Best Practice: Constructing Identifiers

DRAFT

CONTENTS

1 Introduction 4
1.1 References 4
2 Words of encouragement and Wisdom 5
2.1 IDs are just strings, don't overthink it 5
2.2 IDs are designed for computers, not humans 5
2.3 Don't look 'inside' IDs—the metadata is unreliable 5
2.4 Why identifiers have weird names like ALID 5
3 different Types of Identifiers 6
3.1 Content ID 6
3.2 Experience ID 7
3.3 Logical Asset ID (ALID) 7
3.3.1 EIDR-based ALIDs 8
3.3.2 Avoiding ALID collisions 8
3.3.3 Why ALID is distinct from ContentID 9
3.3.4 ALID Rules and Recommendations 9
3.4 Track, Image, Text and Interactive IDs 9
4 'md:' format IDs 11
5 Using EIDR IDs 12
5.1 EIDR Format (EIDR-s, EIDR-x and URNs) 12
5.2 EIDR Object Type 12
5.3 EIDR in Avails 13
6 Constructing Asset Identifiers from top-level IDs 14
6.1 Derived Experience IDs 15
6.2 Derived Presentation IDs 15
6.3 Derived Component IDs 16

This work is licensed under a Creative Commons Attribution 3.0 Unported License.

NOTE: No effort is being made by the Motion Picture Laboratories to in any way obligate any market participant to adhere to this specification. Whether to adopt this specification in whole or in part is left entirely to the individual discretion of individual market participants, using their own independent business judgment. Moreover, Motion Picture Laboratories disclaims any warranty or representation as to the suitability of this specification for any purpose, and any liability for any damages or other harm you may incur as a result of subscribing to this specification.

REVISION HISTORY

Version	Date	Description
1.0	May 25, 2017	Initial publication. Derived from Best Practices for Delivery

DRAFT

1 INTRODUCTION

This document describes best practices for creating identifiers. This document provide an informal description of these identifiers to promote a more intuitive understanding of the various identifiers, how they are created and how they are used.

It is applicable to: Avails, Common Metadata, Media Entertainment Core (MEC), Media Manifest, Media Manifest Core (MMC), Cross-Platform Extras (CPE) and other MovieLabs Digital Distribution Framework (MDDF) Specs. This document assumes familiarity with the referenced specifications.

1.1 References

[CM]	Common Metadata, TR-META-CM, www.movielabs.com/md/md
[Manifest]	MovieLabs Common Media Manifest Metadata v1.4, TR-META-MMM, www.movielabs.com/md/manifest
[Avail]	EMA Content Availability Data (Avails), TR-META-AVAIL, www.movielabs.com/md/avails
[MEC]	Media Entertainment Core, TR-META-MEC, www.movielabs.com/md/mec
[EIDR-UG]	EIDR 2.0 Registry User's Guide, eidr.org/technology/
[EIDR-ID]	EIDR ID Format, eidr.org/technology/
[RFC7302]	RFC 7302, Entertainment Identifier Registry (EIDR) URN Namespace Definition, https://tools.ietf.org/html/rfc7302
[DOI]	Digital Object Identifier (DOI), www.doi.org
[ISO26324]	ISO 26324:2012, *Information and documentation -- Digital object identifier system*
[EIDR-V]	How to Use EIDR to Identify Versions for Distribution Purposes: Edits, Languages and Regional Releases (FAQ), eidr.org/technology
[EIDR-Format]	EIDR: ID FORMAT Ver. 1.3, http://eidr.org/documents/EIDR_ID_Format_v1.3.pdf
[EIDR-Proxy]	EIDR and the DOI Proxy, http://eidr.org/documents/EIDR_and_the_DOI_Proxy.pdf

DRAFT

2 WORDS OF ENCOURAGEMENT AND WISDOM

There are multiple identifiers associated with these specs. These are necessary to sustain the information model, but sometimes they seem overwhelming. The most common pitfall in identifiers is overthinking.

2.1 IDs are just strings, don't overthink it

In most cases, the ID is just a string and the only relevant action is to match it to a field somewhere else. Where identifiers get tricky is making sure that identifiers uniquely identify one object. Fortunately, there others have done much of the heavy lifting (e.g., EIDR) and this document provides some recipes.

2.2 IDs are designed for computers, not humans

IDs are designed to be computer readable. The most important factors are that the ID be well formed and unique with the scope of usage.

