Asset Ordering, Delivery and Tracking

CONTENTS

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

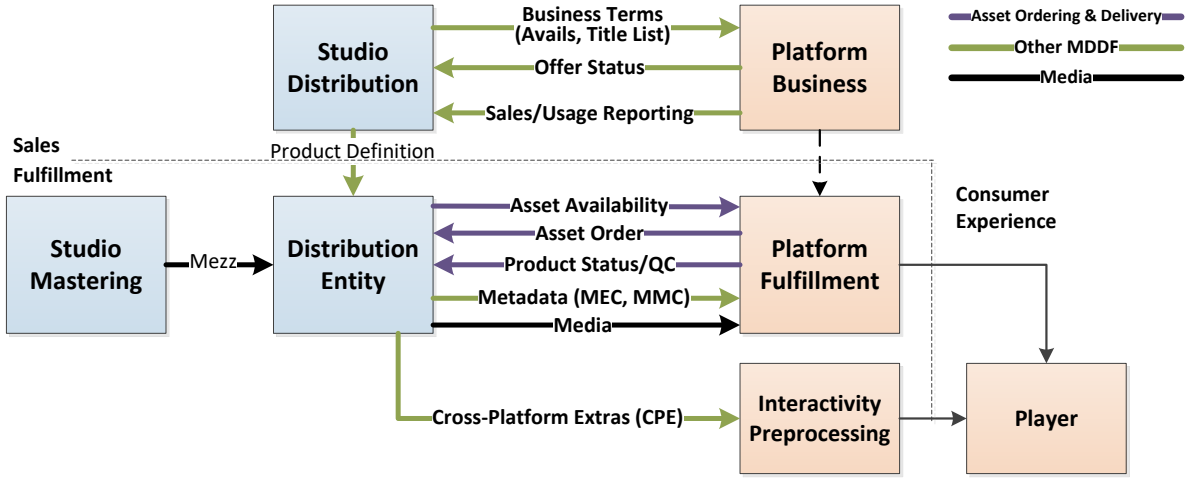
NOTE: No effort is being made by the Motion Picture Laboratories to in any way obligate any market participant to adhere to Common Metadata. Whether to adopt the Common Metadata in whole or in part is left entirely to the individual discretion of individual market participants, using their own independent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

REVISION HISTORY

Version	Date	Description
1.0	December 14, 2019	Original Version
1.1	December 8, 2020	• Added EventTerms to ProductLogEvent-type • Added ProgressDetail to provide additional details on ProgressCode • Added ability to pass detailed metadata in DeliveryAssetReference-type for workflows that otherwise could not connect metadata to the asset • Corrected MDDFID label (spec only) • Added Severity to ErrorDescription to report severity of issues, including whether it has been resolved. • Added capability to capture validator errors

1 INTRODUCTION

This document defined data used in the delivery of assets. The assumed model is the MovieLabs Digital Distribution Framework (MDDF), although it can work with other models as well. The following illustration shows the MDDF flow, with Asset Ordering and Delivery data shown in purple.



This specification is designed to work with other MDDF specifications or with proprietary/legacy specifications.

This document is published in conjunction with Media Delivery Core (MDC) which provides practical examples for using this specification. www.movielabs.com/md/mdc

1.1 Overview

The Asset Ordering and Delivery Process is addressed in three parts

- Rights Management – Generation and delivery of Avails or Title List from a Studio/Asset Provider to the Retailer/Platform and Business-focused status (Offer Status) from the Retailer/Platform
- Asset Planning – All processes (by the Studio/Asset Provider) associated with determining which assets (audio, video, subtitles, artwork, metadata, etc.) will be made available for delivery
- Asset Delivery – Processes associated with the delivery of assets from a Studio/Asset Provider to the Retailer/Platform, and status of assets at the Retailer/Platform

These are illustrated in Figure 1 below.

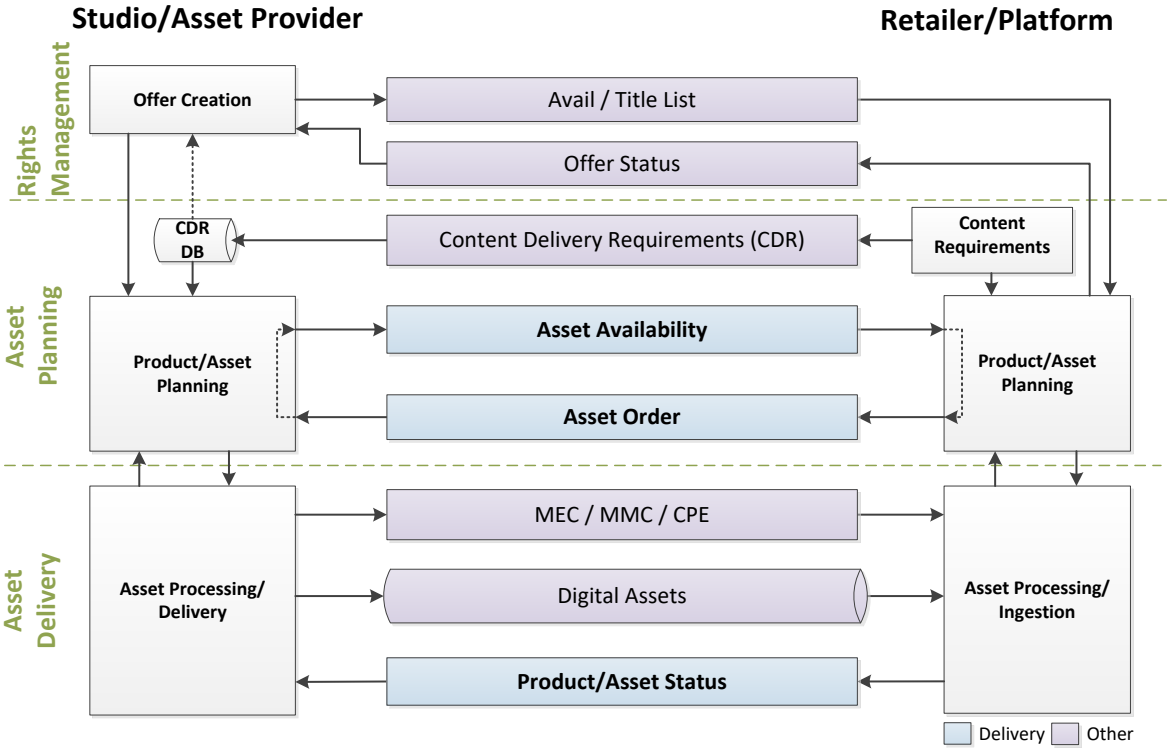
The Rights Management process is covered by *Avails and Title List* and is not further discussed in this document. Offer Status is part of *Avails and Title List*. See www.movielabs.com/md/avails for more information.

