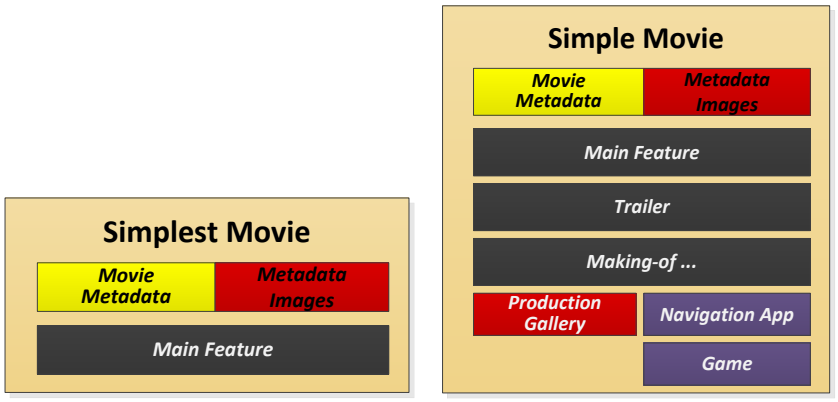
Experiences need to work with all content relationships such as episode/season/series, movie series and franchises. Consequently, Experiences can references subordinate (child) Experiences recursively. This is explained in the following section.

Note that a DECE Media Package (DMP) [DMP] can store an entire Experience.

8.1 Experience Structure

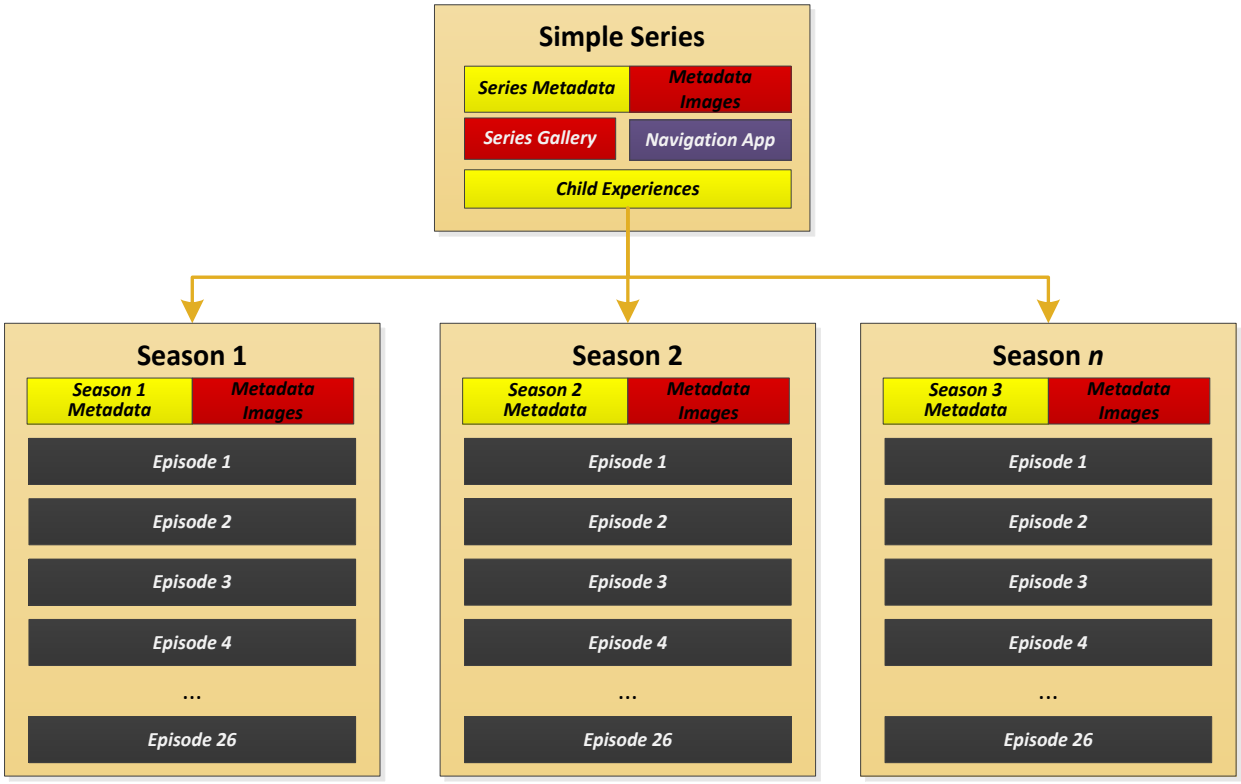
A viewing experience can be more than a single title. The Experience model supports a broad range of content relationships. The simplest cases are single titles and episodic title.

The following example represents the simplest movie Experience ("Simplest Movie") and a more complex Experience ("Simple Movie").



The Simple Movie has metadata and the audio, video and subtitles for the main feature. Also included, but not shown, is metadata for the main feature. As the Experience references Sequences and Presentations, all features of Presentations and Inventory including chapters, track selection information and detailed metadata is available. "Simple Movie" includes an application, actually an AppGroup, to navigate the various titles and a game. It also includes a Gallery of production images.

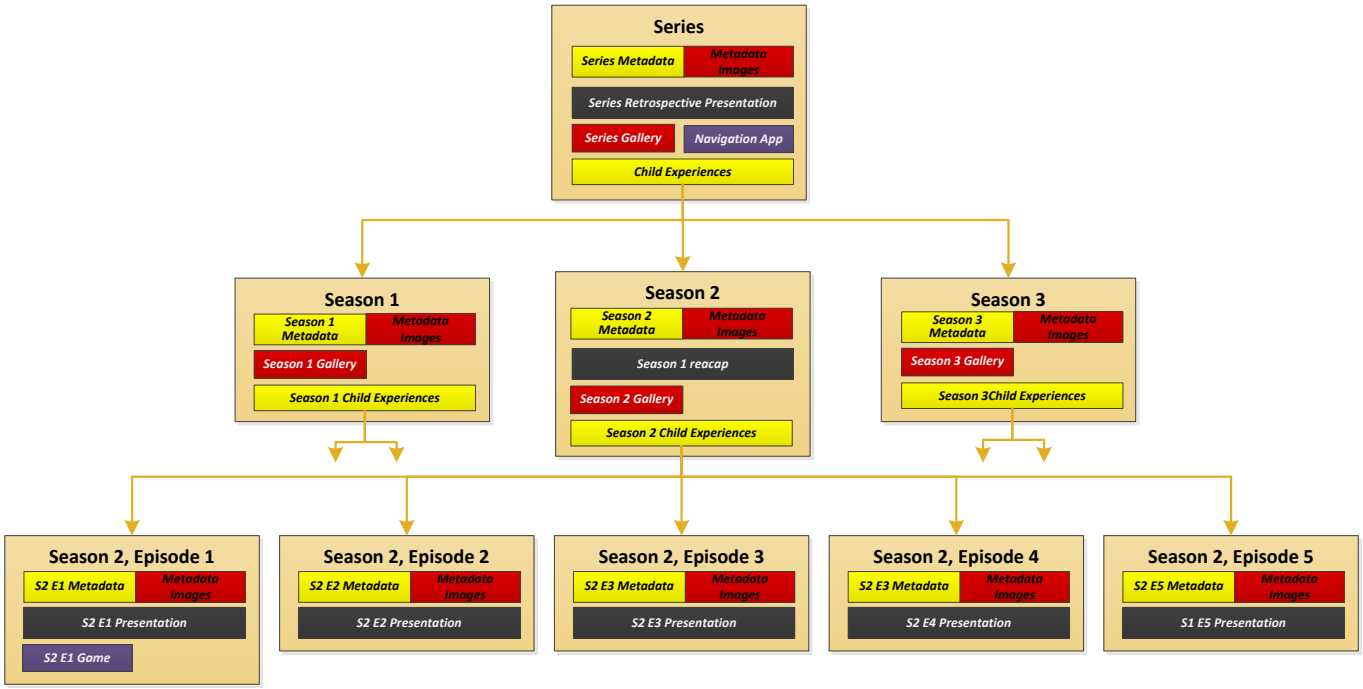
The following example shows episodic material. At the top level is an Experience element describing the TV series.