Some conventions such as ID type are structurally important as they allow unique identifiers to be assigned using the same root identifier (e.g., EIDR ID).

Where possible, we used conventions that would enhance the readability of the ID. For example, you can distinguish a Content ID from an Experience ID by the type embedded in the string. You can determine an ID type by viewing the scheme.

2.3 Don't look 'inside' IDs—the metadata is unreliable

Never use information in the ID lieu of metadata, *except for human readability and debugging.*

That is, you might find a track type, language or some other useful information in the ID. *Don't use it!* For example, you might see an ID that looks like this: `md:subtrack:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:subtitle.de.large`. We certainly hope this is a German large subtitle, but it might not be. Rely on the appropriate metadata (e.g., Track/Subtitle/Language). The only reliable aspect of any identifier is that it is unique.

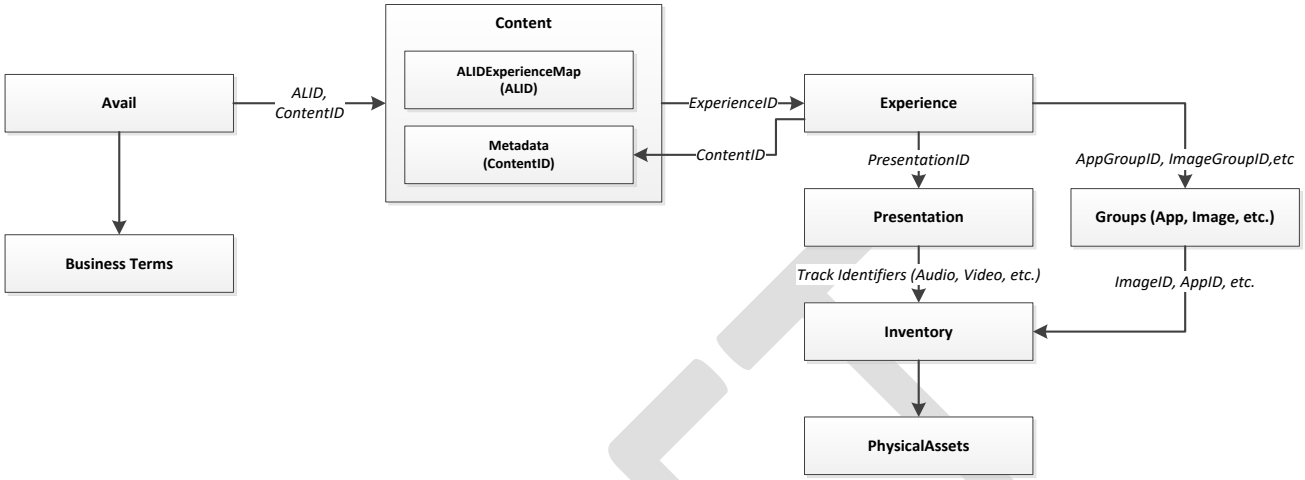
That's not to say the information isn't useful. If an EIDR doesn't match the title at hand, that's a red flag for a QC function, and it's perfectly reasonable to use these data for sanity checks.

2.4 Why identifiers have weird names like ALID

The identifiers are named somewhat abstractly because when we tried more meaningful names everyone got confused by preconceived notions. You'll note that terms like 'product' are rigorously avoided because they are not precise and people have preconceived notions of what they mean. The fact that nobody has a preconceived definition of "logical asset" allows us to define it precisely.

3 DIFFERENT TYPES OF IDENTIFIERS

The following diagram shows the relationship between identifiers:



"Content" is at the center. Avails avail Content and Experiences fulfill Content. Logical Asset IDs (ALIDs) and Content IDs identify Content. Although ContentID and ALID are similar they represent distinct concepts and are sometimes different. To understand the distinctions, a careful reading of the sections below is strongly recommended.

Avails reference Content and Business terms. The important concept is that although there may be multiple assets and transactions, *each Avail corresponds with exactly one set of Content*.

Experiences reference Content via ContentID. But, more important to Avails, the ALIDExperienceMap allows one to determine which Experiences can fulfill an Avail.

From the Experiences, one can map the Physical Assets that are used to deliver the experience to the consumer.

3.1 Content ID

The Content ID (also referred to as ContentID and CID) is used exclusively *as a reference to metadata*. As such, it's important for User interface, but it has no actual function in other parts of the workflow, such as Rights management, licensing, distribution, and packaging.