Asset Planning determines what assets are to be made available for delivery and when to meet obligations with partners. Asset policies are captured in "Content Delivery Requirements".

Avail or title-specific requests are included in Avail Confirmations, Asset Orders, and Asset Availability.

Asset Delivery has several parts including a Media Manifest Core (MMC) delivery spec, the assets themselves, and Product Status information including both general status of assets and error reporting. MMC is documented elsewhere (www.movielabs.com/md/mmc), and this specification is neutral to assets delivered—we attempt to support almost any format. This specification documents Product Status. Note that although Asset Ordering and Delivery is designed to work with MEC, MMC and CPE, it does not depend on the use of these specs.

Figure 1: Asset Distribution Workflow Composite



1.2 Document Organization

This document is organized as follows:

1. Introduction—Provides background, scope and conventions
2. General Types Encoding
3. Asset Availability
4. Asset Order
5. Product Status

1.3 Document Notation and Conventions

As a general guideline, 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 requirements need not use the formal language above.

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.

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 initiaLowercaseAttribute.

- XML structures are formatted as Courier New, such as `md:id-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 General Notes

All required elements and attributes must be included.

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

UTF-8 [RFC3629] encoding shall be used when ISO/IEC 10646 (Universal Character Set) encoding is required.

1.4 Normative References

[Avails]	Content Availability Metadata, TR-META-AVAIL, http://www.movielabs.com/md/avails
[CM]	Common Metadata, TR-META-CM, http://www.movielabs.com/md/md
[MEC]	Media Entertainment Core, TR-META-MEC, http://www.movielabs.com/md/mec/
[Manifest]	MovieLabs Common Media Manifest Metadata v1.5, TR-META-MMM, www.movielabs.com/md/manifest
[MMC]	Media Manifest Core, TR-META-MMC, www.movielabs.com/md/mmc
[EIDR]	Entertainment Identifier Registry (EIDR), http://eidr.org/technology/
[QCVocab]	Quality Control (QC) Vocabulary, http://www.movielabs.com/md/qcvocabulary
[Ratings]	*Common Metadata Content Ratings*. www.movielabs.com/md/ratings.
[XML]	"XML Schema Part 1: Structures", Henry S. Thompson, David Beech, Murray Maloney, Noah Mendelsohn, W3C Recommendation 28 October 2004, http://www.w3.org/TR/xmlschema-1/ and "XML Schema Part 2: Datatypes", Paul Biron and Ashok Malhotra, W3C Recommendation 28 October 2004, http://www.w3.org/TR/xmlschema-2/

The exact version of the referenced specification is determined by the version of the schemas referenced by the schema associated with this specification. This allows this specification to apply the version of the schema published simultaneously, and through forward compatibility apply to later versions as well.

1.5 Informative References

[CPE]	Cross-Platform Extras (CPE), www.movielabs.com/cpe. See also CPE-Manifest, www.movielabs.com/cpe/manifest

1.6 Status

This specification is ready for implementation. As requirements evolve, we anticipate that the identification of additional use cases will motivate changes. Implementers should anticipate future revisions. Reasonable measures will be taken to ensure changes are backwards compatible.

1.7 Best Practices for Maximum Compatibility

Metadata typically evolves with the addition of new elements, attributes and vocabularies. Existing applications should be capable of accepting metadata, even though there might be more data than expected. Strict XML validation precludes an orderly evolution and can be counterproductive to the flexibility needed in real implementations.

Metadata specifications and schema updates are designed to support backwards compatibility. For example, element and attributes can be added, but required elements are not removed; or more generally ordinality of elements and attributes can be widened but not narrowed. Values are not changed in either syntax or semantics. Therefore, we strongly encourage implementations to either be diligent in tracking to the latest version or follow the backwards compatibility rules provided here.

An XML document is considered compatible if its structure does not preclude the extraction of data from the document. For example, a document with additional elements and attributes do not preclude schema parsing and data extraction.

- Do not reject compatible XML documents, unless they fail schema validation against the definition for an exact version/namespace match.
- Extract data from compatible XML documents whenever possible
- It is allowable to ignore elements and attributes whose presence is not allowed in the specification and schema versions against which the implementation was built. For example, if the original schema allows one instance and three instances are found, the 2nd and 3rd instance may be ignored.

We will try to update metadata definitions such that following these rules work consistently over time. Sometimes, changes must be made that are not always backwards compatible, so we will do our best to note these.

2 GENERAL TYPES ENCODING

This section describes types that are used throughout the Asset Ordering and Delivery Specification, generally in more than one type definition.

The consistent use of these definitions ensures consistency between objects. Usage applies to all uses unless otherwise stated.

2.1 Attribute Groups

2.1.1 LanguageAssets-attr

The LanguageAssets attribute group defines assets associated with a language. It is used both to define rules and to reference assets.

Attribute Group	Definition	Value	Card.
DeliveryLanguageRules-attr			
audio	Audio in this language is required or desired.	xs:string	0..1
video	Video in this language is required or desired. Only applies if there are multiple versions of the video.	xs:string	0..1
timedText	Timed text localization requirements as specified below	xs:string	0..1
SDH	SDH timed text requirements as specified below	xs:string	0..1
descriptive	Descriptive audio is required or desired. See encoding information below.	xs:string	0..1
signed	Video with signing is required or desired. See encoding information below.	xs:string	0..1
metadata	Localized metadata is required or desired. See encoding information below.	xs:string	0..1
subdubPreferred	Indicates that timed text or dub is preferred.	xs:boolean	0..1

The following values apply to all attributes. They are used to indicate the disposition of an asset.

- ‘available’ – The asset is available, or will be within the terms of an agreement, typically before a request
- ‘processing’ – Asset is being processed for delivery (e.g., after a request)
- ‘delivered’ – Asset has been delivered and considered completed unless recipient indicates otherwise

- 'rejected' – Asset has been requested, but will not be delivered
- 'offered' – The asset can be made available (e.g., can be requested or can be ordered) 'available' – Asset is available, but has not been requested

The following values apply specific attributes. They are used to specify requirement for assets.

@audio is encoded as follows:

- 'required' – Localized audio is required. Can be delivered in any format as opposed to 'premium' where premium formats are required. Default for Original.
- 'premium –Localized asset is required in premium format (i.e., multichannel or object-based audio).
- 'desired – Localized audio is desired. It is not a requirement for launch.

