2005-04-08

AAF ASSOCIATION SPECIFICATION

Advanced Authoring Format (AAF) Edit Protocol

Table of Contents

NOTES – The user's attention is called to the possibility that implementation and compliance with this specification may require use of subject matter covered by patent rights. By publication of this specification, no position is taken with respect to the existence or validity of any claim or of any patent rights in connection therewith. The AAFA, including the AAFA Board of Directors, shall not be responsible for identifying patents for which a license may be required by an AAF specification or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

1 Scope

This document defines the Edit Protocol, designed for the storage and exchange of program metadata in AAF files. It defines constraints that shall be applied to the more generalized AAF Specification in order to achieve predictable interoperability between implementations created by different developers.

The Edit Protocol is intended to meet the requirements of audio-visual editorial interchange applications, supporting interchange of metadata that describes edit decisions, audio and visual effects, as well as embedded, non-AAF files.

2 Normative References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this Document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this document are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative documents referred to applies.

IETF RFC 2119 Key words for use in RFCs to Indicate Requirement Levels

AAF Object Specification Version v1.1

IETF RFC 1738 – Uniform Resource Locators (URL)

IETF RFC 2396 – Uniform Resource Identifiers (URI)

SMPTE 320M – Channel Assignments and Levels on Multichannel Audio Media

SMPTE 258M – Transfer of Edit Decision Lists

MIDI Specification Version 1

3 Definition of Terms

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 RFC 2119.

4 Introduction

4.1 Goals

The goal of the Edit Protocol is to provide users with a mechanism for reliable interchange of program metadata, including:

- Edit Decision information, such as source and record time codes, source tracks, physical source names, frame- and sample-accurate clip definitions, clip- and time-based comments, clip names and track names.
- Visual Effects information, such as dissolves, SMPTE wipes, 2D DVE spatial positioning and zooming effects, frame repeat effects, motion and speed change effects, flip and flop effects, as well as composite layering effects.
- Audio Data, such as clip- and track-based gain, stereo pan, fade in and out, symmetrical and asymmetrical crossfades and MIDI data.
- Predictable application behavior, including predictable fallback behavior when an application is not able to process elements of a file.

- (Optional) Embedded media files, such as production audio files, as well as non-AAF files, such as scripts, logs, etc.

4.2 Precepts

The following guidelines have been followed in drafting this specification document:

4.2.1 Simplest Structure

The Edit Protocol defines and promotes the use of the simplest data structure for any given situation within its scope.

4.2.2 Accuracy and Predictability

Interchange of the metadata specified in this document shall be performed in an accurate and predictable manner. Accuracy is defined such that at the end of an interchange, given the capabilities of the importing application, the composition “sounds and looks” as much as possible like the original exported composition. Upon import, an Edit Protocol compliant application shall clearly report any alteration to the composition during the process according to fallback behaviors that are described in this document.

4.2.3 Import/Export Model

The Edit Protocol supports an Import/Export model where metadata is imported from or exported to an AAF file. In this model an application is not required to import, maintain and subsequently export metadata objects from the AAF file that it cannot understand. Applications shall not however replace or represent a subsequent exported file as the originally imported AAF file.

An application shall implement the rules regarding immutability of Mobs as described in the AAF Object Specification v1.1. These rules specify how Mobs shall be identified when they are processed by an application. Specifically, conditions are defined in which the identification of the Mob (MobID) must change, and in which the identification of the Mob must not change. These rules still apply in the import/export model used by the Edit Protocol.

4.2.4 Application Feature Set

The Edit Protocol refers only to interpretation of AAF files into existing feature sets.

The Edit Protocol does not prescribe the required feature set of the target system. For example, an audio workstation does not need to add video capability it does not already have in order to be compliant; nor would a cuts-only editor need to add support for effects, nor would a metadata-only product be expected to perform media relinking.

This does not remove the requirement that importing applications shall log fallback behavior for the benefit of the user as defined in this document.

4.2.5 Non Edit Protocol Objects and Dark Metadata

The Edit Protocol does not address interchange of valid AAF objects defined outside its scope. This does not mean that objects outside of the scope of the Edit Protocol cannot interchange. Inclusion of extraneous data (i.e. non-Edit Protocol Objects and/or Dark Metadata) shall not invalidate, obfuscate or change the meaning of the core Edit Protocol data.

4.2.6 Legacy

The word legacy is used in several places within this document. This word is used to mean “pre-Edit Protocol”. When referring to files, it means that “pre-Edit Protocol” files may exist which do not obey the rules in this protocol document. When referring to applications, it means that “pre-Edit Protocol” applications may exist whose behavior does not meet the rules in this document. Some specific behavior guidelines are given to improve backwards compatibility.

Support for legacy features is not required for compliance with this document.

5 Importing, Fallback handling and logging

The Precepts of section 4.2 recognize that application features sets differ between AAF applications. The goal of importing and fallback handling in the Edit Protocol is to ensure that differences in feature sets are handled gracefully on import to an application, so that the user experiences a work environment where there are “no surprises” and where AAF exchanges behave in a consistent way.

The importing, fallback handling and logging requirements are as follows:

- An application shall not lock-up, crash or give-up on exporting or importing an AAF file. When an action is likely to take a considerable amount of time, an indication of this should be provided.
- Where the feature set of an application is insufficient to completely or accurately import and present the AAF file, an application shall observe the fallback behaviors specified throughout this document (e.g. Section 6.3.1 in Top-Level Composition importing). The fallback behaviors are intended to allow the user to import a useful subset of the AAF file contents into the importing application, up to the limit of its features.
- An application shall provide a mechanism to log diagnostic information for each import or export operation. The log may be temporary, but shall be available to an expert user.
- Certain fallback behaviors shall be logged, as defined in this specification. Wherever appropriate, the diagnostic log should identify the Mob, Track and Position at which the fallback behavior was required.
- Where the feature set of an exporting application supports program metadata that the Edit Protocol specifically disallows (e.g. a program consisting of different audio sample rates which is disallowed in Section 7.3), the application shall not export that metadata to an Edit Protocol AAF file. The omission of any metadata on export in order to comply with the Edit Protocol should be logged.