A Content ID can describe an actual work, such as a movie, a TV episode or a short subject, or it can be an object used to group things such as a season, series or franchise.

Everything identified by a Content ID should have Basic Metadata record. Basic Metadata is formally defined in the Media Entertainment Core [MEC].

Keep in mind that Metadata is used by every User Interface including those for processing orders and order resolution.

ContentID covers a particular version of the work, regardless of where that work is released. Metadata can contain multiple instances if localized information, so on ContentID can cover many regions and languages. Only if the edit changes, is a new ContentID required.

3.2 Experience ID

The Experience ID identifies an Experience. It's not any more complicated than that, but is essential to have the capability to refer to an Experience.

In this flow, one gets from an Avail to an Experience using the Media Manifest's AvailExperienceMap.

3.3 Logical Asset ID (ALID)

The Logical Asset Identifier (ALID) defines the content that a person has Rights to; that is, what is the subject of an Avail. The ALID goes into the Avail and it is used to map Avails to Experiences. In other words, The ALID is the glue that ties everything together.

To understand an ALID, it is useful to look at what the ALID identifies. When the studio decides the assets in an Avail, an ALID is created. The ALID might cover a single asset (e.g., a movie) or it can be an organized collection of assets (e.g., a season) or an arbitrary collection of assets (e.g., 'movies starring Kirk Douglas'). At the time of the Avail, it is not necessary for the assets associated with the ALID to be fully defined—it can be updated later. However, by the time the assets are to be fulfilled, the ALID must be well defined.

How does one know the assets in the ALID? There are four ways;

- The Avail's AssetList (XML only) lists the assets. This is intentionally somewhat loose because it is understood that Avails are published before products are fully defined.
- The Media Manifest maps ALIDs to Experiences (ALIDExperienceMap). From there, the Media Manifest (Experience, Presentation, Inventory, etc.) fully defines what is addressed by the ALID
- EIDR unambiguously identifies what an ALID refers to. EIDR definitions correspond to the various assets an Avail must address (i.e., movies, episodes, seasons, arbitrary collections, etc.). We recommend EIDR IDs because of the flexibility and precision. Note that EIDR definitions can be created in rough form, if full information is not available at Avail time, and revised later.
- One or more other IDs in the Avail (e.g., AltID) contains information the retailer can use to figure out what assets are associated with this ALID.

Furthermore, to be clear, the following describe what an ALID is not:

- An ALID is not necessarily unique to a combination of assets and terms. A studio might assign multiple IDs to the same combination. This might be useful if the same Avail structure is offered to different Retailers.

- An ALID is not a content identifier. In many cases, an Avail will correspond with a single asset. However, the IDs are not interchangeable. Asset is fundamentally different from Asset+Terms. Put differently, an Avail uses Content IDs to refer to items that is being made available. It may seem redundant, but it's not and someday you'll thank me for this.

It is also important to unambiguously identify the content referred to by the Avail, even if it is not fully defined at Avail time. This is what the Distribution Entity must deliver and what the Retailer may offer. The identifier that refers to the collection of assets associated with an Avail is an ALID.

An ALID might be used by the studio or various service providers partnering with the studio so Avails must be identified in a globally unique manner. Consequently, the ALID should be defined globally unique. The EIDR-x form should be used to distinguish between the different Avails for the same content.

3.3.1 EIDR-based ALIDs

The simplest way to create an ALID is `md:alid:eidr-x:`<EIDR>`:`<extension>, where <EIDR> is the Edit-level EIDR and <extension> is something unique. For example, `md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:craigsmovies.com_july_NorthAmerica`. Note that the extension part distinguishes this ALID from other ALIDs for the same title. If it's not obvious why you need the extension part, please read on.

3.3.2 Avoiding ALID collisions

The potential exists for more than licensors to avail the same title (same EIDR); generally, across windows or across regions. Licensors are required to avoid conflicts in Avails they produce, but to avoid collisions between licensors it is necessary to use more than ALID to uniquely identify an Avails. We have established the convention of requiring uniqueness for the combination of ALID and Licensor.

For example, consider two studios (studio1 and studio2) licensing `md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H`, one in the US and one in the UK. As they are, unfortunately, using the same ALID, the only way to distinguish them is the combination of ALID and licensor: {`md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H`, studio 1} and {`md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H`, studio 2}.