In this example, there is a gallery and navigation application. To accommodate grouping Experience is recursive with ExperienceChild pointing to Experience elements for subordinate Experiences. In this case, the subordinate Experiences are season.

As with the "Simple Movie" example above, each Season contains multiple audiovisual entries; in this case, one for each episode. In this example, the navigation app at the series level handles navigation of all episodes.

Following is a more complex series example. It is unlikely material will be this complex, but the example is useful to illustrate flexibility.



As above there is a Series-level object that includes series-level metadata, a navigation application and a series gallery. In this case there also a Presentation containing a series retrospective. As the retrospective applies across all seasons it is maintained at the series level.

The series consists of three seasons each with metadata, metadata images and a gallery unique to that season. Season 2 also includes a Season 1 recap Presentation.

In this example, the seasons reference individual episodes (only Season 2 shown). This allows additional material to be attached to the episode. In this case, Season 2, Episode 1 includes a game unique to that episode. As shown, this example is equivalent to the Simple Series above, with the exception that game. This scenario is most useful when there is material designed for individual episodes.

8.2 Experience List

The ExperienceList element contains one or more instances of Experience. Each instance may be international, or it might be targeted towards some combination of region and language. Playback behavior is to select the best fit for the applicable region.

Element	Attribute	Definition	Value	Card.
ExperienceList-type				
Experience		Each instance contains a collection of Title and Gallery elements, possibly specific to a combination of language, region and/or version.	manifest:Experience-type	1..n

Experience elements may have language attributes and Region elements that indicates which instance should be used. The first filter is language. That is, a player should first select Experience elements with a suitably matching language. An Experience element without a matching language is assumed to match all languages not present in other Experience elements. Once language is matched, the Region should be matched.

The following is the order of preference for matches (highest first)

- Matching language and Region
- Matching Region element and no language attribute present
- Matching language attribute and no Region element present
- Any other match of language
- Any other match of Region
- No match (no rules on selection)

A match cannot occur if either ExcludedLanguage or ExcludedRegion is matched.

For the purposes of matching

- The absence of a matching ExcludedLangauge element is considered equivalent to no language match.
- The absence of a matching ExcludedRegion is considered equivalent to no Region present.

It is up to the implementation whether it processes the version attribute.

8.3 Experience-type

The Experience element defines which titles and galleries apply.

Element	Attribute	Definition	Value	Card.
Experience-type				
	ExperienceID	Unique Identifier for this Experience	manifest:ExperienceID-type	0..1
	version	Experience version, used when player accepts multiple versions. For this specification, version='1'.	xs:double	
	updateNum	Version of this document. Initial release should be 1. This is a value assigned by the manifest creator that should only be incremented if a new version of manifest is released. If absent, 1 is to be assumed. This is generally only used for selective update workflows.	xs:integer	0..1
Language		Language for which Experience was authored	xs:language	1..n (optional choice)
ExcludedLanguage		Excluded language for which Experience was authored.		
Region		Region for which Experience was authored	md:Region-type	1..n (optional choice)
ExcludedRegion		Excluded region from which Experience was authored.		
ContentID		ContentID for Basic Metadata describing the Experience.	md:ContentID-type	0..1 (optional choice)
BasicMetadata		Basic Metadata describing the Experience.	md:BasicMetadata-type	
Audiovisual		Audiovisual titles. The first title must be the main title and have Type='main'.	manifest:Audiovisual-type	0..n
App		Identifies applications associated with this Experience.	manifest:ExperienceApp-type	0..1

Gallery		An image galleries associated with this Experience. Gallery's purpose is to define playback assets. PictureGroupID is used for inventory.	manifest:Gallery-type	0..n
PictureGroupID		References to PictureGroups. This includes PictureGroups in Galleries as well as PictureGroups in metadata.	md:PictureGroupID-type	1..n
ExperienceChild		Child Experience elements. Usage is defined below.	manifest:ExperienceChild-type	0..1

ContentID can refer either to the ContentID associated with metadata, or if Inventory/Manifest/Alias is used, ContentID may refer to an alias ContentID (i.e., Alias/@ContentID). In this case, the aliased ContentID will be substituted as part of the dealiasing process.

Organization uses the Common Metadata Completion Object (md:CompObj-type) to fully organize titles within the Experience. This structure can include hierarchies such as a show comprised of seasons further comprised of episodes. In this usage, CompObjEntry-type is constrained as follows:

- DisplayName shall not be included
- EntryNumber should be included
- EntryClass should be included
- Entry is included if appropriate
- ContentID is the only valid choice and it shall be a ProgramID.

PictureGroupID must be used in applications where it might not be immediately clear how images are organized. For example, when Experience is used as part of a delivery manifest PictureGroupID informs the ingestion process where to find images such as metadata images and gallery images. This will include references to all Picture Groups referenced in Gallery elements. The reason for this apparent redundancy is that Gallery is to be used for the organization of playback, not the organization of ingestion. In circumstances where Experience is part of a complete package (e.g., a DECE DMP) it is not necessary to include PictureGroupID elements.

8.3.1 Audiovisual

A collection of playable video, audio, subtitles and other behavior defined by an Audiovisual element. This can be the main feature, promotions such as trailers, or value added material such as making-of videos or deleted scenes.