6 Material metadata

6.1 Overview

The Mobs in an AAF file reference one another to describe a *derivation chain* (or *Mob chain*). The derivation chain describes editing decisions, clips, file source material and physical source material.

The Edit Protocol provides three methods of specifying the source material associated with a clip:

- a clip references file source material stored internally to the AAF file and the physical source material (if any) it was derived from
- a clip references file source material stored externally to the AAF file and physical source material (if any) it was derived from
- a clip references physical source material only

An application shall export each clip using one of these methods.

An application shall import clips that use any of these methods.

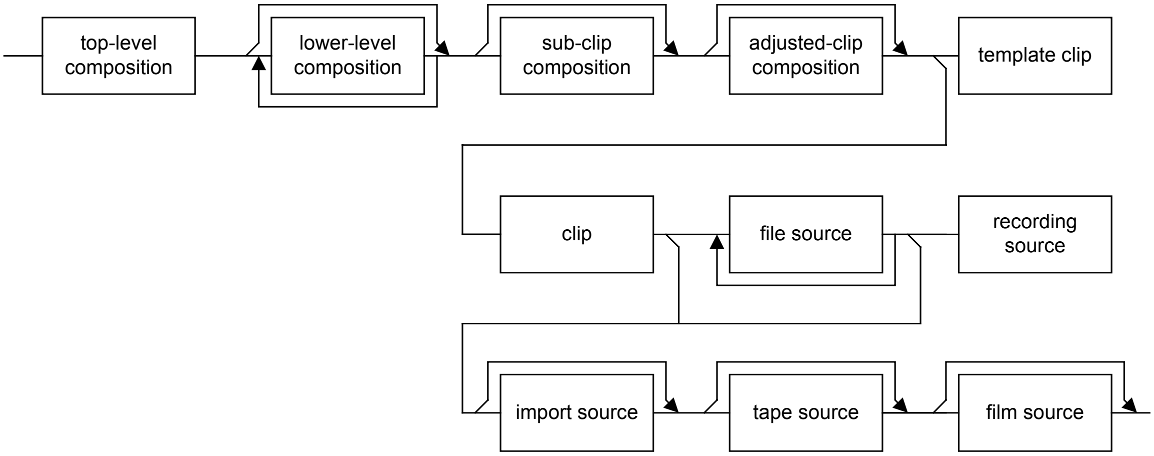
This approach allows applications that export file source material to interoperate with applications that do not. For example, an application that exports file source material is required to gracefully import material that does not contain it; an application that does not export file source material is required to gracefully import material that contains it.

The Edit Protocol uses a number of different kinds of material, as follows:

- top-level composition – specifies a composition that is not referenced by another composition in the file
- lower-level composition – specifies a composition that is referenced by another composition in the file
- sub-clip composition – specifies a sub-section of a clip

- adjusted-clip composition – specifies an adjustment to a clip
- template clip – specifies a clip that has no source material
- clip – specifies a clip that has source material
- file source – specifies file source material whose essence format is described by a sub-class of FileDescriptor
- recording source – specifies recording metadata for file source material without a physical source
- import source – specifies the source of file source material imported by application-specific means
- tape source – specifies tape source material
- film source – specifies film source material

The derivation chain is specified by referencing one material object from another, using SourceClip objects. The possibilities allowed by the Edit Protocol for referencing between Mobs are shown in the figure below.



Referencing between Mobs in the AAF derivation chain

6.2 Derivation chain references

The derivation chain is specified by referencing one Mob from another using SourceClip objects.

An exporting application should include and reference the Mobs for the entire derivation chain to the extent that it is aware of it. The end of the derivation chain (i.e. the original material), to the extent it is aware of it, shall be specified using a zero-value SourceClip object. This indicates that the exporting application did not know of any earlier sources.

If a reference is made to a CompositionMob or MasterMob, that Mob shall be included in the AAF file.

If a reference is made to a SourceMob, that Mob should be included in the AAF file.

Informative note: It is highly desirable to include referenced SourceMobs in the AAF file wherever possible. When an exporting application needs to reference a SourceMob that does not exist at the time, or exists only in an external system, the application may reference a SourceMob that is not included in the AAF file.

6.2.1 Importing application

Where a SourceMob is referenced but not included in the AAF file, the importing application shall attempt to resolve the reference to the SourceMobs that it was previously aware of.

Where a CompositionMob uses a SourceClip object to reference a TimelineMobSlot, the importing application shall treat any references beyond the extent of the TimelineMobSlot (before the beginning or after the end) as if they had referenced Filler, i.e. present as black picture or silence audio. This situation may occur within a Two-Parameter Audio Dissolve effect.

For SourceMob references that remain unresolved, the importing application shall search the part of the derivation chain that is available to find a source it may support. If no supported source is found, some placeholder essence shall be substituted.

6.3 Top-level Composition

A top-level composition shall be specified using a CompositionMob with Mob::UsageCode property equal to Usage_TopLevel. A top-level composition shall not be referenced by another Mob in the AAF file.

A top-level composition shall only reference the following kinds of material:

- a lower-level composition
- a sub-clip composition
- an adjusted-clip composition
- a template clip
- a clip

The top-level composition Mob::Name property shall be a valid name identifying the composition. Where multiple compositions are exported, each shall have a different name.

A top-level composition shall contain one or more timecode tracks and include a Primary timecode track. Timecode tracks shall use the MobSlot::PhysicalTrackNumber property to distinguish their type, as shown in the Table below.

Informative note: Multiple timecode tracks are typically present when working with mixed frame rate sources.