More precisely, an ALID can cover correspond with individual assets or sets of assets (e.g., seasons); as well as multiple sets of terms (e.g., HD EST, HD VOD, SD EST, SD VOD, etc.). In the context of an Avail in combination with Licensor, the ALID uniquely identifies a combination of assets and business terms.

When an Avail or an Avail update arrives, the context of a Licensor allows a retailer to know which avail is to be created or updated.

3.3.3 Why ALID is distinct from ContentID

The simple answer is that they are different both conceptually and in practice. One way to look at is that an ALID defines the entitlement (and fulfillment associated with the entitement), and the ContentID defines metadata. What's confusing is that most of the time, they're the same thing. But, not always.

In general, when ALIDs and ContentIDs diverge, there are multiple ALIDs to a single ContentID.

It's always helpful to keep ContentID constant. That facilitates metadata databases, reference to third party metadata and so forth. Keeping track of multiple ContentIDs leads to all kinds of efficiency and quality issues. For example, when a correction is made in one place, it's unlikely to propagate to other instances.

That said, why would you have multiple ALIDs? Studios can use ALIDs to control the scope of the entitlement. For example, if the entitlement is regional, then distinct ALIDs are generated per region. If the entitlement is subject to window constraints, ALIDs can be generated for different windows. If desired, the studios can even generate distinct ALIDs for individual retailers to ensure that exclusive fulfillment options are enforced. Note that some of these features can be addressed via other mechanism, particularly in the Media Manifest, but that does not exclude their necessity in Avails and entitlement databases.

3.3.4 ALID Rules and Recommendations

The combination of ALID and Licensor SHALL be globally unique. ALID SHOULD be globally unique. A Licensor SHALL NOT use an ALID for a different combination of assets and terms. An Avail associated with an ALID SHALL have only one disposition.

An ALID SHOULD comply with the ALID format above. This is not a strict requirement, but it will make global uniqueness much easier and avoid us IDs in the wrong context. Note that UUIDs avoid the first issue, but not the second.

An ALID SHOULD be based on an EIDR ID.

An ALID defines the product. See Section **Error! Reference source not found.** for information on deriving ALID from Excel Avails. Accordingly, the same ALID may be used in more than one Avail.

When an ALID refers to a single asset (e.g., a movie, a TV season or a TV episode), it should contain the same EIDR ID as the asset's ContentID.

3.4 Track, Image, Text and Interactive IDs

As we move from the abstract references to content to physical components, it becomes necessary to define IDs specifically for those components. To avoid erroneous references (e.g., referencing an audio track when a video track was intended), we define IDs for each track type:

DRAFT

Audio, video, subtitle, etc. We also define ID types for non-track media such as images, text and interactive entities. Section

Sometimes IDs are internal to a document, such as internal to a Media Manifest. In that case, discipline around IDs is necessary for consistency. The conventions for this are discussed in Section 4 and 6.

IDs can be particularly important when referencing external files, such as an Interoperable Master Format (IMF) file. We have a separate Best Practice for referencing IMF from Media Manifest.

Physical media (i.e., Manifestations) can also be referenced via EIDR. There are a variety of EIDR Best Practices that can be found at www.eidr.org/technology. Also, Section 5 discusses EIDR use.

DRAFT

4 'MD:' FORMAT IDS

In many cases, any string will work for IDs, but to facilitate QC, debugging and to avoid collisions, we have defined conventions for ID formatting. The structure used is defined in Common Metadata [CM], Section 2.1. We extend the format to include *types* that are required for Media Manifest.

Identifiers are based on <MDID> where

<MDID> ::= "md:"<type> ":"<scheme>":"<SSID>

<scheme> and <SSID> are defined in [CM], Section 2.1. For example, 'eidr-s' and 'eidr-x'

In addition to the <type> definitions on Common Metadata, the following should be used for their respective ID types:

ID	<type>	Constrained form
PackageID	packageid	"md:packageid:"<scheme>":"<SSID>
ProductID* or ALID	alid	"md:alid:"<scheme>":"<SSID>
PresentationID	presentationid	"md:presentationid:"<scheme>":"<SSID>
Experience	experiencedid	"md:experienceid:"<scheme>":"<SSID>
TransactionID	transationid	"md:transactionid:"<scheme>":"<SSID>

*ProductID is sometimes used in lieu of ALID (e.g., in the Avails spreadsheet). It is a Logical Asset identifier and consequently uses the ALID format.