@timedText and @SDH are encoded as follows:

- 'required' –Timed text is required.
- 'desired –Timed text is desired. It is not a requirement for launch.
- 'either—Either language or 'SDH' subtitles are required. Both @timedText and @SDH must be encoded 'either'.
- Note: If both language and 'SDH' subtitles are required. Both @timedText and @SDH must be encoded 'required'

@descriptive and @signed is encoded as follows:

- 'required' – Localized asset is required.
- 'preferred' – Localized asset is desired. It is not a requirement for launch.

Note that @timedText generally refers to timed text with Type of 'normal' as defined in [CM], Section 5.2.7.1. @SDH general refers to timed text with type 'SDH' as defined in the same section.

@subdubPreferred is used to indicated that either timed text or audio is required, and which one is preferred. This has precedence over the values in @audio, @timedText and @SDH, but these attributes can still provide useful information. When present, @audio must be either 'required' or 'premium' to indicate whether standard or premium audio is required. When present, at least one of @timedText and @SDH must be 'required' or both @timedText and @SDH must be 'either'. Values such as 'desired' have no meaning because it is redundant with the core meaning of @subdubPreferred which expresses the desire.

- 'sub' – Indicates timed text is preferred
- 'dub' – Indicates audio dub is preferred

Use of @subdubPreferred represents minimum requirements. It is generally acceptable to deliver more, although bilateral agreements could indicate otherwise. Following are some example combinations using @subdubPreferred:

- To indicate a subtitle is preferred, and either a localized timed text, or a SDH timed text is acceptable, encode as follows: @subdubPreferred='sub', @timedText= 'either', @SDH= 'either' (preferred: localized timed text *or* SDH, acceptable: audio)
- To indicate a subtitle is preferred, but both localized timed text and SDH are required, encode: @subdubPreferred='sub', @timedText= 'required', @SDH= 'required' (preferred: localized timed text *and* SDH, acceptable: audio)
- To indicate that audio is preferred, but if timed text is provided either localized timed text or SDH is acceptable, encode: @subdubPreferred='audio', @timedText= 'either', @SDH= 'either' (preferred: audio, acceptable: localized timed text *or* SDH)
- If either audio or any form of timed text (i.e., localized timed text or SDH) is required, @subdubPreferred should not be used

2.2 Date Types

Response Date and LeadTime are used to express when results are expected.

2.2.1 ExpectedDate-type

dateIsTarget indicates that ResponseDate is aspirational. The degree to which this must be honored is subject to bilateral service level agreements.

The object associated with the expectation depends on context. For example, in the context of an asset-related object, the expected date is for the asset.

Element	Attribute	Definition	Value	Card.
ExpectedDate-type		Expected date for delivery or action	md:YearDateOrTime-type	
	dateIsTarget	If 'true' indicates ResponseDate is not a hard deadline. Details determined bilaterally. If 'false' or absent, date is a hard date.	xs:boolean	0..1

2.2.2 LeadTime-type

LeadTime is expressed as a negative duration for deliverables that occur prior to the window (the typical case).

durationIsTarget indicates that LeadTime is aspirational. The degree to which this must be honored is subject to bilateral service level agreements.

Element	Attribute	Definition	Value	Card.
LeadTime-type		Lead time for deliverables relative to window start date. Negative values represent time before window.	xs:duration	
	durationIsTarget	If 'true' LeadTime is a target; that is, not a fixed duration. if 'false' or absent, duration is hard duration.	xs:boolean	

2.3 Message and Terms Types

2.3.1 DeliveryPublisher-type and DeliveryPlatform-type

These fields are provided to allow the recipient of a message to see who it is from, and who it is for; especially, when those parties are ambiguous.

There are up to three parties involved in each transaction: Content Provider/Publisher/Studio, Platform/Retailer and Service Provider. Information might be exchanged between studios and platforms directly (in either direction), or via service providers.

DeliveryPublisher-type and DeliveryPlatform-type provides information about who is sending or receiving information. Whether the Publisher or Platform is the sender or receiver depends on the direction of the message. DeliveryPublisher-type is used to define the Publishers and/or Service Providers acting on behalf of Publishers, whether it is the sender or recipient. DeliveryPlatform-type provides the same data for Platforms and their Service Providers.

A source or destination can have multiple Publisher or Platform instances. This allows a single transaction to apply to a variety of parties. For example, given a company organized around territorial business units (e.g., "Sofaspud Films, US; and Sofaspud Films, EMEA), multiple instances can indicate that this transaction applies to multiple business units.

When service providers are in the transaction, from the standpoint of these interfaces, they are a proxy for another party. For example, a service provider might send information to a platform on behalf of a studio; or, a platform might send information to a service provider for eventual delivery to a studio.

ServiceProvider should only be included when the Service Provider is then sender or recipient of the message. Service Providers are assumed to be single entities, so there is no need for multiple instances.

2.3.2 DeliveryPublisher-type

DeliveryPublisher-type is used to reference a publisher (e.g., studio) and any service provider intermediaries. Generally, when a service provider is included, the XML object is coming from or intended for that service provider.

Element	Attribute	Definition	Value	Card.
DeliveryPublisher-type				
Publisher		Publisher for whom the document was created	md:OrgName-type	0..n
ServiceProvider		Service Provider delivering document	md:OrgName-type	0..1
Contact		Contact information for this document, typically from a Service Provider.	md:ContactInfo-type	0..1

2.3.3 DeliveryPlatform-type

DeliveryPlatform-type is used to reference a platform (e.g., retailer, SVOD service, D2C service) and any service provider intermediaries. Generally, when a service provider is included, the XML object is coming from or intended for that service provider.

Element	Attribute	Definition	Value	Card.
DeliveryPlatform-type				
Platform		Platform/Retailer for whom the document was created	md:OrgName-type	0..n
ServiceProvider		Service Provider delivering document	md:OrgName-type	0..1
Contact		Contact information for this document, typically from a Service Provider.	md:ContactInfo-type	0..1

2.3.4 DeliveryInstructions-type

Delivery instructions provides addition information on how the message is to be handled. These parameters pass information to convey urgency and importance—it is up to the partners to determine how these are handled.