Physical Track Number	Time Code Type
1	Primary timecode
2	Alternative timecode, e.g. film
3-6	Reserved

PhysicalTrackNumber values for timecode tracks in a CompositionMob

6.3.1 Fallback behavior

If the importing application does not support the required number of top-level compositions, a subset shall be imported into the application. This fallback behavior shall be logged. The user should be able to select which top-level compositions are imported from a list of their Mob::Names.

6.4 Lower-level Composition

A lower-level composition shall be specified using a CompositionMob with Mob::UsageCode property equal to Usage_LowerLevel. A lower-level composition shall be referenced by a top-level or lower-level composition in the AAF file.

A lower-level composition shall only reference the following kinds of material:

- a sub-clip composition
- an adjusted-clip composition
- a template clip
- a clip

The lower-level composition Mob::Name property shall be a valid name identifying the composition. Where multiple compositions are exported, each shall have a different name.

6.4.1 Fallback behavior

If the importing application does not support the required depth of references to lower-level compositions, the application shall replace the references to lower-level compositions with equivalent references to clips, i.e. the linkage from the top-level composition to the clip shall not be lost. This fallback behavior shall be logged.

6.5 Sub-clip Composition

Some applications support the notion of a sub-clip in which a section of a clip is identified and thereafter treated in the user interface as an additional clip. This notion is typically used to allow the user to sub-divide an existing long clip into shorter sections for convenience.

A sub-clip composition shall be specified using a CompositionMob with Mob::UsageCode property equal to Usage_SubClip.

Legacy note: An alternative representation of a sub-clip composition is prevalent in legacy files. This representation is a CompositionMob with Mob::AppCode property equal to 2. An importing application may treat this as a sub-clip composition. Any other value of Mob::AppCode property in a CompositionMob may be ignored and the CompositionMob treated as a lower-level composition.

A sub-clip composition shall only reference the following kinds of material:

- an adjusted-clip composition
- a template clip
- a clip

Each essence track of a sub-clip composition shall contain exactly one SourceClip.

The sub-clip Mob::Name property shall be a valid name identifying the sub-clip.

6.5.1 Importing application

Importing applications shall classify incoming Mobs as follows.

A CompositionMob with Mob::UsageCode property equal to Usage_SubClip shall be treated as a sub-clip composition. Any other value of Mob::UsageCode property shall be retained; however the Mob shall be treated as a lower-level composition.

A sub-clip composition that does not comply with the restrictions for exporting sub-clips should be treated as a lower-level composition.

6.5.2 Fallback behavior

If the importing application does not support sub-clips, the Mob shall be treated as a lower-level composition.

6.6 Adjusted-clip Composition

Some applications support the notion of an adjusted-clip in which an effect is applied directly to a clip and applies to all uses of that clip, e.g. an audio gain effect.

An adjusted-clip composition shall be specified using a CompositionMob with Mob::UsageCode property equal to Usage_AdjustedClip.

Legacy note: An alternative representation of an adjusted-clip composition is prevalent in legacy files. This representation is a MasterMob containing an OperationGroup. An importing application may treat this as an adjusted clip composition.

An adjusted-clip composition shall only reference the following kinds of material:

- a template clip
- a clip

Each essence track of an adjusted-clip composition shall contain exactly one OperationGroup.

The adjusted-clip Mob::Name property shall be a valid name identifying the adjusted-clip.

6.6.1 Fallback behavior

If the importing application does not support adjusted-clips, the Mob shall be treated as a lower-level composition.

6.7 Template Clip

Some applications support the notion of a composition in which “template” clips are referenced which initially reference no sources. Later, the clips may be updated to reference particular sources. This allows a composition to be created before the source material is available.

Informative note: A composition that references template clips is subtly different to a composition that references no clips. The former contains metadata about the clip’s intended grouping of tracks and the timing relationships between them, while the latter does not.

A template clip shall be specified using a MasterMob with Mob::UsageCode property equal to Usage_Template.

Each essence track of a template clip shall contain exactly one SourceClip. The SourceClip shall have zero-value to denote the end of the known derivation chain.

If all the template clip essence tracks are updated to refer to particular sources, then the template clip shall be converted to a clip.

The template clip Mob::Name property shall be a valid name identifying the template clip.

6.7.1 Fallback behavior

If the importing application does not support template clips, the Mob shall be treated as a clip with some placeholder essence in it. This fallback behavior shall be logged.

6.8 Clip

A clip shall be specified using a MasterMob with no Mob::UsageCode property.

A clip shall only reference the following kinds of material:

- a file source
- an import source
- a tape source
- a film source

A MasterMob shall not reference more than one essence track of a particular file source from each time position in the MasterMob.

The clip Mob::Name property shall be a valid name identifying the clip.

6.9 File Source

A file source shall be specified using a SourceMob with an EssenceDescriptor that is a sub-class of FileDescriptor. A file source is also known as a file SourceMob.

Informative note: A file source will typically have one track, to describe a single track of essence. The AAF Edit Protocol does not limit the number of tracks in a file SourceMob however. Instead it makes the constraint that a MasterMob shall not reference more than one essence track of a particular file source from each time position in the MasterMob.

The essence associated with a file SourceMob shall either be internal to the AAF file in an EssenceData object or external to the AAF file. In either case, the file SoureMob shall have a FileDescriptor::ContainerDefinition property specifying the file kind of the essence container.

Where essence is external to the AAF file, the EssenceDescriptor::Locator property should contain a Locator object specifying the location of the essence.

Informative note: It is highly desirable for an exporting application to include a Locator specifying the location of the essence. However, if this information is not available to an exporting application, it is a valid use-case for sources to be identified by MobID only, and for the importing application to rely upon an asset manager to locate the essence.

The EssenceDescriptor::Locator property should include at least one NetworkLocator that complies with the following constraints:

- The NetworkLocator has a URI that complies with RFC 2396.
- It is an absolute or a relative URI.
- If it is an absolute URI, it conforms to the RFC1738 file URI scheme.
- If it is a relative URI, the base URI is determined from the URI of the AAF file itself.