The following additional naming conventions are used optionally for identifiers in Media Manifest:

ID	<type>	Constrained form
App Group	appgroupid	"md:appgroupid:"<scheme>":"<SSID>
Picture Group	picturegroupid	"md:picturegroupid:"<scheme>":"<SSID>
Playable Sequence	playablesequence	"md:playablesequence:"<scheme>":"<SSID>
Video Track ID	vidtrackid	"md:vidtrackid:"<scheme>":"<SSID>
Audio Track ID	audtrackid	"md:audtrackid:"<scheme>":"<SSID>
Subtitle Track ID	subtrackid	"md:subtrackid:"<scheme>":"<SSID>
Interactive Track ID	interactiveid	"md:interactiveid:"<scheme>":"<SSID>
Image ID	imageid	"md:imageid:"<scheme>":"<SSID>
Picture ID	pictureid	"md:pictureid:"<scheme>":"<SSID>
Gallery ID	galleryid	"md:galleryid:"<scheme>":"<SSID>

DRAFT

5 USING EIDR IDS

Information on using EIDR IDs can be found in the references [EIDR-UG], [EIDR-ID] and [EIDR-V]. Additional information is provided in this section.

5.1 EIDR Format (EIDR-s, EIDR-x and URNs)

EIDR IDs are based on the Digital Object Identifier (DOI), standardized as ISO 26324 [ISO26324] and described [DOI]. A full EIDR ID looks something like this: `10.5240/5EE7-A973-819A-DC1A-CDD8-H`. It can be converted to a resolvable form to obtain metadata: https://resolve.eidr.org/EIDR/object/10.5240/5EE7-A973-819A-DC1A-CDD8-H. It can also be resolved as https://doi.org/10.5240/5EE7-A973-819A-DC1A-CDD8-H and https://doi.org/urn:eidr:10.5240:5EE7-A973-819A-DC1A-CDD8-H. For more information see [EIDR-format] and [EIDR-Proxy].

However, the Common Metadata identifier format uses URN format which requires percent encoding for certain characters such as slash ('/' = '%2f'). So, it's easier to use a form of EIDRs that does not include the "10.5240/" prefix. The forms relevant here are EIDR-S (for 'short') and EIDR-X (for 'eXtended'). These and other forms are defined in [EIDR-ID].

These forms are as follows:

"`md:alid:eidr-s:`"<EIDR suffix>

"`md:alid:eidr-x:`"<EIDR suffix>":"<extension>

For example,

```
md:alid:eidr-s:5EE7-A973-819A-DC1A-CDD8-H
md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:craigsmovies.com_july_NorthAmerica
```

The short form is used to express an EIDR ID with no other information. The extended form is used when identifiers are derived from the EIDR ID, but additional information is required to add uniqueness. Examples will follow throughout this document.

In many cases, it is desirable to derive a new ID from an existing EIDR ID rather than obtain a unique EIDR ID for an object. In these cases, eidr-x is used. For example a unique EIDR cannot be created for an entitlement so EIDR-X is used to create unique ALIDs.

When EIDR is not part of a constructed ID, the preferred form is the EIDR URN as defined in [RFC7302]. An EIDR in URN format looks like this: `urn:eidr:10.5240:7791-8534-2C23-9030-8610-5`.

5.2 EIDR Object Type

When using EIDR it is important to use the correct EIDR Object Type. EIDR Object Types are defined in [EIDR-UG], Section 4.

EIDR Types should be assigned based on the asset types. In general if there is a single asset (or season) the EIDR type should represent a particular Edit of that asset. If there is a collection of objects, it should be an EIDR Compilation (collection of assets). However, if this is impractical, the type of the primary asset can be used.

Asset Type	EIDR Type	Alternate EIDR Type
Movie	Edit	
Episode	Edit of Episode	
Season	Season	
Movie with extras	Compilation	Edit
Episode with extras	Compilation	Edit
Season with Extras	Compilation	Season

5.3 EIDR in Avails

In general, the EIDR-X form is most appropriate for ALID. This is because there may be multiple Avails for a single title. The <extension> part makes the Avail unique. For example, let's assume a single title: Do the Right Thing, EIDR Edit = 5EE7-A973-819A-DC1A-CDD8-H. ALID might be:

```
md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:craigsmovies.com_july_NorthAmerica
md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:craigsmovies.com_july_Europe
md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:craigsmovies.com_aug_NorthAmerica
md:alid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:craigsmovies.com_aug_Europe
```

Because the extended EIDR (EIDR-X) form was used, it is possible to distinguish retailers, timing and region. This is done in a human-readable form, but it could also be done with something computer-friendly.