Element	Attribute	Definition	Value	Card.
Audiovisual-type				
	ContentID	Content Identifier associated with the Basic Asset Metadata describing this Program.	md:ContentID-type	0..1
Type		Type of this title (see below)	xs:string	
SubType		Additional detail on type	xs:string	0..n
PresentationID		Identifier for the Presentation associated with this title. Used when there is no need for a Playable Sequence.	manifest:PresentationID-type	0..1 (choice)
PlayableSequenceID		Identifier for a Playable Sequence. Used when PlayableSequence is referenced.	manifest:PlayableSequenceID-type	
PlayableSequence		Playable Sequence definition. Used when a separate PlayableSequence element cannot be referenced.	manifest:PlayableSequence-type	
ContentID		ContentID for Basic Metadata describing the Experience.	md:ContentID-type	0..1 (optional choice)
BasicMetadata		Basic Metadata associated with the Program.	md:BasicMetadata-type	

At least one of PresentationID, PlayableSequenceID or PlayableSequence must be included if the intent is to play content. If the intent is information, such as pre-order, then these may be excluded, but ContentID or BasicMetadata must be included.

ContentID can refer either to the ContentID associated with metadata, or if Inventory/Manifest/Alias is used, ContentID may refer to an alias ContentID (i.e., Alias/@ContentID). In this case, the aliased ContentID will be substituted as part of the dealiasing process

Type describes the top-level nature of the Program. Note that useful information in the Program can also be found in BasicMetadata or ReferenceID information in the referenced Presentation. Type is enumerated as follows:

- 'Main' – Main title (typically the feature)
- 'Promotion' – Trailers, teasers, etc.

- 'Bonus' – Additional material related toward the Main Program, such as, deleted scenes, making-of, etc.
- 'Other' – Any other material included

SubType can add additional detail, especially to 'Bonus' Type. SubType may include terms such as 'Deleted Scenes' or 'Making-of'.

8.3.2 ExperienceApp

ExperienceApp defines an application associated with an Experience.

The ExperienceApp is independent of target platform—it references AppGroup elements which encapsulate that information.

An application used to navigate the experience is called a Navigation Application. If a Navigation Application is present, it must be first and include Type='nav'. If child experiences are referenced by the Experience element, the Navigation Application is assumed to either navigate the child Experiences or be capable of handing off to control to Navigation Applications in the children.

The name of the application (AppName) and content rating (Rating) are included as metadata. It is likely that additional metadata will be added in the future.

Element	Attribute	Definition	Value	Card.
ExperienceApp-type				
	AppID	Content Identifier associated with the Basic Asset Metadata describing this Program.	md:ContentID-type	0..1
Type		Type of this title (see below)	xs:string	
AppGroupID		Reference to the App Group that references the application.	manifest:AppGroupID-type	
AppName		Readable name of the application. This may be localized with the language attribute.	xs:string	0..n
	language	Language of the AppName.	xs:language	0..1
Rating		Basic Metadata Content Rating. Common Rating encoding should be used.	md:ContentRating-type	

Type uses the following encoding

- 'nav' – Navigation Application; that is, an application used to navigate the experience.
- 'game' – Game
- 'other' – A type not otherwise defined here.

If AppName/@language is absent and a single Experience/@language is present, that language can be assumed. If there is no @language match, and one AppName contains no @language attribute, that AppName should be used. At most one AppName can be included without an @language attribute.

8.3.3 Gallery

The user interface for the presentation of images is called a Gallery. The Gallery contains enough information to provide a simple display of images.

A gallery contains

- Name – Used for gallery selection)
- Picture Group – Images associated with Gallery. The Gallery will include all images in the Picture Group.
- Background – Image or video background with optional audio.
- Auto-advance timing – If system is to display images automatically, how long to dwell on each slide.

Element	Attribute	Definition	Value	Card.
Gallery-type				
	GalleryID	Identifier that uniquely identifies this instance of Gallery-type. This is used as a reference for presentation implementations. For example, a menu intended to be associated with this gallery could refer to this GalleryID.	md:GalleryID-type	0..1
Type		Type of the gallery. Reserved for future use.	xs:string	0..1

PictureGroupID		Picture Group containing Pictures for gallery	manifest:PictureGroupID	
GalleryName		Title of Gallery	xs:string	0..n
	language	Language of gallery	xs:language	0..1

8.3.4 ExperienceChild

Each ExperienceChild element references a child Experience. The model is described in Section 8.1.

Element	Attribute	Definition	Value	Card.
ExperienceChild-type				
Relationship		Defines the relationship between the parent and child Experiences	xs:string	0..1
Experience		Child Experience	manifest:Experience-type	(choice)
ExperienceID		Reference to child Experience	manifest:ExperienceID	

Relationships are the mirror image of Common Metadata BasicMetadataParent-type/@relationshipType. That is BasicMetadata reference child to parent while Experiences point parent to child. The same controlled vocabulary used. Relationships uses the controlled vocabulary for BasicMetadataParent-type/@relationshipType as defined in [TR-META-CM], Section 4.1.4.2.

For example, if the Experience is a series and the child is season, the ‘isseasonof’ is used. If the Experience is a season and the child is an episodes, the ‘isepisodeof’ relationship is used.

9 MAPPING ALIDS TO EXPERIENCES

A Logical Asset defines a set of content. The mapping from ALID to Experience is performed through a Logical Asset identifier (ALID) that identifies an entire set of content to the Experiences associated with that content.

An Avail uses an ALID to identify the content associated with that Avail. Consequently, this map is used to map Avails to Experiences.

9.1 ALID-Experience Map List

The AvailsExperienceMaps element contains one or more instances of ALIDExperienceMap.

Element	Attribute	Definition	Value	Card.
AvailsExperienceMapList-type				
ALIDExperienceMap		Each instance contains a collection of mappings	manifest:ALIDExperienceMap-type	1..n

9.2 ALID-Experience Map

This type maps ALIDs to Experience through IDs.

As multiple ALIDs can map to the same content (i.e., same content, different business rules) and an Experience can be associated with multiple ALIDs, so the mapping between ALIDs and Experiences is many-to-many. However, it is recommended that all ALIDs map to one and only one Experience. That is, while an Experience can satisfy multiple ALIDs, for any given ALID there should be only one Experience.

Element	Attribute	Definition	Value	Card.
ALIDExperienceMap-type				
ALID		Logical Asset ID included in an Avail.	md:AssetLogicalID-type	1..n
ExperienceID		ID of the Experience associated with the ALID.	manifest:ExperienceID-type	1..n
	condition	The condition of the Experience in the context of this ALID. (see below)	xs:string	0..1

The @condition attribute indicates the offering condition of given experiences. This is the means to communicate the Experience that reflects the user's ability to acquire and access content. The primary use case for @condition is to distinguish which Experiences to use during different windows. Other applications are allowed.