When exporting an AAF file containing relative URIs, the exporting application should also export the target resources. If this is not possible, absolute URIs should be substituted or appended as additional NetworkLocators.

If a file source has any of the following sources, then those source mobs should be included and referenced by the file source:

- an import source
- a tape source
- a film source

If a file source does not have one of the above sources, then it shall reference and include a recording source.

6.9.1 Importing application

An importing application may use the EssenceData object in the AAF file if it is present or may use the external essence located by the NetworkLocator.

Locators are hints, not authoritative references. Importing applications should verify that URIs can be successfully resolved before attempting to use them.

6.9.2 Fallback behavior

If the importing application does not support file sources or the particular file source format or cannot locate the file source essence, it shall search along the source derivation chain to find a source it may support. If no supported source is found, some placeholder essence shall be substituted.

If file source importing fails because the importing application cannot locate the file source essence, the file source MobID and searched paths shall be logged.

If file source importing fails because the reading or parsing of the file source essence was unsuccessful, then the application should import as much of the essence as possible and pad any remaining duration with placeholder essence. This fallback shall be logged.

6.10 Recording Source

When no physical source exists for file source material, such as in the case of live recordings, a recording source may be used to represent the source. A recording source is analogous to a tape source except that it does not represent a source that physically existed. It is used to provide a timecode reference to file source material.

A recording source shall be specified using a SourceMob with a RecordingDescriptor. A recording source is also known as a recording SourceMob.

Each essence track of a recording source shall contain exactly one SourceClip. The SourceClip shall have zero-value to denote the end of the known derivation chain.

The recording source Mob::Name property shall be a valid name identifying the source, e.g. a name describing a recording session.

6.11 Import Source

An import source represents the source of file source material imported by application-specific means.

An import source shall be specified using a SourceMob with an ImportDescriptor. An import source is also known as an import SourceMob.

The EssenceDescriptor::Locator property shall include at least one NetworkLocator that complies with the constraints set out in Section 6.9.

If an import source has any of the following sources, then those source mobs should be included and referenced by the import source:

- a tape source
- a film source

Legacy note: An alternative representation of an import source is prevalent in legacy files. This representation is a SourceMob with an Avid Source File essence descriptor. An importing application may treat this as an import source.

6.11.1 Fallback behavior

If the importing application does not support import sources or the particular import source format as determined by inspection of the file or cannot locate the file source essence, it shall search along the source derivation chain to find a source it may support. If no supported source is found, some placeholder essence shall be substituted.

If import source importing fails because the importing application cannot locate the file source essence, the file source MobID and searched paths shall be logged.

If import source importing fails because the reading or parsing of the file source essence was unsuccessful, then the application should import as much of the essence as possible and pad any remaining duration with placeholder essence. This fallback shall be logged.

6.12 Tape Source

A tape source shall be specified using a SourceMob with a TapeDescriptor. A tape source is also known as a tape SourceMob.

The tape source Mob::Name property shall be a valid name identifying the tape name.

A tape SourceMob may contain one or more timecode tracks. Timecode tracks shall use the MobSlot::PhysicalTrackNumber property to distinguish their type, as shown in the Table below.

Informative note: Multiple timecode tracks are typically present when working with mixed frame rate sources.

Physical Track Number	Time Code Type
1	Primary timecode
2	Reserved
3	Aux1
4	Aux2
5	Aux3
6	Aux4
7	Aux5
8-12	Reserved

PhysicalTrackNumber values for timecode MobSlots in a tape SourceMob

If a tape source has a film source, then it should be included and referenced by the tape source.

6.12.1 Fallback behavior

If the importing application does not support tape sources or the particular tape source, it shall search along the source derivation chain to find a source it may support. If no supported source is found, some placeholder essence shall be substituted.

6.13 Film Source

A film source shall be specified using a SourceMob with a FilmDescriptor. A film source is also known as a film SourceMob.

The film source Mob::Name property shall be a valid name identifying the film reel name.

A film SourceMob may contain one or more edgecode tracks. Edgecode tracks shall use the MobSlot::PhysicalTrackNumber property to distinguish their type, as shown in the Table below.

Physical Track Number	Edge Code Type
1	Keycode Number
2	Ink Number
3	Aux. Ink Number

PhysicalTrackNumber values for edgecode MobSlots in a film SourceMob

6.13.1 Fallback behavior

If the importing application does not support film sources, it shall search along the source derivation chain to find a source it may support. If no supported source is found, some placeholder essence shall be substituted.

7 Track metadata

7.1 Edit rate values and edit point position

The value used for the edit rate of an essence track depends upon:

- whether the track is video or audio
- the nominal sample rate
- whether the application supports sample-accurate audio edits
  - If an application does not support sample-accurate audio editing, then it may use video frame edit rates on audio tracks.
- whether the audio rate has been pulled-up or pulled-down
  - The edit rate should indicate the playback speed of the essence.

The edit rate value shall be equivalent to the values in the following Table. An equivalent value is a rational number that is numerically equivalent to a rational or rounded value listed in the Table. A rational with a zero or negative numerator or denominator shall not be used.