Element	Attribute	Definition	Value	Card.
DeliveryHandling-type				

Comments		Any comments. Should be included if ExceptionFlag='true'	xs:string	0..1
ExceptionFlag		Indicates message requires human attention	xs:boolean	0..1
Priority		Any bilaterally agreed-upon priority system	xs:nonNegativeIntegoer	0..1
ResponseDate		Expected response date	delivery:ExpectedDate-type	0..1 (choice)
LeadTime		Lead time for deliverables relative to window start date. Negative values represent time before window.	delivery:LeadTime-type	

ExceptionsFlag is an indication that something requires human intervention and should be given human attention.

Priority is specified relative to a given DueDate. Note that processing of Priority will require Best Practices that define factors to optimize when prioritizing deliveries of different types across different dates (i.e., factoring in urgency versus importance).

2.3.5 DeliveryScope-type

Delivery Scope allows an object such as an asset order or status report to define the context for the message. That is, defining the scope of the assets for which the object was generated. For example, if the delivery is associated with a particular Avail in France, one would use the ALID and Region to refer to the delivery. If the data is specific to a language or format profile, the Language and FormatProfile elements can be used. TransactionID (same as AvailID in Excel) is an efficient means of referring to a specific Avail over (Transaction element in XML, or row in Excel).

Element	**Attribute**	**Definition**	**Value**	**Card.**	
DeliveryScope-type					
ALID		ALID	md:id-type	0..1	
AlternateID		Alternate ID from Avail	md:ContentIdentifier-type	0..n	
TransactionID		Transaction ID from Avail	md:id-type	0..n	
EIDRURN		EIDR in URN format	md:EIDRURN-type	0..1	
Region		Region(s)	md:Region-type	1..n	0..1 choice
ExcludedRegion		Excluded Region(s)	md:Region-type	0..n	

Language		Language	xs:language	0..n
	asset	Corresponds with LocalizationOffering in Avails [Avails], Section 2.2.2.1 (i.e., 'sub', 'dub', 'subdub', 'any')	xs:string	0..1
FormatProfile		Format Profile as defined in Avails [Avails], Section 2.2.3	xs:string	0..n
	HDR		xs:string	
	WCG		xs:string	
	HFR		xs:string	
	NGAudio		xs:string	

2.3.6 Progress Codes, DeliveryProgressCode-type, ProgressDetail

Progress codes provide general guidance regarding the status of a delivery. Specific information is found in ErrorDescription, when included.

Depending on context, progress codes may refer to specific assets or to multiple assets.

When referring to a single asset, Progress Code values include

- 'Ready' – Asset is ready for use at the Platform. For example, it has been delivered (received) and approved (as applicable). No additional delivery is required.
- 'In-Process' – There is no status to report as asset is being processed
- 'Missing' – Asset is expected but has not been delivered.
- 'Error' – There is an issue with the asset.

Each asset has a state, but when referring multiple assets, the status could be a combination of codes (i.e., some might be *ready*, some might be *in-process*, some might be *missing*, and some might have *errors*). Consequently, Progress Code values for multiple assets are defined as follows:

- 'Ready' – All assets are ready for use at the Platform.
- 'In-Process' – There is no status to report as assets are being processed; with non, so far, with an issue.
- 'Issue' – One or more assets are missing and/or in error. If there are multiple issues, there can be an ErrorDescription instance for each issue.

The reported status of multiple assets (group) is equivalent to the lowest achieved status of any one asset in the group.

For example, if one is reporting on a single asset and it is missing, status would be "Missing". However, if one was reporting on multiple assets and one was missing, status would be "Issue".

2.3.6.1 The DeliveryProgressCode-type

The DeliveryProgressCode-type complex type is used when referring to assets that have some combination of audiovisual media, artwork and metadata. It allows progress to be reported against each.

Element	Attribute	Definition	Value	Card.
DeliveryProgressCode-type		Progress code	xs:string (by extension)	
	media	Progress code for media part	xs:string	0..1
	artwork	Progress code for image/artwork part	xs:string	0..1
	metadata	Progress code for metadata part	xs:string	0..1
	other	Progress code for other parts (e.g., interactive)	xs:string	0..1

2.3.6.2 ProgressDetail

ProgressDetail holds any additional progress information needed for reporting. ProgressDetail should be consistent with ProgressCode.

For example, if ProgressCode is "In-Process", ProgressDetail might be "In-QC". "In-QC" provides additional detail.

2.4 Types that reference objects directly

2.4.1 DeliveryAssetReference-type

DeliveryAssetReference-type provides a means to reference virtually any asset that can be referred to by an identifier. This contrasts with 'referencing by description' (e.g., "French dub") as is defined in ObjectOrderObject-type defined in Section 4.1.1.

Element	Attribute	Definition	Value	Card.
DeliveryAssetReference-type				0..1
TrackReference		TrackReference per [Manifest], Section 2.2.3	xs:string	0..n

TrackIdentifier		TrackIdentifier per [Manifest], Section 2.2.3	md:ContentIdentifier-type	0..n
EIDRURN		EIDR identifier along with scope/structural type	md:EIDRURN-type	0..n
MDDFID		Reference track identifiers as per [Manifest]	delivery:DeliveryMDDFID	0..n
MDDFTrack		Contains references to tracks (as in MDDFID), and also includes track metadata.	manifest:InventorySingleTrack-type	0..n
FileInfo		Reference to a file	manifest:FileInfo-type	0..n
Container		Reference to a container	Manifest:ContainerReference-type	0..n
IMFRef		Reference to information in an Interoperable Master Format (IMF) file.	Delivery:DeliveryIMF-type	0..n
OtherIdentifier		Any other applicable identifier	md:ContentIdentifier-type	0..n

2.4.1.1 DeliveryMDDFID-type

Allows reference via MDDF identifiers. This includes ContentID for metadata and various identifiers used in Media Manifest [Manifest].

The first section is track IDs. Then it gets into other Manifest objects such as Presentations.

When using something other than MDDF, use TrackIdentifier for track, and OtherIdentifier for other objects.