Naming conventions in extensions are intended to support uniqueness and human readability and SHOULD NOT be parsed automatically to extract information.

To simplify processing, Avails provides the means to provide multiple levels of EIDR in the metadata. For example, Asset/Metadata includes TitleEIDR-URN and EditEIDR-URN. Analogies are in EpisodeMetadata, SeasonMetadata and SeriesMetadata.

6 CONSTRUCTING ASSET IDENTIFIERS FROM TOP-LEVEL IDS

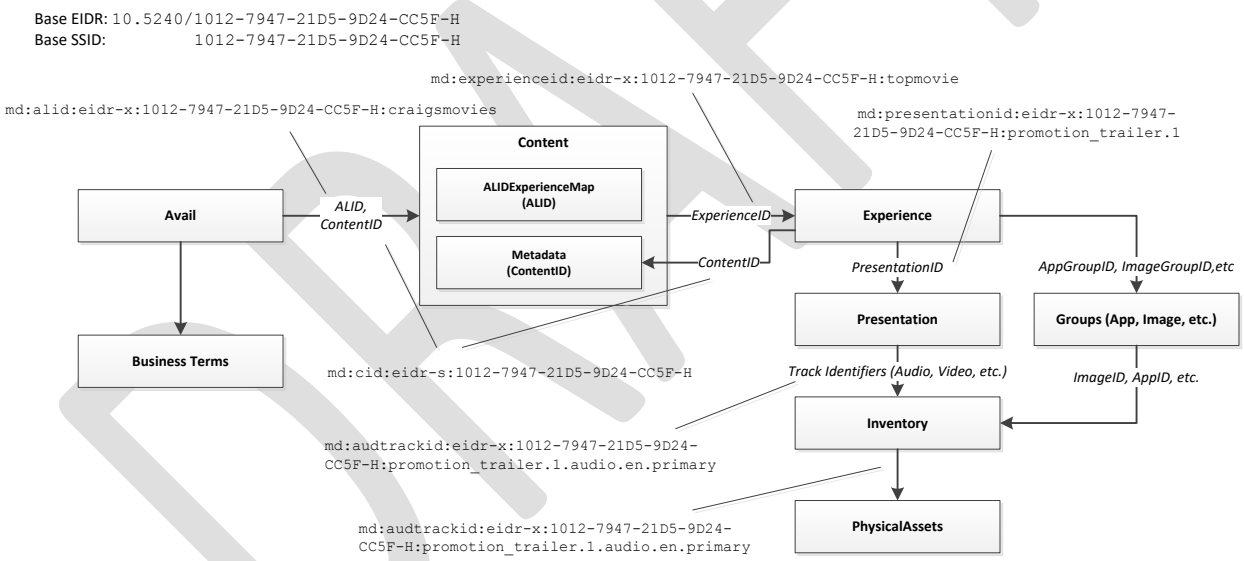
This section defines a recommended practice for building the IDs required by the Manifest. This practice is designed to make the manifest easier to create, read and maintain.

This practice should be used when unique identifiers do not already exist for an object. For example, if there is an ID for a movie's trailer, it should be used. However, it there is no ID for that trailer, this practice describes how to construct a unique trailer ID from the movie's ID.

The identifier structure is designed to handle all cases, but there are simplifications that will simplify identifier creation and interpretation.

ContentID, ALID and Experience ID are generally rooted in the same Content so the same base identifier can be used for all. In some cases, not discussed in this document, ExperienceIDs would have their own EIDR ID and therefore have a different base. So, it is important to not make assumptions about base identifiers. It is always possible to tell identifiers apart (e.g., ContentID from ALID) because they are structured differently (e.g., md:cid… vs. md:alid…).

Following is an example of identifiers using this convention.



ID form remains consistent with the other ID recommendations. That is, the ID is of the form:

<MDID> ::= "md:"<type> ":"<scheme>":"<SSID>

This practice addresses the <SSID> part. The following form is used. Note: It looks complicated, but you'll see from the examples it's quite simple.

<SSID>::= <Experience SSID> | <Presentation SSID> | <Component SSID>

When constructing an identifier, it is important that the completed <SSID> is formatted in compliance with the <scheme>.