Nominal	Track Type	59.94i Pull-Down		59.94i Pull-Up		50i Pull-Down		50i Pull-Up	
		Rational	Rounded	Rational	Rounded	Rational	Rounded	Rational	Rounded
24	A or V	24000/1001	23.976	24024/1000	24.024	576/25	23.040	600/24	25.000
25	A or V	25000/1001	24.975	25025/1000	25.025	600/25	24.000	625/24	26.042
30	A or V	30000/1001	29.970	30030/1000	30.030	720/25	28.800	750/24	31.250
50	A or V	50000/1001	49.950	50050/1000	50.050	1200/25	48.000	1250/24	52.083
60	A or V	60000/1001	59.940	60060/1000	60.060	1440/25	57.600	1500/24	62.500
44100	A	44100000/1001	44056	44144100/1000	44144	1058400/25	42336	1102500/24	45938
48000	A	48000000/1001	47952	48048000/1000	48048	1152000/25	46080	1200000/24	50000
88200	A	88200000/1001	88112	88288200/1000	88288	2116800/25	84672	2205000/24	91875
96000	A	96000000/1001	95904	96096000/1000	96096	2304000/25	92160	2400000/24	100000
176400	A	176400000/1001	176224	176576400/1000	176576	4233600/25	169344	4410000/24	183750
192000	A	192000000/1001	191808	192192000/1000	192192	4608000/25	184320	4800000/24	200000

Allowed Edit Rate values

7.1.1 Exporting Application

An edit point position may be at any edit unit boundary, as defined by the edit rate value.

7.1.2 Importing Application

An importing application shall accept the edit rate values defined in Section 7.1 and corresponding edit point positions.

7.1.3 Fallback behavior

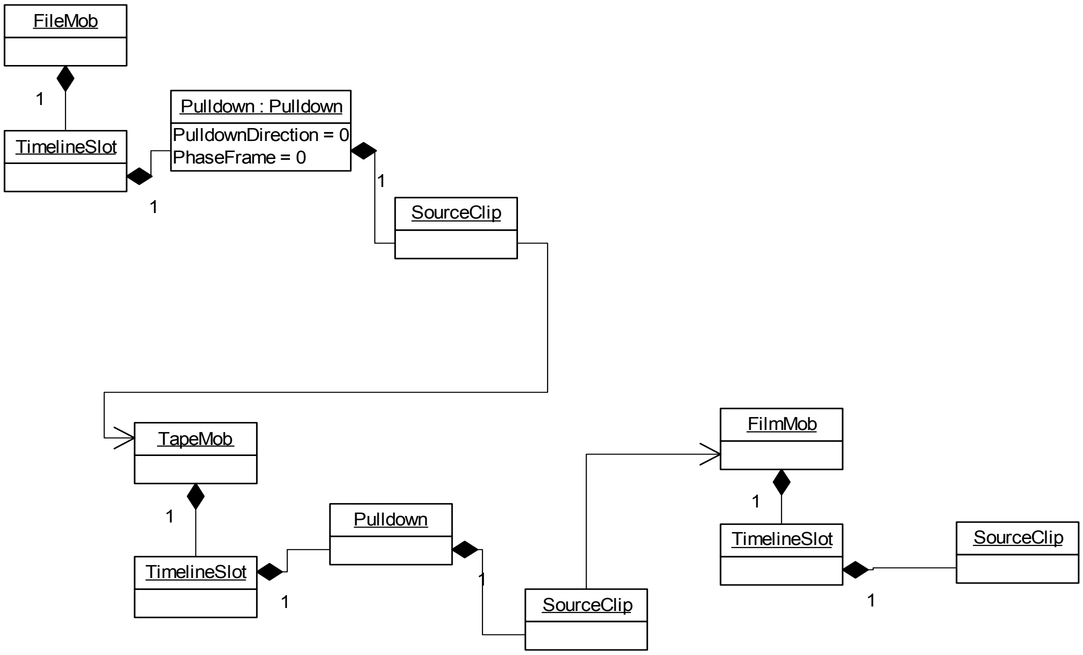
If the importing application does not support edit points at the specified position, it shall round the position to the nearest position that can be supported. Any rounding shall avoid introducing gaps where none existed before and minimize changes to the overall duration. This fallback behavior shall be logged.

7.2 Pull up and pull down

Pull up and pull down shall be specified using a Pulldown object. A Pulldown object may be used in the following contexts:

- In a TimelineMobSlot of a file SourceMob, where the Pulldown object contains a SourceClip referencing a tape SourceMob (e.g. a file SourceMob at 24 Hz referencing a tape SourceMob of 30 Hz)
- In a TimelineMobSlot of a tape SourceMob, where the Pulldown object contains a SourceClip referencing a film SourceMob (e.g. a tape SourceMob at 30 Hz referencing a film SourceMob of 24 Hz)
- In a TimelineMobSlot of a tape SourceMob, where the Pulldown object contains a SourceClip referencing a tape SourceMob (e.g. a tape SourceMob at 30 Hz referencing a tape SourceMob of 24 Hz)

The following diagram illustrates the first two cases listed above:



7.2.1 Importing application

An importing application shall accept Pulldown objects in the contexts listed above.

7.3 Audio edit rate and sample rate

All audio tracks within a file shall have the same edit rate and sample rate.

7.4 Video edit rate and sample rate

The video edit rate shall be less than or equal to the video sample rate.

7.5 Number of essence tracks

The Edit Protocol does not limit the number of essence tracks in a Mob.

In a CompositionMob, multiple essence tracks may be exported as multiple MobSlots or held within a NestedScope. When essence tracks are exported as multiple MobSlots, the tracks are synchronized but independent. When essence tracks are held within a NestedScope, the tracks are synchronized and a mapping to a single output track is specified.

The MobSlot::Name property shall be a valid name identifying the essence track, e.g. V1, A1, A2, TC1. Where multiple essence tracks are exported, each shall have a different Name.

7.5.1 Fallback behavior

If the importing application does not support the required number of essence tracks of a particular kind, a subset shall be imported into the application. This fallback behavior shall be logged. The user should be able to select which essence tracks are imported from a list of their MobSlot::Names.

7.6 Track mapping

The MobSlot::PhysicalTrackNumber property shall be specified for essence MobSlots within the following kinds of Mob:

- CompositionMob
- MasterMob

- recording SourceMob
- import SourceMob
- tape SourceMob
- film SourceMob

In a CompositionMob or MasterMob, PhysicalTrackNumber is the output channel number that the MobSlot should be routed to when played. In a SourceMob, PhysicalTrackNumber is the track number within the source that the MobSlot is describing.