Element	**Attribute**	**Definition**	**Value**	**Card.**
DeliveryMDDFID-type				
AudioTrackID		Audio track ID	manifest:AudioTrackID-type	(choice)
VideoTrackID		Video track ID	manifest:VideoTrackID-type	
SubtitleTrackID		SubtitleTrack ID	manifest:SubtitleTrackID-type	

ImageID		Image ID	manifest:ImageTrackID-type	
InteractiveTrackID		Interactive object (e.g., app) ID	manifest:InteractiveTrackID-type	
ContentID		Content ID. Content ID references metadata, so this ID is used to reference a Common Metadata/MEC metadata object.	md:ContentIID-type	
AncillaryTrackID		Ancillary track ID	manifest:AncillaryTrackID-type	
TextObjectID		Text object ID	manifest:TextObjectTrackID-type	
PresentationID		Presentation ID	manifest:PresentationID-type	
PlayableSequenceID		Playable Sequence ID	manifest:PlayableSequence-type	
PictureGroupID		Picture Group ID	manifest:PictureGroup-type	
AppGroupID		Application Group ID	manifest:AppGroupID-type	
TextGroupID		Text Group ID	manifest:TextGroupID-type	
ExperienceID		Experience ID	manifest:ExperienceID-type	
TimedSequenceID		Timed Sequence ID	manifest:TimedSequenceID-type	
TransactionID		Avails and Title List Transaction ID	md:id-type	
ManifestID		Identifier of a Manifest [Manifest]	md:id-type	

2.4.1.2 DeliveryIMFRef-type

References UUIDs for IMF CPLs, OPLs and virtual tracks.

Element	Attribute	Definition	Value	Card.
DeliveryIMFRef-type			extension of manifest:PresentationIMFRef-type	

VirtualTrackID		Referenced virtual track ID(s)	xs:string	0..n

NOTE: This object may need to be extended to reference other components of an IMF, particularly individual files, sidecars, etc. This specificity might be needed to more granularly request components or to report errors with more specificity.

3 ASSET AVAILABILITY

The Asset Availability describes the status of asset delivery from the Studio/Asset Provider to the Retailer/Platform. This can include assets in any stage of processing and delivery. Some conditions include:

- Assets that have been delivered (retailer perspective on that delivery notwithstanding)
- Assets that are being prepared
- Assets that could potentially be provided by request (perhaps with a fee)

Note that asset status information is sent in both directions. The mirror image of the Asset Availability object is AssetOrder-type sent from the retailer to the studio.

If a platform wishes to inform a publisher that it already has certain assets, perhaps from another source, or perhaps it generated them itself (e.g., subs and dubs); Asset Availability can be sent from Platform to Publisher.

3.1 AssetAvailability-type

AssetAvailability-type describes the state of a localization or track.

Element	Attribute	Definition	Value	Card.
AssetAvailability-type				
	updateNum, workflow, etc.	Workflow attributes	md:Worflow-attr	0..1
Compatibility		Spec compatibility	md:Compatibility-type	
Source		Source of this request	delivery:DeliveryPublisher-type	0..1
Destination		Platform or service provider receiving status	delivery:DeliveryPlatform-type	0..1
AssetAvailabilityID		ID associated with this Asset Availability. Can be used for tracking.	md:id-type	0..1
OrderID		ID(s) for orders addressed by this delivery.	md:id-type	0..n
Description		Description of request	xs:string	0..1

Scope		Information to associate the order with the offer associated with this delivery.	delivery:DeliveryScope-type	0..1
AssetDisposition		Status of asset(s)	delivery:AssetAvailabilityObject-type	1..n
Instructions		Any other instructions	delivery:DeliveryInstructions-type	0..1

3.1.1 AssetAvailabilityObject-type

This complex type contains the disposition of an 'object' which is either assets associated with a language (e.g., French subtitle track) or a specific asset as it would be described in Media Manifest Inventory.

If the Studio/Asset Provider is expressing asset disposition from the perspective of languages, AssetLanguage is used. This construct defines a set of assets associated with a particular language. The status applies to all objects referenced within the AsssetLanguage object. For example, if @audio and @SDH are set, then those assets are being statused. @subdub indicates subtitles and/or dubs are provided at the discretion of the content provider.

Element	Attribute	Definition	Value	Card.	
AssetAvailabilityObject-type					
Language		Language of set of assets for which disposition is provided.	xs:language		choice
	audio, timedText, SDH, etc.		delivery:LanguageAssets-attr	0..1	
	subdub	If true, audio dubs, timed text, or both are provided at the discretion of the content provider. Does not apply to original language.	xs:boolean	0..1	
	OV	Audio is the Original Version (original language)	xs:boolean	0..1	
Track		Describes a single track as it would be described in Media Manifest.	manifest:InventorySingleTrack-type		

StatusCode		Code that indicates status of asset or assets identified in ObjectReference or ObjectDescription	xs:string	
ErrorReference		ErrorReference associated with ErrorDescription in ProductStatus. Associated with rework (i.e., StatusCode='rework').	xs:string	0..n
ExpectedDate		Expected availability or delivery date	md:YearDateOrTime-type	0..1
BusinessTerms		Business terms, such as cost to generate or deliver asset	md:Terms-type	0..n
TechnicalTerms		Additional technical terms relating to asset delivery	md:Terms-type	0..n
Instructions		Any other instructions	delivery:DeliveryInstructions-type	0..1

StatusCode indicates the status of the particular asset. Values include

- 'available' – Asset is available, but has not been requested
- 'offered' – Asset is not available but can be made available upon request; possibly with associated business terms. Note that an asset can, in some workflows, still be requested even if it is not offered.
- 'processing' – Asset is being processed for delivery (e.g., after a request)
- 'delivered' – Asset has been delivered and considered completed unless recipient indicates otherwise
- 'rework' – Same as 'processing' except that it is being reworked as a result of a QC issue
- 'rejected' – Asset has been requested, but will not be delivered
- 'recalled' – Asset has been delivered, but has a problem and should not be used

Language attributes values are defined in LanguageAsset-attr in Section 2.1.1. The value 'available' should be used when the asset is available for delivery. 'offered' should be used when the asset can be made available (e.g., can be requested or can be ordered). When an asset is available to order, one might use business terms to dictate the terms.

If workflows call for more precision, the following may be used

- 'ingested' – Asset has been successfully ingested

- ‘validated’ – Asset has been successfully validated
- ‘abandoned’ – All further actions related to this Asset has been canceled
- ‘superseded’ – Asset has been replaced by a new Asset
- ‘transmitted’ – Asset has been successfully delivered
- ‘deleted’ – The Asset has been deleted from the system
- ‘on-hold’ – Asset is in a hold state
- ‘orphaned’ – The Asset has no associated title

4 ASSET ORDER

An Asset Order represents a request from the Retailer/Platform (or Service Provider) to the Studio/Asset Provider (or Service Provider) for assets and defines objects to be delivered.