The following values for @condition should be used for the associated conditions. For example, if the condition is a pre-offer, use the term 'Pre-offer'. Note that 'asset' refers to logical asset:

- 'Unavailable' – Asset cannot be sold or fulfilled. Typically no information is shared.
- 'Pre-offer' – Asset is intended for offer at a future date. Typically, information is shared, but with no opportunity to acquire. This also applies to holdbacks where asset cannot be sold or fulfilled.
- 'Pre-order' – Asset can be sold, but not fulfilled.
- 'Hold' – Asset has been sold, but it cannot be fulfilled. This applies to both pre-orders and holdbacks.
- 'For-sale' – Asset can be sold and fulfilled. This is the default and is assumed if @condition is absent.
- 'Acquired' – Asset has been acquired (through sale or promotion) and can be fulfilled. Acquired also covers the state where content has been purchased, it can be fulfilled, but not sold.

10 DELIVERY FILE MANIFEST

Content is generally delivered in multiple files, referred to here as a Package. The File Manifest describes the various pieces associated a Package.

This structure assumes the following files:

- Manifest file – A file with data described in this section
- Metadata files
- Media files – audio, video, subtitle, apps, images, text, interactivity/apps)
- Avails files
- Ancillary files (any other files)

To group files, there is the concept of a Package. A Package is all the files contained within the manifest, including the manifest itself. A Package is identified with a unique PackageID.

File formats are not addressed here, but these types represent the expression of information in files.

10.1 FileManifestInfo-type

This defines the Manifest. The manifest includes the definition of a Package and defines the contents of the Package. This includes a listing of all files included together along with identifying information about each file.

Element	Attribute	Definition	Value	Card.
ManifestInfo-type				
PackageID		Unique identifier for package	xs:string	
PackageDateTime		Date and time package generated	xs:dateTime	
Publisher		Organization with whom the package is associated. This is the entity to be contacted with any inquiries associated with the Manifest.	md:OrgName-type	
AvalisEntryID		ID for any Avails associated with this Package.	md:id-type	0..n
TotalFilesInPackage		Count of files	xs:int	
FileInfo		Information about each file in manifest	manifest:FileInfo-type	1..n
ExceptionFlag		Human attention is required. See Comments.	xs:boolean	0..1

Comments		Any comments or instructions to accompany Package	xs:string	0..1

10.1.1 FileInfo-type

FileInfo-type is used to describe each file in the Package. The data in this element should correspond with physical attributes of the file. For example, a file's name corresponds with Location, its file extension or type embedded in the file corresponds with Type, and Hash can be generated from its contents.

The FileInfo-type information ensures that a file is correctly identified. As the Package may be delivered separately from other files, the FileInfo-type ensures the correct files are identified.

It is envisioned that Location will be used to facilitate network downloading of assets. The metadata is delivered without the file, and the filed is retrieved from Location.

Element	Attribute	Definition	Value	Card.
FileInfo-type				
Location		Information that can be used to locate the file. This may be excluded, if Identifier is sufficient to distinguish the file. Location is encoded in accordance with the Location-type definition above.	manifest:Location-type	
Identifier		Identifiers associated with this file.	md:ContentIdentifier-type	0..n
FileVersion		Version of the file. If absent '0' is assumed. A higher number indicates a newer version.	xs:nonNegativeInteger	0..1
FileDate		File version date or dateTime. This can be used to distinguish this version of the file as part of file identification.	xs:union(xs:date, xs:dateTime)	0..1
Type		Type of file	xs:string "manifest" "metadata" "media" "avail" "ancillary"	
Length		The length of the file in bytes. If a Hash is used, this is the exact number of bytes that was used when generating the Hash.	xs:nonNegativeInteger	

Hash		File hash of the entire file. Multiple hashes can be included.	md:Hash-type	0..n
MIMEType		MIME type of file	xs:string	0..1
WrapperFormat		Description of how file is packaged. If this is a file of type media, this value corresponds with Container Type from [CM], Section 6.2.1.1.	xs:string	0..1
ContainerMetadata		If file is a media container, ContainerMetadata includes information about how to decode the file.	md:ContainerMetadata-type	0..1
Delivery		Information about how the file is delivered.	manifest:FileDelivery-type	

Common Metadata defines the appropriate encoding for different hash methods. For packaged files, MD5 is the preferred method.

10.1.2 FileDelivery-type

This element describes how a file will be delivered.

Unless otherwise noted, FileInfo/Location is assumed to be the local filename.

Element	Attribute	Definition	Value	Card.
FileDelivery-type				
DeliveryMethod		The mechanism by which the file has been or will be delivered.	xs:string	
TransferLocation		Location where file will be transferred to or from.	xs:anyURI	0..1
Organization		Organization delivering file. This may be different from FileManifest/Publisher	md:OrgName-type	0..1
Contact		Contact information in case there are any comments, questions or problems.	md:ContactInfo-type	0..1
EstDeliveryDate		Estimated delivery date	xs:date	0..1
OtherInstructions		Other instructions for the receiving party	xs:string	0..1

The value of certain fields depends on the delivery method. Some DeliveryMethod values indicate file status rather than method of delivery. Encoding is as follows:

DeliveryMethod	Definition	TransferLocation	Other
http	retrieved via HTTP GET	URL where file is located	Local filename must match filename in path
ftp-pull	Recipient will retrieve file via FTP	ftp site	
ftp-push	Sender with send file via FTP	<not used>	
email	Sender will send via email	email file will be sent to in 'mailto:' format.	
physical		Filename on media. Full path if appropriate	
filesite	Intended for file sharing sites such as Dropbox	Reference to file in appropriate account	
other	Some other method agreed upon by parties		See OtherInstructions for details
delivered	This file was delivered in an earlier version of this File Manifest	n/a	EstDeliveryDate should include the date when delivered.
future	This file will be delivered at a future time. This is to inform the recipient to expect the file at a later time.	n/a	If known, EstDeliveryDate should include an estimate of when the file will be delivered.
removed	This file is no longer applicable and should be removed by the recipient.		

10.2 FileDeleteManifest

The FileDeleteManifest element of FileDaleteManifest-type is provided to allow the sender to signal to the recipient that files already delivered should be removed.

This structure is similar to the File Manifest, but each file listed is intended to be deleted. In general, the information in the File element should match the original FileManifest exactly.

Element	Attribute	Definition	Value	Card.
FileDeleteManifest-type			manifest:FileManifest-type (by extension)	
	type	Type of delete operation. There are currently no pre-defined values and may be used bilaterally.	xs:string	0..1
	reference	Reference information as defined bilaterially.	xs:string	0..1

Description		A description of this deletion manifest. This information is for reference, and will not generally result in human intervention.	xs:string	0..1
Instructions		Any instructions accompanying deletion manifest. The presence of this element should signal human intervention.	xs:string	0..1

11 MEDIA MANIFEST EDIT

Some workflows require that a Media Manifest be modified after its delivery. This section describes an element that enables the editing process.

Edits must be performed with care as they can corrupt the internal structure of a Media Manifest. For example, if an Inventory item is removed but is still referenced, but the Media Manifest may be syntactically correct but it will not be valid for playback. It is recommended that update workflows account for situations likely to cause corruption. The element includes features to support good processes. For example, the ManifestID and updateNum attributes can be used to ensure updates are applied in the correct order.