Informative note: Typically, for each kind of essence data definition within a Mob, PhysicalTrackNumber starts at 1 and counts up. For audio tracks, the value of PhysicalTrackNumber may be repeated by more than one track.

For audio tracks belonging to a stereo pair, PhysicalTrackNumber=1 shall mean left, PhysicalTrackNumber=2 shall mean right.

Any SourceReference between two MobSlots with PhysicalTrackNumbers implicitly defines a track mapping.

7.6.1 Fallback behavior

If the importing application does not support the specified PhysicalTrackNumber on a track it is importing, then it shall fallback to a track number that can be supported. This fallback shall be logged. The user should be able to control the mapping of the PhysicalTrackNumber values in the AAF file into the application.

7.7 Mark points

In some applications, the most recently set Mark IN and Mark OUT points that were set by the user on a clip or composition may be preserved. In addition, the position in the clip or composition that was most recently presented to the user may be preserved.

The marked IN and OUT points and user position shall be specified using the TimelineMobSlot::MarkIn, TimelineMobSlot::MarkOut and TimelineMobSlot::UserPos optional properties respectively.

7.7.1 Importing Application

An importing application may assume that the mark points also apply to other tracks for which no marks are specified.

7.8 Multi-channel audio

Within a CompositionMob or MasterMob, multi-channel audio (e.g. stereo audio or 5.1 surround) shall be represented using multiple tracks with mono sound Data Definition. Within a CompositionMob or MasterMob, a multi-channel sound Data Definition shall not be used.

The intended placement of each audio track when played shall be indicated using the MobSlot::PhysicalTrackNumber property. The Physical Track Number shall be an integer in the range 1 to the total number of audio tracks in the program. Six channel surround tracks shall be numbered in accordance with SMPTE 320M Standard Assignment B, as shown below:

Physical Track Number	Position
1	Left
2	Right
3	Center
4	Low Frequency Effects
5	Left Surround
6	RightSurround

PhysicalTrackNumber values for six channel surround sound tracks

Within a SourceMob, multi-channel audio shall be represented in either of the following ways:

- a single track with a multi-channel sound Data Definition and a single audio EssenceDescriptor object for multiple channels
- multiple tracks with a mono sound Data Definition and a MultipleDescriptor containing the same number of mono audio EssenceDescriptors

To reference a single channel of a multi-channel track from a mono track, the SourceReference::ChannelIDs property shall be used with a single element in the array.

To reference multiple channels of a multi-channel track from a multi-channel track, the SourceReference::ChannelIDs property shall be used with multiple elements in the array.

To reference multiple mono tracks from a multi-channel track, the SourceReference::MonoSourceSlotIDs shall be used with multiple elements in the array.

7.8.1 Exporting Application

The exporting application may export multi-channel essence as a single file SourceMob. However, where the multi-channel essence requires a codec that is not widely available, the exporting application should also export the individual channels as separate essence and file SourceMobs, linked into the Mob reference chain.

7.8.2 Importing Application

The importing application shall accept multi-channel audio SourceMobs represented in either of the ways described in Section 7.7.

The application shall accept the SourceReference::ChannelIDs and SourceReference::MonoSourceSlotIDs properties in SourceMobs.

The application shall accept the SourceReference::ChannelIDs property (with a single element in the array) in a MasterMob.

7.8.3 Fallback behavior

If the importing application does not support multi-channel audio sources or the particular multi-channel audio file source format, it shall search along the source derivation chain to find a source it may support. If no supported source is found, some placeholder essence shall be substituted.

7.9 Clips with multiple file source choices

In some applications, a clip is available with multiple file source choices, e.g. a choice of essence type or resolution for the same clip. A clip with multiple file source choices shall be specified using an EssenceGroup object in a MasterMob.

7.9.1 Importing application

An importing application shall support EssenceGroup objects in a MasterMob. Importing applications should select one of the Choices in the EssenceGroup based upon a defined criterion. The criterion may be any of the following:

- an application-defined essence type
- an application-defined essence quality measure
- a user-defined essence type
- a user-defined essence quality measure
- a user selection

7.9.2 Fallback behavior

If an importing application cannot support application or user-defined criteria, it shall by default select the first entry in the Choices property

If an application finds no Choice which matches a defined criterion, it may select an acceptable alternative or substitute placeholder essence. This fallback behavior shall be logged.

7.10 Gaps

In some applications, a gap in the timeline is used to position later sections when not all of the preceding material is specified and to fill time in tracks or layers that is not referenced or played. A gap, where the output is unspecified for a defined duration, shall be specified using a Filler object.

7.10.1 Importing application

An importing application shall accept Filler objects. If Filler is played, it shall be presented as a black picture and silent audio.

Informative note: A Filler object may be used as the input to an effect, e.g. a fade to black effect is implemented as a dissolve to filler. It is for this reason that the presentation of Filler is defined.

7.11 Timecode and edgecode

A track of edgecode shall be specified using one or more Edgecode objects.

A track of timecode shall be specified using one or more Timecode objects or a TimecodeStream12M object.

An Edgecode object specifies a continuous section of film edgecode.

A Timecode object specifies a continuous section of timecode and a TimecodeStream12M object contains a stream of SMPTE 12M Timecode.

To specify a discontinuity in edgecode or timecode, multiple Edgecode objects or Timecode objects shall be placed in a Sequence – the discontinuity occurs at the junction between Edgecode objects or Timecode objects.

A Primary timecode track (see Section 6.3) in a top-level or lower-level composition shall consist of a single Timecode object.

Source timecode may be specified in any SourceMob.

7.11.1 Importing application

An importing application shall accept Edgecode, Timecode and TimecodeStream12M objects and shall support discontinuities in Timecode objects and Edgecode objects within SourceMobs.

8 Composition mixdown clip (Optional)

In some applications, a set of tracks within a composition are mixed-down and rendered into a clip for use as a “guide track” for subsequent post-production operations.