4.1 AssetOrder-type

Element	Attribute	Definition	Value	Card.	
AssetOrder-type					
	updateNum, workflow, etc.	Workflow attributes	md:Worflow-attr	0..1	
Compatibility		Spec compatibility	md:Compatibility-type		
Source		Source of this message	delivery:DeliveryPlatform-type	0..1	
Destination		Publisher to whom the status is being sent	delivery:DeliveryPublisher-type	0..1	
OrderID		ID associated with this order. Can be used for tracking.	md:id-type	0..1	
AssetAvailabilityID		ID(s) associated with Asset Availability objects addressed by this Order.	md:id-type	0..n	
Description		Description of request	xs:string	0..1	
Scope		Information to associate the order with the offer associated with this delivery.	delivery:DeliveryScope-type		
Asset		Identifies assets and specifies terms specific to that asset	delivery:AssetOrderObject-type	1..n	choice
RequestCode		Single Request Code that covers entire scope. RequestCode values are defined in Section 4.1.2	xs:string		
TermsAcrossAssets		Specifies terms that apply to all assets identified in the Asset object	delivery:AssetOrderTerms-type	0..n	
Instructions		Any other instructions	delivery:DeliveryInstructions-type	0..1	

4.1.1 AssetOrderObject-type

AssetOrderObject-type specifies the object to be delivered, and possibly terms specific to that object.

Element	Attribute	Definition	Value	Card.	
AssetOrderObject-type			Delivery:AssetOrderTerms-type (by extension)		
Language		Order assets based on language	xs:language		(choice) 1..n
	audio, timedText, SDH, etc.		delivery:LanguageAssets-attr	0..1	
OV		Order assets based on the Original Version (original language).	xs:language		
	audio, timedText, SDH, etc.		delivery:LanguageAssets-attr	0..1	
Description		Reference to objects, such as tracks, by description (e.g., *French dub*).	manifest:Inventory-type		
Reference		Reference to objects such as tracks, requested	delivery:DeliveryAssetReference-type		

4.1.2 AssetOrderTerms-type

AssetOrderTerms-type provides a menu of terms objects that can be used to represent the terms applicable to the order.

Element	Attribute	Definition	Value	Card.
AssetOrderTerms-type				
RequestCode		Code that indicates order status for the object	xs:string	
ExpectedDate		Expected availability or delivery date	delivery:ExpectedDate	0..1 (choice)
LeadTime		Expected date relative to some milestone (e.g., air date)	delivery:LeadTime	

BusinessTerms		Business terms, such as cost to generate or deliver asset	md:Terms-type	0..n
TechnicalTerms		Additional technical terms relating to asset delivery	md:Terms-type	0..n
Instructions		Any other instructions	delivery:DeliveryInstructions-type	0..1

RequestCode indicates how the AssetOrder request should be handled. For example, it could be a request that assets be delivered, it could be a request of estimated date for delivery, and/or it could be a request to price the delivery of assets. RequestCode applies to the entire Scope. For example, if Scope is Region/Country="de", and RequestCode is 'deliver', the request is to deliver everything for Germany.

Values for RequestCode include

- 'deliver' – Deliver asset
- 'redeliver' –There was a problem with delivery and redelivery is requested
- 'cancel' – Asset is not needed. Request can be cancelled.
- 'request' – Asset is not on an AssetAvailability list, but is requested to be delivered.

Note that the main element and Instructions both contain dates and durations. The dates in the element are the data for asset delivery. Dates in Instructions are for the expected response.

5 PRODUCT STATUS

Product Status provides the means for communicating status once there is some agreement on the assets to be delivered.

Depending how ProductStatus is used in the workflow, assets are referenced by description (e.g., "The French dub"), or precisely (e.g., "Track with track ID = …"). The former case usually applies before specific assets are known; either before delivery, or a byproduct of an error (e.g., missing asset). The latter, (i.e., the precise reference with an ID) applies when specific assets are being referenced, such reporting a QC issue with a particular track.

When we use the term "Object", we are describing items abstractly (e.g., "French Dub"). When we say "Asset" we are referring to tangible items (i.e., tracks or files). The descriptive reference is implemented in the ObjectStatus portion of this element. The precise references are implemented in AssetStatus.

A Quality Control (QC) Report is a special case of a ProductStatus object. This report provides the means to identify issues media, metadata and other files. In the simplest form, the QC Report can identify the object in question and the convey associated issue. The QC Report also supports additional data associated with particular media types. For example, timecode ranges can be conveyed for any audio, video and timed text. For uniformity, errors are reported using the standardized QC Vocabulary found in [QCVocab]. What distinguishes a QC report is the presence of ErrorDescription, an object that provides specific information about anomalies associated with deliveries.

5.1 ProductStatus

ProductStatus-type defines the ProductStatus element.

This element provides two means of reporting status, reflected in ObjectStatus and AssetStatus. The primary difference between these elements is AssetStatus reports detailed status (and errors) for objects that exist, while ObjectStatus provides high-level status for objects that either already exist or are expected to exist. As before, we use the term "Object" to refer to something we're describing (i.e., with metadata) and "Asset" to refer to something that can be referenced with an identifier.

Element	Attribute	Definition	Value	Card.
ProductStatus-type				
	updateNum, workflow, etc.	Common set of workflow attributes (defined in Common Metadata)	md:Workflow-attr	
Compatibility		Spec compatibility	md:Compatibility-type	
Source		Source of this message	delivery:DeliveryPlatform-type	0..1

Destination		Publisher to whom the status is being sent	delivery:DeliveryPublisher-type	0..1	
OrderID		ID associated with the order	md:id-type	0..n	
AssetAvailabilityID		ID associated with Asset Availability object	md:id-type	0..n	
Description		Description of status (overview)	xs:string	0..1	
Scope		Information to tie this status to an Avail or other offer	delivery:DeliveryScope-type	0..1	
OverallProgressCode		Overall status progress code(s). This is a rollup of codes when it applies to multiple assets.	delivery:DeliveryProgressCode-type	0..1	
ProgressDetail		Progress detail as defined in Section 2.3.6.	xs:string	0..n	
ObjectStatus		Status of a category of assets referenced descriptively.	delivery:ProductObjectStatus-type	1..n	0..1 choice
AssetStatus		Status of specific assets referenced by identifiers or names.	delivery:ProductAssetStatus-type	1..n	
Instructions		Handling instructions. Includes exception flag.	delivery:DeliveryInstructions-type	0..1	
Log		Event Log	delivery:ProductLog-type	0..1	

5.2 Product Object Status

ProductObjectStatus-type provides status of asset delivery processing.