Specific update workflows are left to best practices and bilateral agreements.

11.1 MediaManifestEdit-type

The MediaManifestEdit element is of MediaManfestEdit-type. This element provide 1) attributes for managing the update workflow, 2) IDs of elements to objects, and 3) objects to add. All operations are performed on a single Media Manifest.

At least one of DeleteObjects and AddObjects must be included. Both may be included.

Element	Attribute	Definition	Value	Card.
MediaManifestEdit-type				
	type	Type of delete operation. There are currently no pre-defined values and may be used bilaterally.	xs:string	0..1
	reference	Reference information as defined bilaterially.	xs:string	0..1
	ManifestID	Unique identifier for this manifest	md:id-type	0..1
	updateDeliveryType	This indicates the Manifest includes just portions required for an updated. It is not a complete Manifest. The exact definition is subject to specific practices and is reference by this string.	xs:string	0..1
	updateNum	Version of this document. Initial release should be 1. This is a value assigned by the manifest creator that should only be incremented if a new version of manifest is	xs:integer	0..1

		released. If absent, 1 is to be assumed.		
	ExtraVersionReference	A string that describes the version of the extras.	xs:string	0..1
DeleteObjects		Objects to be deleted from Media Manifest.	manifest:MediaManifestEditDelete-type	0..1
AddObjects		Objects to be added to Media Manifest after DeleteObjects are deleted.	manifest:MediaManifestEditAdd-type	0..1
BasicMetadata		BasicMetadata instance representing and update. Specifics are left to best practices. One instance can exist for each ContentID.	md:BasicMetadata-type	0..n

11.1.1 MediaManifestEditDelete-type

The MediaManifestEditDelete-type lists identifiers of objects to be deleted from an existing Media Manifest.

All delete operations are performed before any Add operations.

Element	Attribute	Definition	Value	Card.
MediaManifestEditDelete-type				
AudioTrackID		Objects with this ID will be deleted.	manifest:AudioTrackID-type	0..n
VideoTrackID			manifest:VideoTrackID-type	0..n
SubtitleTrackID			manifest:SubtitleTrackID-type	0..n
ImageID			manifest:ImageID-type	0..n
InteractiveTrackID			manifest:InteractiveTrackID-type	0..n
PlayableSequenceID			manifest:PlayableSequenceID-type	0..n
PresentationID			manifest:PresentationID-type	0..n

PictureID			manifest:PictureID-type	0..n
GalleryID			manifest:GalleryID-type	0..n
AppGroupID			manifest:AppGroupID-type	0..n
ExperienceID			manifest:ExperienceID-type	0..n
ALIDExperienceMap		Matching map will be deleted.	manifest:ALIDExperienceMap-type	0..n
ContentID		Basic Metadata with this ID will be deleted.	md:ContentID-type	0..n
LocalizedInfoRef		LocalizedInfo elements matching Content, language and optionally region will be deleted. If Region absent, LocalizedInfo elements matching ContentID and language will be deleted, regardless of region.	manifest:MediaManifestEditLocRef-type	0..n

11.1.1.1 MediaManifestEditLoc-type

This element is designed to match LocalizedInfo elements within md:BasicMetadata-type. ContentID and Langauge must match exactly. Region is matches from general to specific. That is, if Region is expressed in terms of country, it matches all LocalizedInfo elements that contain regions in that country. If Region is expressed in countryRegion form, then only an exact match applies. If LocalizedInfo does not contain a Region element, then the presence of the Region element here will always fail to match.

Element	Attribute	Definition	Value	Card.
MediaManifestEditLocRef-type				
ContentID		Matches md:BasicMetadata-type/@ContentID	md:ContentID-type	
Language		Matches md:BasicMetadata-type/LocalizedInfo/@language	xs:language	
Region		Matches md:BasicMetadata-type/LocalizedInfo/Region	md:Region-type	0..1

11.1.2 MediaManifestEditAdd-type

The MediaManifestEditAdd-type is almost identical to the MediaManifest-type; the differences being the removal of attributes—these are promoted to the parent element—and the cardinality of top-level elements all being optional—certain elements are required in MediaManifest-type.

Element	Attribute	Definition	Value	Card.
MediaManifestEditAdd-type				
Compatibility		Same as MediaManifest-type	manifest:Compatibility-type	0..1
Inventory			manifest:Inventory-type	0..1
Presentations			manifest:PresentationList-type	0..1
PlayableSequences			manifest:PlayableSequenceList-type	0..1
PictureGroups			manifest:PictureGroupList-type	0..1
AppGroups			manifest:AppGroupList-type	0..1
Experiences			manifest:ExperienceList-type	0..1
ALIDExperienceMaps			Manifest:ALIDExperienceMapList-type	0..1

ANNEX A. TRACK SELECTION PROCESS

This section describes the intended use of Track Selection information found in the Presentation element.

The following stages occur in track selection:

1. The Device assigns a default System Language
2. A User optionally changes System Language; and may selects preferences such as audio and subtitle languages, and subtitle type
3. The Device selects default audio track and subtitle track (Primary Subtitling Presentation Track), if applicable
4. A User may optionally select specific audio track or subtitle track (Primary Subtitling Presentation Track)
5. The Device selects subtitle tracks for forced subtitles (Alternate Subtitling Presentation Track), if applicable
6. Playback can begin. User selections may require repeating some steps above. For example, changing tracks (Step 4) would require performing Step 5.

This Annex uses the following terminology:

- The following subtitle definitions are used to describe what is in a subtitle track
    - Forced Subtitle: A subtitle with only one instance of Inventory/Subtitle/Type where that instance equals ‘forced’.
    - Other Subtitle: A subtitle with no instances of Inventory/Subtitle/Type equal to “forced”
    - Mixed Subtitle: A subtitle with at least one instance of Inventory/Subtitle/Type equal to ‘forced’; and at least one instance of Inventory/Subtitle/Type not equal to “forced”
        - Within a Mixed Subtitle track, subtext and subpicture elements that are to be displayed as forced subtitles are referred to as ‘forced elements’ and elements that are not to be displayed as forced elements are referred to as ‘non-forced elements’

- From a User's perspective, subtitles are either "on" or "off", however, in both cases subtitle elements may be displayed. The following definitions indicate what subtitles elements are presented when subtitles are off and on, what tracks contain those elements, and what audio track contains audio for playback
    - Primary Subtitling Presentation Mode: corresponds to subtitles are "on". When in Primary Subtitling Presentation Mode, the Primary Subtitling Presentation Track will be presented.
    - Primary Subtitling Presentation Track: The subtitle track that is to be presented during Primary Subtitling Presentation. An Other Subtitle track or a Mixed Subtitle track will be decoded and presented during Primary Subtitling Presentation.
    - Alternate Subtitling Presentation Mode: corresponds to subtitles are "off". When in Alternate Subtitling Presentation Mode, only forced elements within the Alternate Subtitling Presentation Track will be presented (if any). An Alternate Subtitle can be forced subtitle elements within a Mixed Subtitle track or a Forced Subtitle track.
    - Alternate Subtitling Presentation Track: The subtitle track that includes the forced subtitle elements to be presented during Alternate Subtitling Presentation. Forced subtitle elements within a Mixed Subtitle track or all elements in a Forced Subtitle track will be presented during Alternate Subtitle Presentation. Note that for a Mixed Track, the Selected Primary Subtitle Track and the Selected Alternate Subtitle Track might be the same track.
- The following definition indicates what audio track contains audio for playback
    - Selected Audio Track: The audio track selected for play.