A composition mixdown clip shall be specified as a clip (as specified in Section 6.8) referencing file sources (as specified in Section 6.9). The beginning of the composition mixdown clip and the beginning of the composition are assumed to be co-timed.

The composition may reference the mixdown clip using the CompositionMob::Rendering property.

9 Auxiliary files (Optional)

9.1 Referencing and embedding auxiliary files

In some applications, a user may need to reference and embed auxiliary files within an AAF Edit Protocol file. For example, a word processor document (such as a script) or standard MIDI file (such as music parts) may appropriately accompany an AAF composition, even though the data contained within the embedded file does not map neatly into the AAF object model.

9.1.1 Exporting application

Each auxiliary file EssenceData object shall be specified using a SourceMob with an AuxiliaryDescriptor. A SourceMob with an AuxiliaryDescriptor is also known as an auxiliary SourceMob.

An auxiliary SourceMob shall contain at least one Slot, which may be a TimelineMobSlot, EventMobSlot or StaticMobSlot as appropriate. The Data Definition of the Segment within the Slot shall be DataDef_Auxiliary.

An AAF Mob (such as an AAF composition) may reference an auxiliary SourceMob using a SourceReference (or sub-class thereof) from a DataDef_Auxiliary Slot in the referencing Mob to the DataDef_Auxiliary Slot in the auxiliary SourceMob.

An embedded auxiliary file shall be specified using an EssenceData object containing the auxiliary file data. The EssenceData object is linked to the auxiliary SourceMob using a common MobID.

9.1.2 Importing application

The importing application may attempt to recognize the type of auxiliary data from the MIME type in the AuxiliaryDescriptor.

9.1.3 Fallback behavior

If the importing application does not support auxiliary data file references, or does not support the particular MIME type of the auxiliary data file, then the auxiliary Slots may be ignored.

9.2 Referencing and embedding MIDI files

A standard MIDI file may be referenced from a composition and embedded in an AAF file as an auxiliary file, as described in Section 9.

No more than one MIDI auxiliary file shall be referenced from a given composition.

The MIDI file shall conform to the standard MIDI specification version 1.

The AuxiliaryDescriptor::MimeType property shall be “audio/midi”.

The beginning of the composition and the beginning of the MIDI file are assumed to be co-timed. The importing application shall parse the MIDI file, including tempo maps, to determine synchronization at further points in the composition.

10 Annotations

10.1 Annotation using TaggedValue or KLVData objects

Annotations are divided into:

- User Comments – which are directly classified and set up by the operator (for example, Bin columns)
- Attributes – which are under the control of the application itself (for example, filter control)
- KLVData – other ancillary data from other devices, or extracted from essence or elsewhere, encoded as KLV

Annotations may be represented using arrays of TaggedValues or KLVData in the following optional properties:

- Mob::UserComments (StrongReferenceVector of TaggedValue)
- Mob::Attributes (StrongReferenceVector of TaggedValue)
- Mob::KLVData (StrongReferenceVector of KLVData)
- Component::UserComments (StrongReferenceVector of TaggedValue)
- Component::Attributes (StrongReferenceVector of TaggedValue)
- Component::KLVData (StrongReferenceVector of KLVData)

Component::UserComments shall only be used on:

- the Segment of a MobSlot
- a CommentMarker (or its subclasses, such as DescriptiveMarker)

The Data Definition for non-essence tracks containing annotations shall be DataDef_DescriptiveMetadata.

CommentMarkers may be used for annotations that are not associated with a specific essence track; otherwise DescriptiveMarkers (or its subclasses) shall be used.

CommentMarkers shall be placed only in EventMobSlots. DescriptiveMarkers may be placed in any kind of MobSlot (e.g. TimelineMobSlot, EventMobSlot or StaticMobSlot).

Exporting applications shall document all TaggedValues in use using TaggedValueDefinition objects in the Dictionary::TaggedValueDefinitions property.

Exporting applications shall document all KLVData in use in KLVDataDefinition objects in the Dictionary::KLVDefinitions property.

The use of the Event::Comment property is deprecated, in favor of the Component::UserComments property with a TaggedValue named Comment. The use of the Event::Comment property is optional.

10.1.1 Importing applications

The importing application shall preserve declared TaggedValues and declared KLVData in the following properties for later export:

- Mob::UserComments (StrongReferenceVector of TaggedValue)
- Mob::Attributes (StrongReferenceVector of TaggedValue)
- Mob::KLVData (StrongReferenceVector of KLVData)
- Component::UserComments (StrongReferenceVector of TaggedValue)
- Component::Attributes (StrongReferenceVector of TaggedValue)
- Component::KLVData (StrongReferenceVector of KLVData)

Declared items are those which have TaggedValueDefinition or KLVDataDefinition objects in the AAF file.

It is optional for the importing application to preserve any other TaggedValues or KLVData items which might be present in imported files. In particular, they are not required to preserve undeclared items, or items which are declared for Properties other than where they are found.

An importing application may preserve other metadata in accordance with the precepts. Not all applications will be able to display all annotations.

Legacy note: Importing applications should be aware that legacy files might contain CommentMarkers in tracks with unexpected DataDefinitions such as DataDef_LegacyPicture or DataDef_LegacySound or others. The importing application may freely convert such DataDefinitions to DataDef_DescriptiveMetadata.

Legacy note: Importing applications may attempt to retrieve user comments from other properties in CommentMarkers, for example Event::Comment or TaggedValues named _ATN_CRM_COM in a CommentMarker::CommentMarkerAttributeList, and place these in Component::UserComments in a TaggedValue named Comment.

10.1.2 Fallback behavior

If an importing application cannot support particular properties of a DescriptiveMarker, e.g. DescribedSlots, Position or Length, it shall ignore them. This fallback behavior shall be logged.