Element	Attribute	Definition	Value	Card.
ProductObjectStatus-type				
Category		Category of object.	xs:string	0..n
	purpose	Purpose of object within category	xs:string	0..1
Progress		Progress of assets	delivery:ProductProgress-type	0..n
Comments		Any additional comments	xs:string	0..1

Log		Log of previous events	delivery:ProductLog-type	0..1
Instructions		Handling instructions.	delivery:DeliveryInstructions-type	0..1

Category is encoded as follows:

- 'feature' – Object is feature
- 'supplemental' – Object is supplemental material (e.g., Extras/Bonus/VAM)
- 'promotional' – Object is promotional material, typically a trailer
- 'image' – Object is an image, typically artwork

5.2.1 ProductProgress-type

ProductProgress-type defines progress with varying precision.

Element	Attribute	Definition	Value	Card.
ProductProgress -type				
	language	Language associated with progress. If absent, progress applies to all languages	xs:language	0..1
	component	Asset component type. If absent, progress applies to all components. See below	xs:string	0..1
ProgressCode		Associated progress code. See Section 2.3.6	xs:string	
	essential	Is the object essential. 'true' if asset is essential.	xs:boolean	0..1
ProgressDetail		Progress detail as defined in Section 2.3.6.	xs:string	0..n
ExpectedDate		Date when asset is (or was) expected to be delivered	delivery:ExpectedDate-type	0..1
Log		Log of previous events	delivery:ProductLog-type	0..1
Instructions		Handling instructions.	delivery:DeliveryInstructions-type	0..1

@component is defined as follows:

- General categories
  - 'media' – media including audio, video and timed text
  - 'artwork' – Artwork; typically metadata artwork

- ‘other’ – Anything not covered by another category

- Specific categories
  - ‘video’ – video track(s)
  - ‘audio’ – audio track(s). Can be original or dub depending on language.
  - ‘timed text’ – timed text/subtitles
  - ‘descriptive’ – Descriptive audio
  - ‘metadata’ – Metadata

5.3 Product Asset Status

ProductAssetStatus-type provides status of asset ingestion availability by the Retailer/Platform or Service Provider.

Element	Attribute	Definition	Value	Card.
ProductAssetStatus-type				
AssetReference		Reference to Asset	delivery:DeliveryAssetReference-type	1..n
ProgressCode		Progress Code. See Section 2.3.6	xs:string	
	essential	Is the asset essential. ‘true’ if asset is essential.	xs:boolean	0..1
ProgressDetail		Progress detail as defined in Section 2.3.6.	xs:string	0..n
ExpectedDate		Date when asset is (or was) expected to be delivered	delivery:ExpectedDate-type	0..1
ErrorDescription		Description of error associated with progress	delivery:QCErrorDescription-type	0..1
Comments		Any additional comments	xs:string	0..1
Log		Log of previous events	delivery:DeliveryLogEvent-type	0..1
Instructions		Handling instructions. Includes exception flag.	delivery:DeliveryInstructions-type	0..1

5.4 QC-specific Objects

5.4.1 QCErrorDescription-type

QCError-Desription-type provide information about the error as discovered by the Retailer/Platform or Service Provider. ErrorCategory and ErrorTerm are from QC Vocabulary [QCVocab].

In the form of CategorySpecific, details on the specific error can be provided. For example, in anything time-based, start and/or end timecode can be provided. For video pictures or images, a bounding box of the problem area can be described.

In some cases, full QC was not performed on an asset. This can be indicated in FullOrPartialQC.

ErrorReference is included to provide a reference to this specific error report.

Element	Attribute	Definition	Value	Card.
QCErrorDescription-type				
ErrorReference		Reference tag that can be used to refer to this error instance elsewhere	xs:string	0..1
ErrorCategory		Error Category, in accordance with QC Nomenclature [QCVocab]	xs:string	
ErrorTerm		Error Term in accordance with QC Nomenclature [QCVocab]	xs:string	
CategorySpecific		Additional data associated with error, based on Error Category.	delivery:QCCategoryError-type	0..n
Comments		Any additional comments	xs:string	0..1
FullOrPartialQC		Indicates whether assets were fully evaluated or if evaluation stopped at first error(s)	xs:string	0..1
QCReportLocation		Provides the location of a QC report when available online	xs:anyURI	0..1
Severity		Indicates severity of issue.	xs:string	0..1

FullOrPartialQC is encoded as follows:

- ‘Full’ – QC was completed
- ‘Partial’ – QC was aborted once error(s) were found. Additional errors may be present.

Severity is encoded as follows.

- 'critical' – Critical issues are defined as those that block progress or launch.
- 'high', 'medium' and 'low' – The precise meaning of these values must be agreed upon between parties
- 'resolved' – use to report that issue has been resolved

5.4.2 QCCategoryError-type

This section contains additional information for errors that are specific to the type of object with an error. Value depends on the QC Nomenclature Category of the error.

Note that definitions are specific to Error Categories (e.g., Video or Audio), and not to specific Error Terms. It is assumed context is sufficient to interpret term-specific data. If not, Best Practices should be developed and/or notes can be put in the Comments field of the parent object.

Multiple instances can be included. For example, the there is video noise in several segments, a Video instance can be included for each segment.

Element	Attribute	Definition	Value	Card.
DeliveryCategoryError-type				
Audio		Audio Category error specifics	delivery:QCErrorAudio-type	1..n (choice)
Video		Video Category error specifics	delivery:QCErrorVideo-type	
TimedText		TimedText Category error specifics	delivery:QCErrorTimedText-type	
Metadata		Metadata Category error specifics	delivery:QCErrorMetadata-type	
Artwork		Artwork Category error specifics	delivery:QCErrorArtwork-type	
Package		Package Category error specifics	delivery:QCErrorPackage-type	
XML		Any XML object not covered by another category.	delivery:QCErrorXML-type	
Excel		Any Excel object not covered by another category. Specifically intended for Excel Avails.	delivery:QCErrorExcel-type	

5.4.2.1 QC Utility types

5.4.2.1.1 QCTimeRange-type

QCTimeRange-type expresses a period on a audiovisual timeline as expressed by timecode.

Element	Attribute	Definition	Value	Card.
QCTimerange-type				
StartTimecode		Track timeline where issue starts.	manifest:Timecode-type	
EndTimecode		Track timeline where issue ends. Omit, if problem persists to end of timeline or if end is unknown. Should be equal to StartTimecode if problem exists on a single video frame.	manifest:Timecode-type	0..1

5.4.2.1.2 QCXMLError-type

Indicates where in an XML document the problem exists. XPath defines the object. Or, if preferred, a line number can reference the object.