A.1. Defined Preferences

The following are Input Variables to default track selection and must be selected prior to default track selection.

- System Language (required)
- User Preferred Audio Type. The type of audio preferred by the user. Type enumeration is as per md:DigitalAssetAudioData-type/Type. By default this should be "primary"

- User Preferred Audio Language (optional) – User preference for audio language which applies to all DCCs
- User Preferred Subtitle Language (optional) – User preference for subtitle language which applies to all DCCs
- User Preferred Subtitle Type (optional) – The type of subtitle preferred by the User for the purposes of selecting default audio and subtitle tracks. Type enumeration is as per md:DigitalAssetSubtitleData-type/Type. By default this should be 'normal'.

Devices are assumed to have the following capabilities

- Allow a User to override Input Variables
- Allow a User to select a specific audio track
- Allow a User to select a specific subtitle track for Primary Subtitling Presentation
- Allow a User to turn "on" and "off" subtitles
    - When "On", decode and present the Primary Subtitling Presentation Track and display all forced and non-forced elements.
    - When "Off": decode and present the Alternate Subtitling Presentation Track and only display forced elements

A.2. Default Audio and Subtitle Track Selection

This section defines algorithms for selecting default audio track and default subtitle track.

Default tracks are selected prior to initial playback and prior to User's making specific tracks selections. Section A.4 defines an "Assumed Device" model to reflect expected device behavior during default track selection.

In a Playable Sequence, track selection is performed on the "primary clip" in accordance with Section 5.2.4 The selection is applied to other tracks where matching tracks exist. Presentation constraints ensures that Presentations will either have the same audio and/or subtitle tracks or a single audio track making track selection on these other tracks deterministic.

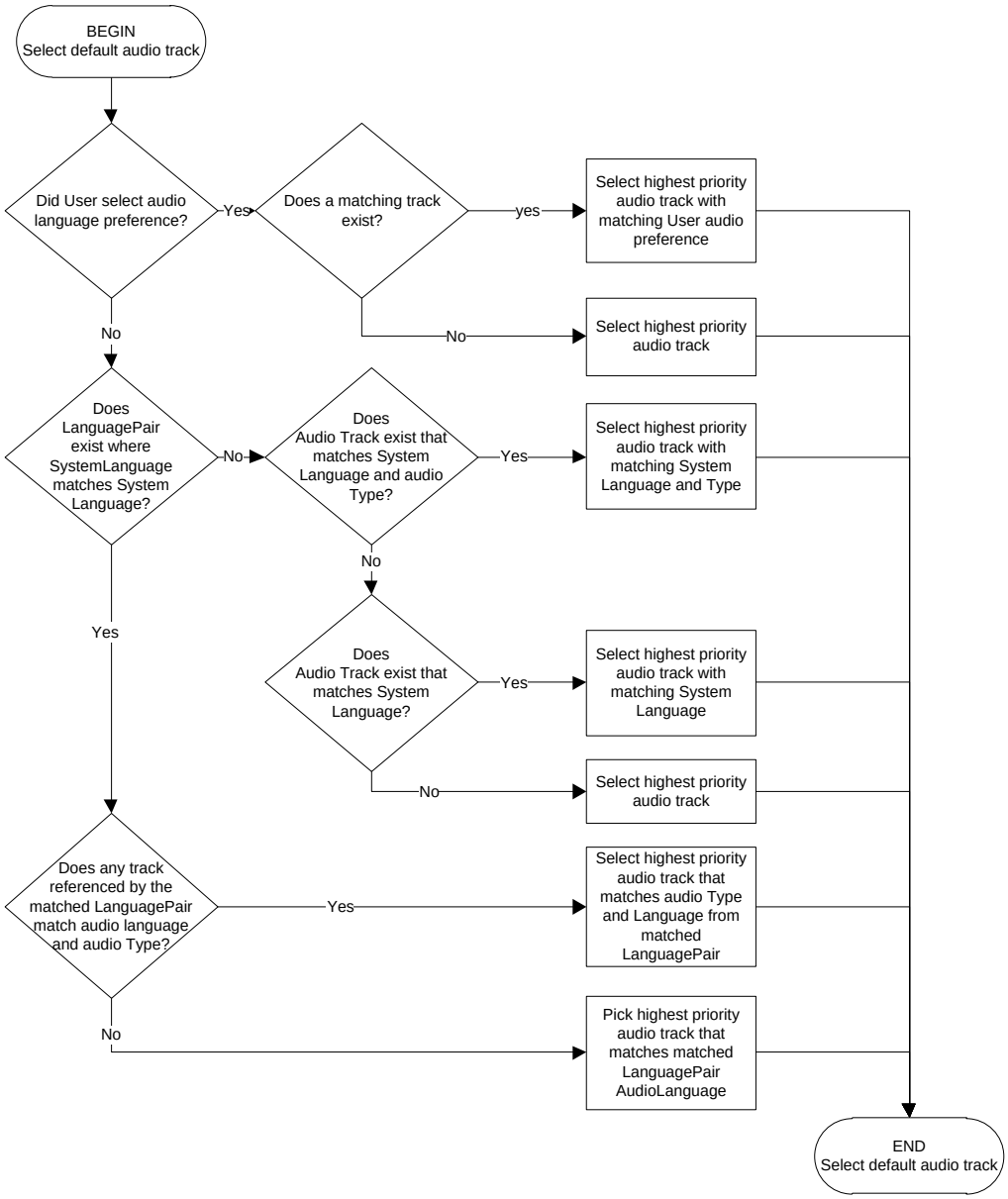
The following rules apply to the decision flow:

- When matching and selecting tracks, only tracks that are playable on the Device should be considered. Tracks that are not playable should be ignored. For example, a track with a CODEC not supported by the Device would never be selected.