The following examples use EIDR IDs; for example, `md:presentationid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:trailer.1`. Other identifier schemes also work, provided one can distinguish between the pure identifier and one with an extension. This practice suggest appending '`-x`' to the org-specific scheme. For example, `org:craigsmovies.com` becomes `org:craigsmovies.com-x` as shown in this example, `md:presentationid:org:craigsmovies.com-x:abc123:trailer.1`. Of course if an EIDR manifestation ID exists, that should be used instead (eidr-s form).

6.1 Derived Experience IDs

Experience IDs are derived using the <Experience SSID> form.

<Experience SSID> ::= <Root SSID>[":"<experience name>]

- <Root SSID> refers to the identifier of the parent object from which the Experience is derived. For example, if the Experience is for a movie, the <Root SSID> is the SSID for the movie. If the Experience is for a season, the <Root SSID> is the SSID for the season.
- <experience name> is a string unique for Experience IDs for the <Root SSID>. That is, it needs to be different for each ExperienceID derived from a given <Root SSID>.

As Experiences often align with the top-level identifier used in ALID and ContentID (i.e., <Root SSID>. In this case, the Experience ID is simply constructed using the base identifier as the SSID; for example, `md:experienceid:eidr-s:5EE7-A973-819A-DC1A-CDD8-H`.

Where there is not a simple mapping Experience ID can be constructed by appending unique data; for example, `md:experienceid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:exp1`.

6.2 Derived Presentation IDs

Presentation IDs are derived using the <Presentation SSID> form.

<Presentation SSID> ::= <Presentation Unique ID> | <Root SSID>":"<av type> ["_"<av subtype>] ["."<index>]

- <Presentation Unique ID> and <Component Unique ID> are identifiers that are assigned to the assets externally to this practice. For example, if an EIDR ID is created for a particular Presentation that would be the <Presentation Unique ID>. Similarly, if an EIDR ID is created for a given audio track, that would be the <Component Unique ID>.
- <Root SSID> refers to the identifier of the parent object from which the Presentation is derived. For example, if the Presentation is a trailer for a movie, the <Root SSID>

DRAFT

is the SSID for the movie. It is assumed there is only one "feature" presentation for the given <Root SSID>

- <av type>, <av subtype> and <index> distinguish this asset from other assets
  - <av type> is as defined in Audiovisual/Type.
  - <av subtype> is as defined in Audiovisual/Subtype. This should be included if Audiovisual/Subtype is present.
  - <index> is a number used to differentiate objects of the same type/subtype.
  - For example: `main, promotion_trailer.1, trailer.2, bonus_making-of.1, bonus_making-of.2, bonus_deleted-scenes.1, bonus_deleted-scenes.2`

Following are examples of Presentation IDs using this approach:

- `md:presentationid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:main`
- `md:presentationid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:trailer.1`
- `md:presentationid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:bonus_deleted-scenes.1`

6.3 Derived Component IDs

Componet IDs are derived using the <Component SSID> form.

<Component SSID> ::= <Component Unique ID> | <Presentation SSID> "."<component type> ["."<index>]

<component type> ::= "video" | <audio type> | <subtitle type>

<audio type> ::= "audio."<language>["."<audio type>"]

<subtitle type> ::= "subtitle."<language>["."<subtitle type>"]

- <language> is a language code as defined in [CM]
- <audio type> is as defined in Inventory/Audio/Type. Additional information can be appended as necessary for clarity or uniqueness.
- <subtitle type> is as defined in Inventory/Subtitle/Type. Multiple types or additional information can be appended as necessary for clarity or uniqueness.
- For example, `video, audio.en.primary, audio.fr.narration, audio.es.dialogcentric, audio.de.commentary, subtitle.fr.normal, subtitle.en.forced subtitle.en.sdh, subtitle.de.large, subtitle.es.commentary`

Following are examples of component IDs using this approach:

- `md:vidtrackid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:main.video`

- `md:audtrackid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:main.audio.de.commentary`
- `md:subtrackid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:bonus_trailer.1.subtitle.de.large`

Physical identifiers such as found in Inventory/{Audio|Video|Subtitle}/TrackIdentifier can be represented as follows. Note that physical assets are generally identified as an APID (Physical Asset ID).

- `md:apid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:main.video`
- `md:apid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:main.audio.de.commentary`
- `md:apid:eidr-x:5EE7-A973-819A-DC1A-CDD8-H:bonus_trailer.subtitle.de.large`

DRAFT