This object can capture errors from the MovieLabs Validator: mddf.movielabs.com. The presumption is that the Validator… fields below are capturing values from this validator. If another validator is used, these fields can be used to capture those values.

Element	Attribute	Definition	Value	Card.
QCXMLError-type				
XPath		XPath reference to object with issue(s)	xs:anyURI	0..1
LineNumber		Line number in file of issue (starts with 1)	xs:positiveInteger	0..1
ValidatorLevel		From Validator: severity (e.g., Error, Warning).	xs:string	0..1
ValidatorTag		From Validator: issue label	xs:string	0..1
ValidatorSummary		From Validator: issue summary	xs:string	0..1
ValidatorFile		From Validator: file with issue	xs:string	0..1

5.4.2.1.3 QCArea-type

QCArea-type defines the area of image or picture area where problem exists.

If issue is a single pixel, Width and Height should be 1.

Element	Attribute	Definition	Value	Card.
QCArea-type				
XOffset		In pixels, x-value of lower left corner of issue region.	xs:integer	
YOffset		In pixels, y-value of lower left corner of issue region.	xs:integer	
Width		In pixels, width of issue region, inclusive of pixel marked by XOffset.	xs:integer	
Height		In pixels, height of issue region, inclusive of pixel marked by YOffset.	xs:integer	

5.4.2.2 QCErrorAudio-type

Audio errors occur during some period of audio (or all of it) and might involve a time offset. QCErrorAudio-type provides the means to express this information.

Element	Attribute	Definition	Value	Card.
QCErrorAudio-type				
TimeRange		Time range where problem exists. If problem is entire range, do not include this element.	delivery:QCTimeRange-type	0..1
TimeOffset		For errors with alignment issues (e.g., AV Sync), the duration of offset. Negative means audio is ahead of video.	xs:duration	0..1

5.4.2.3 QCErrorVideo-type

QCErrorVideo-type provides means to express time and picture area where issue occurs. Time can be as specific as a single frame. Area can be as specific as a single pixel.

Element	Attribute	Definition	Value	Card.
QCErrorVideo-type				
TimeRange		Time range where problem exists. If problem is entire range, do not include this element.	delivery:QCTimeRange-type	0..1

Area		Area picture where problem exists. If the problem covers the entire picture, do not include this element.	delivery:QCArea-type	0..1

5.4.2.4 QCErrorTimedText-type

Expresses the period on the timeline where the problem occurs.

TimeOffset is only used when the error involves an offset, particularly around early or late timed text.

Element	Attribute	Definition	Value	Card.
QCErrorTimedText-type				
TimeRange		Time range where problem exists. If problem is entire range, do not include this element.	delivery:QCTimeRange-type	0..1
TimeOffset		For errors with alignment issues (e.g., subtitle Sync), the duration of offset. Negative means subtitle is ahead of video.	xs:duration	0..1
Text		Text that is in error	xs:string	0..1

5.4.2.5 QCErrorMetadata-type

Errors in metadata are expressed as XML errors. If JSON is being used, XPath does not apply, but LineNumber does.

Element	Attribute	Definition	Value	Card.
QCErrorMetadata-type				
XMLError		Reference to location of XML Error	delivery:QCXMLError-type	

5.4.2.6 QCErrorArtwork-type

QCErrorArtwork-type expresses issues relating to artwork, such as the area in the image where problem exists.

A common issue with images is text. The Text element provides a means to indicate the text in error. Generally, Text should refer to the text in the image rather than the correct text. That is, if there is misspelling include the misspelled text, not the correct text.

Element	Attribute	Definition	Value	Card.
QCErrorArtwork-type				
Area		Area picture where problem exists	delivery:QCArea-type	0..1
Text		Text on image that is in error	xs:string	0..1

5.4.2.7 QCErrorPackage-type

QCErrorPackage-type provides the means to provide additional information about the portion of the package with an issue. Note that the package in question has already been expressed elsewhere in the structure, such as AssetStatus/AssetReference/Container, …/IMFRef or …/FileInfo. This object provides the means to provide additional information about the specific portion of the package (i.e., 'object') with an issue.

Element	Attribute	Definition	Value	Card.
QCErrorPackage-type				
ObjectInError		Object with package with issue	xs:string	1..n
	disposition	Disposition of object	xs:string	0..1

ObjectInError describes the object with an issue. For example, if an audio track is missing, this would be the description of the track.

@disposition indicates the disposition of the object. This needs not be included if context is clear from the error. Examples of @disposition are:

'missing' – indicates the object is missing

'not-expected' – indicates the object was not expected to be part of the package

5.4.2.8 QCErrorExcel-type

Provides the means to express the location of errors in Excel, such as Excel Avails.

Reference can be an entire column (Column), an entire row (Row), a single cell (Cell), or a cell range (Cell and @endCell).

This object can capture errors from the MovieLabs Validator: mddf.movielabs.com. The presumption is that the Validator… fields below are capturing values from this validator. If another validator is used, these fields can be used to capture those values.

Element	Attribute	Definition	Value	Card.

QCErrorExcel-type				
Column		References an entire column	xs:string pattern: [A-Z]+	(choice)
Row		References an entire row	xs:nonNegativeInteger	
Cell		References a single cell or the first cell (upper left) of a range	xs:string pattern: [A-Z]+[0-9]+ *pattern is not enforced by schema*	
	endCell	References the last cell of a range. May not be identical, to the left or above Cell.	pattern: [A-Z]+[0-9]+	
ValidatorLevel		From Validator: severity (e.g., Error, Warning).	xs:string	0..1
ValidatorTag		From Validator: issue label	xs:string	0..1
ValidatorSummary		From Validator: issue summary	xs:string	0..1

5.5 Logs

A log provides a history of events.

5.5.1 ProductLog-type

A log is an ordered sequence of events. Log should be ordered from earliest to latest events.

Element	Attribute	Definition	Value	Card.
ProductLog-type				
Event		A reportable event	delivery:ProductLogEvent-type	1..n

5.5.2 ProductLogEvent-type

ProductLogEvent-type captures a single event for the log.

Element	Attribute	Definition	Value	Card.
ProductLogEvent-type				

EventType		Type of event. Can be a Progress Code.	xs:string	
Timestamp		Time of event. Should be date or date plus time.	md:YearDateOrTime-type	
Description		Description of event	xs:string	0..1
ErrorReference		Reference to a specific error corresponding with an instance of AssetStatus/ErrorDescription/ErrorReference	xs:string	0..n
EventTerm		Additional data for log in name/value pairs	md:Terms-type	0..n

EventType values will be enumerated in Best Practices.