- When multiple elements match equivalently
    - If there are additional User preference and at least one element matches this preference, filter elements based on the User preferences. For example, if the user prefers original audio tracks, and an original audio track matches other criteria, select that track (i.e., a track with Type='primary' and @dubbed absent or 'false').
    - Then, If elements are prioritized, return the element with the highest priority;
    - Otherwise, return the element that appears first in the metadata. For example, if a language lookup matches two LanguagePairs equally well, the first LanguagePair to appear in the Presentation would be selected.
- If more than one `Presentation` element is present, the `Presentation` element with `TrackSelectionNumber` equal to 0 is referenced for automatic default track selection.
- In the diagrams, when an audio track is "selected" it is selected as the Selected Audio Track. When a subtitle track is selected, it is selected as a Selected Primary Subtitle Track, unless otherwise noted.
- In conditions referring to matching tracks of a given language, Inventory/Audio/Language is used for audio language matching and Inventory/Subtitle/Language is used for subtitle language matching.
- In conditions referring to matching tracks of a given type Inventory/Audio/Type is used for audio Type matching, and Inventory/Subtitle/Type is used for subtitle Type matching.
- When referring to Tracks referenced by LanguagePair this refers to all tracks referenced by Presentation/AudioTrackReference that match Inventory/Audio/Language in union with tracks referenced by Presentation/SubtitleTrackReference that match Inventory/Subtitle/Language.

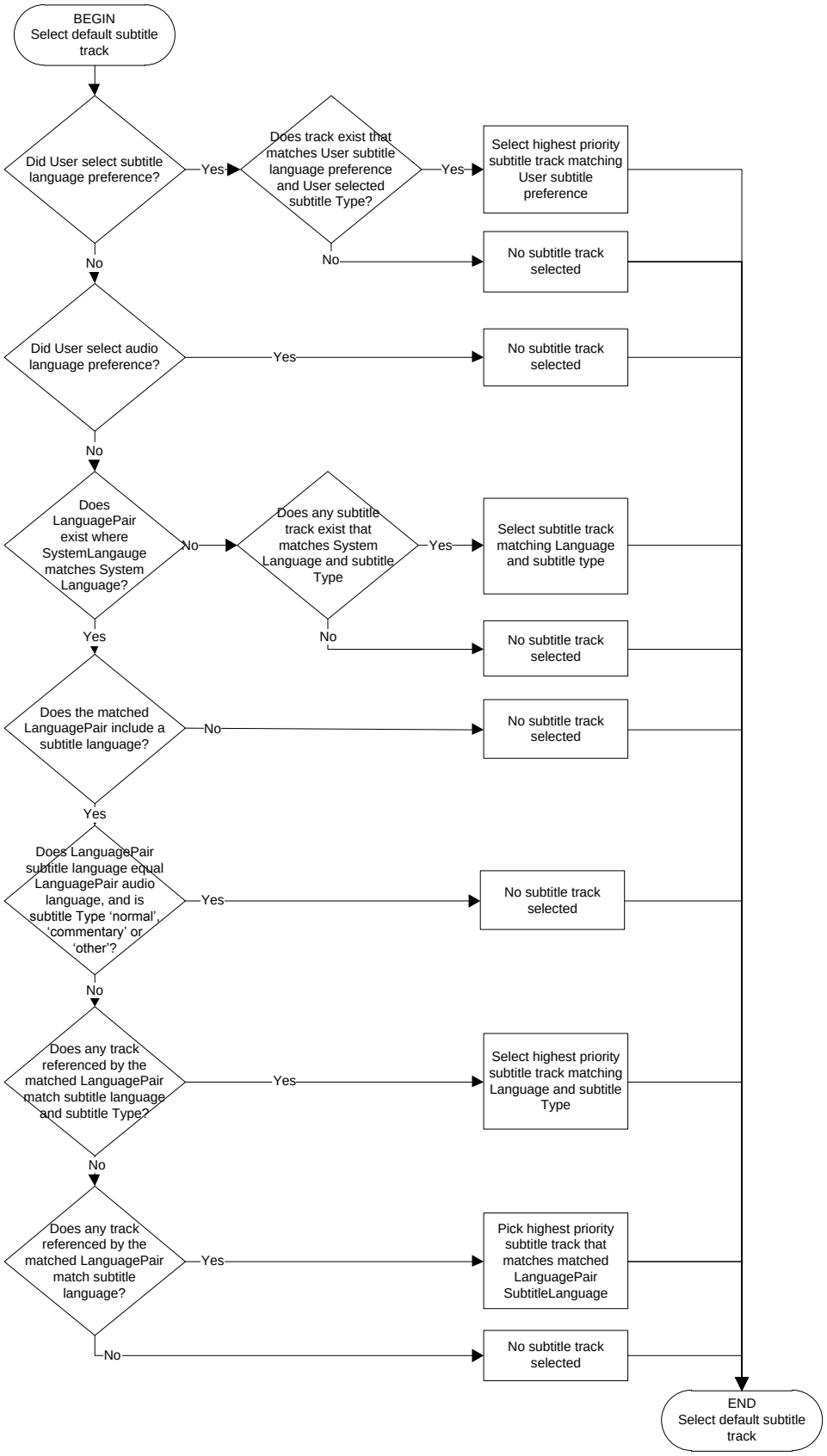
A.2.1. Default Audio Track Selection

This flow describes the assumed algorithm for selecting a Default Audio Track.



A.2.2. Default Primary Subtitling Presentation Track Selection

This flow describes the assumed algorithm for selecting a Default Subtitle Track.


BEGIN
Select default subtitle track
Did User select subtitle language preference?
Yes
Does track exist that matches User subtitle language preference and User selected subtitle Type?
Yes
Select highest priority subtitle track matching User subtitle preference
No
No subtitle track selected
No
Did User select audio language preference?
Yes
No subtitle track selected
No
Does LanguagePair exist where SystemLangauge matches System Language?
No
Does any subtitle track exist that matches System Language and subtitle Type
Yes
Select subtitle track matching Language and subtitle type
No
No subtitle track selected
Yes
Does the matched LanguagePair include a subtitle language?
No
No subtitle track selected
Yes
Does LanguagePair subtitle language equal LanguagePair audio language, and is subtitle Type 'normal', 'commentary' or 'other'?
Yes
No subtitle track selected
No
Does any track referenced by the matched LanguagePair match subtitle language and subtitle Type?
Yes
Select highest priority subtitle track matching Language and subtitle Type
No
Does any track referenced by the matched LanguagePair match subtitle language?
Yes
Pick highest priority subtitle track that matches matched LanguagePair SubtitleLanguage
No
No subtitle track selected
END
Select default subtitle track

A.3. Alternate Subtitling Presentation Track Selection

An Alternate Subtitle Track is used for Forced Subtitles.

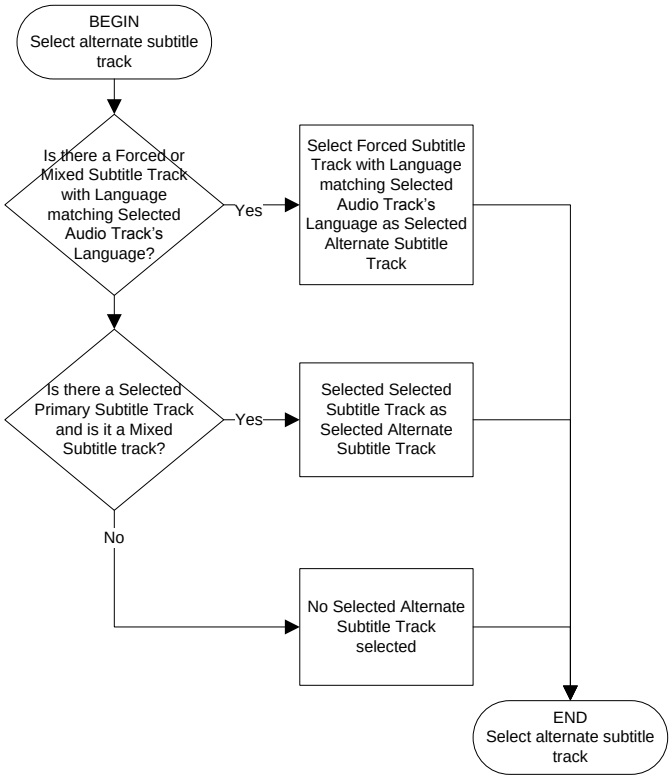
Forced subtitles are displayed either in conjunction with other subtitles, or when subtitles are turned off. That is, if subtitle is off and a suitable forced subtitle track (i.e., either a Forced Subtitle track or a Mixed Subtitle Track) is present, it will be displayed.

A forced subtitle track is expected to match the language of a selected audio track.

If a subtitle track contains information that allows differentiation between elements that are forced and not forced, then the forced subtitle track should be interpreted as the mixed track with only forced elements presented.

A.3.1. Select Alternate Subtitle Track

This flow describes the assumed algorithm for selecting the Alternate Subtitle Track.



A.4. Assumed Device Model

This section defines an "Assumed Device" that exhibits the desirable behavior for default track selection and playback.

A.4.1. Subtitle-specific Track Selection

Assumed Devices SHALL determine Primary Subtitling Presentation Track, as defined above and in accordance with Section A.4.4.

Assumed Devices SHALL determine Alternate Subtitling Presentation Track, as defined above and in accordance with Section A.4.4.

The Assumed Device SHALL be capable of playing all glyphs defined for the Subtitle.

As a recommended practice, text subtitle track SHALL NOT be considered playable if both of the following are true:

- the Subtitle Track Language matches a 'Language Subtags' of [DMedia], Annex D.2, Table D-1
- the Assumed Device does not support rendering of all glyphs that correspond to the Unicode Code Points defined for matching Language Subtag in [DMedia], Annex D.2, Table D-1

An Assumed Device SHALL NOT select tracks that are not playable.

A.4.2. Device Subtitling Mode

An Assumed Device SHALL be in either Primary Subtitling Presentation Mode or Alternate Subtitling Presentation Mode.

As an initial condition, unless otherwise specified, an Assumed Device SHALL be in Alternate Subtitling Presentation Mode.

If a Primary Subtitling Presentation Track is selected through the Subtitling Track Selection process (Section A.3.1), the Assumed Device SHALL be in the Primary Subtitling Presentation Mode, unless the User has opted to turn subtitles off. That is, when a Primary Subtitling Presentation Track is selected, the Device is in Primary Subtitle Presentation Mode unless the User selects otherwise.

When a Primary Subtitling Presentation Track is selected through the Subtitling Track Selection process (Section A.3.1), Assumed Devices SHOULD provide the means for a User to turn subtitles off (Alternate Subtitling Presentation Mode) and on (Primary Subtitling Presentation Mode).

A.4.3. Subtitle Playback

During playback, when in Primary Subtitling Playback Mode, the Assumed Device SHALL decode and present the Primary Subtitling Mode Track.

During playback, when in Alternate Subtitling Playback Mode and an Alternate Subtitle Presentation Track has been selected through the Subtitling Track Selection process (Section A.3.1), Assumed Devices SHALL decode and present forced elements as per [DMedia] Section 6 from the Alternate Subtitle Presentation Track. In the case of a Forced Subtitle Track, this is all elements.

Assumed Devices are not expected to decode and present more than one subtitle track simultaneously.

A.4.4. Audio and Subtitle Track Selection

The purpose of this section is to define behavior for Devices with respect to audio and subtitle track selection. This allows Devices to select default tracks consistent with the intent of the Content Providers.

Information is provided to assist Devices in selecting the appropriate audio and subtitle information. Information is contained in Presentation/TrackMetadata and Presentation/TrackSelections.

Track selection is made based on Device defaults (e.g., region and language), User preferences (e.g., language, accessibility), available tracks, TrackMetadata information and, if available, TrackSelections information. Generally, Users may override Device track selection. However, there are no overrides for Alternate Subtitle Presentation Track.

Additional terminology used in this section is defined at the beginning of his annex.

A.4.4.1. Default Track Selection

The Presentation element provides information that can be used to select default tracks in accordance with the Content Provider's intent. These data are defined in Section 5, and the assumed algorithm in Sections A.2 and A.3.

System Language and other language preferences SHALL be at least one language that can be represented as Language Tags as per [RFC5646] from the IANA Language Subtag Registry [IANA-LANG].

Assumed Devices SHALL set the System Language.

Assumed Devices SHOULD set the default System Language to match the user Interface language or Operating System language.

Devices SHOULD provide the means for Users to set the System Language.

Devices SHOULD provide System Language settings to include dialects. For example, “Latin Spanish” (IANA tag ‘es-419’) should be distinguished from “Castilian Spanish” (IANA tag ‘es’).

Devices SHOULD provide the means for User to select an audio preference for Type compatible with Presentation/TrackMetadata/Track/Audio/Type. For example, the User can select a Type of ‘dialogcentric’.

Devices SHALL provide the means for User to select a subtitle preference for Type compatible with Presentation/TrackMetadata/Track/Subtitle/Type. For example, the User can select a Type of ‘SDH’.

Devices SHALL provide the means for User to select a language preference for audio.

Devices SHALL provide the means for User to select a language preference for subtitles.

Devices SHOULD provide the means for User to select a preference dubbed audio or original audio.

Prior to playback, Assumed Devices SHALL use track selection methods described in this Annex to select default audio and subtitle tracks.

A.4.4.2. User Track Selection

Assumed Devices SHALL provide the ability for a User to select the Selected Audio Track.

Assumed Devices SHALL provide the ability for a User to select the Primary Subtitling Presentation Track.

Assumed Devices SHALL NOT allow a User to select a Forced Subtitle as the Primary Subtitling Presentation Track.

Assumed Devices SHALL NOT allow the User to select the Alternate Subtitling Presentation Track.

Assumed Devices MAY persistently store User track selections for later use. In the future, these can be used in lieu of default track selection for this DCC.