10.2 Annotation using DescriptiveFramework objects (Informative)

The design of MXF provides two new standard mechanisms for annotation, described in SMPTE EG42:

- The DescriptiveMarker class (subclass of CommentMarker, known as DMSegment in SMPTE 377M)

- The DescriptiveFramework class hierarchy (known as DMFramework in SMPTE 380M) defined by SMPTE 380M and future documents.

DescriptiveMarkers may be used independently of DescriptiveFrameworks.

The Edit Protocol requires exporting applications to support the DescriptiveMarker class, including the DescriptiveMarker::Description property. Note however that this is an optional property in AAF, and may not be present in an AAF file.

The Edit Protocol does not require either exporting or importing applications to explicitly support MXF DMS-1 metadata. Applications may support this or other EG42 based Descriptive Metadata Schemes, following the precepts of Section 4.2.4.

11 Effects Model

11.1 Effect Definitions

All effects in an AAF file shall be defined with OperationDefinition and ParameterDefinition objects in the Dictionary.

The DefinitionObject::Name property of all DefinitionObjects in the Dictionary shall be a valid, meaningful name.

An OperationDefinition object in an AAF file shall have a data definition value consistent with the data definition of an OperationGroup object that references it. The effects defined in the Edit Protocol should be used with the AAF v1.1 data definition values of DataDef_Picture or DataDef_Sound. The deprecated data definition values of DataDef_LegacyPicture and DataDef_LegacySound should only be used when working with legacy AAF v1.0 files.

11.1.1 Fallback behavior

If the importing application does not support an effect or parameter, it may be ignored. This fallback behaviour shall be logged, including the DefinitionObject::Name and DefinitionObject::Description (if present) of the unsupported effect or parameter, to provide the user with an indication of the intended effect.

11.2 Invoking effects

An effect shall be invoked using an OperationGroup object. The OperationGroup specifies the effect to use and specifies the values of any parameters (constant value or time-varying).

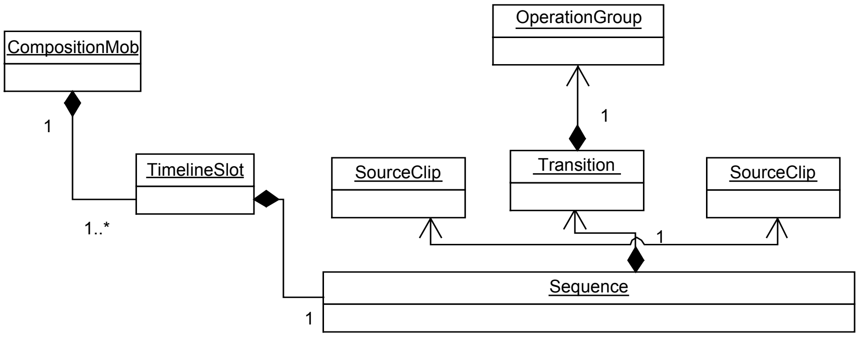
Effects shall be invoked using the simplest construction of objects for a given scenario, following the precepts of Section 4.2.1.

The context of the OperationGroup depends on the context of the effect, as described in the following sections.

11.2.1 Transition effect

The OperationGroup shall be contained in a Transition object between two Segments. The Transition object causes the Segments to overlap.

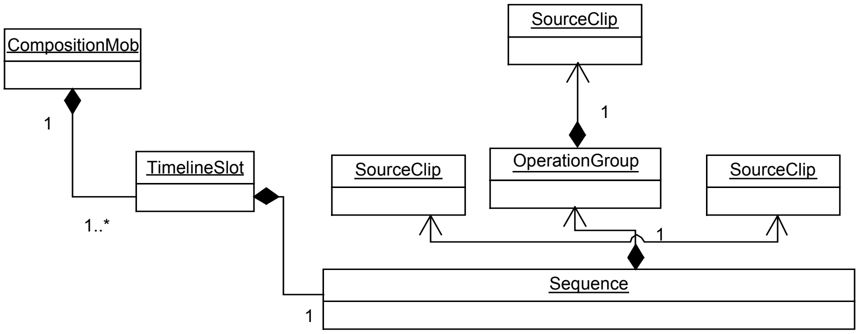
For example:



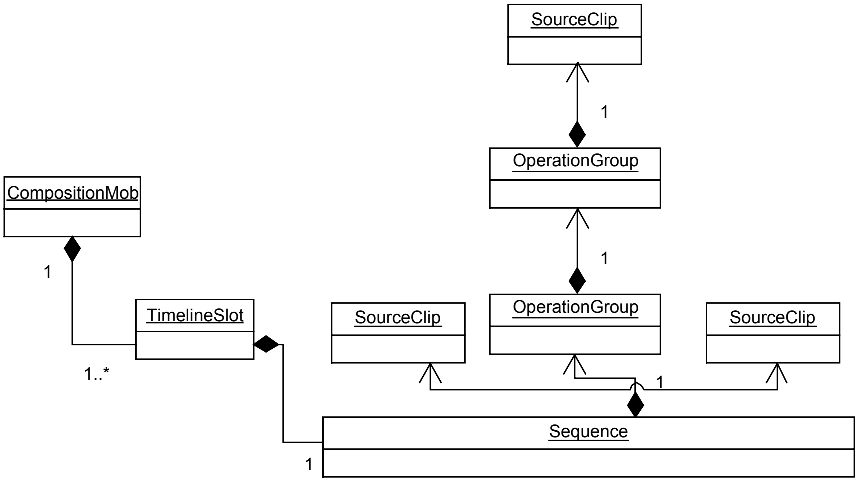
11.2.2 Composite effect

The OperationGroup shall not be contained in a Transition object. The OperationGroup::InputSegments property specifies the input segments that the effect requires.

For example, a composite effect in a CompositionMob with a single effect requiring one input is constructed so:



A composite effect in a CompositionMob with two effects chained together and requiring one input is constructed so:

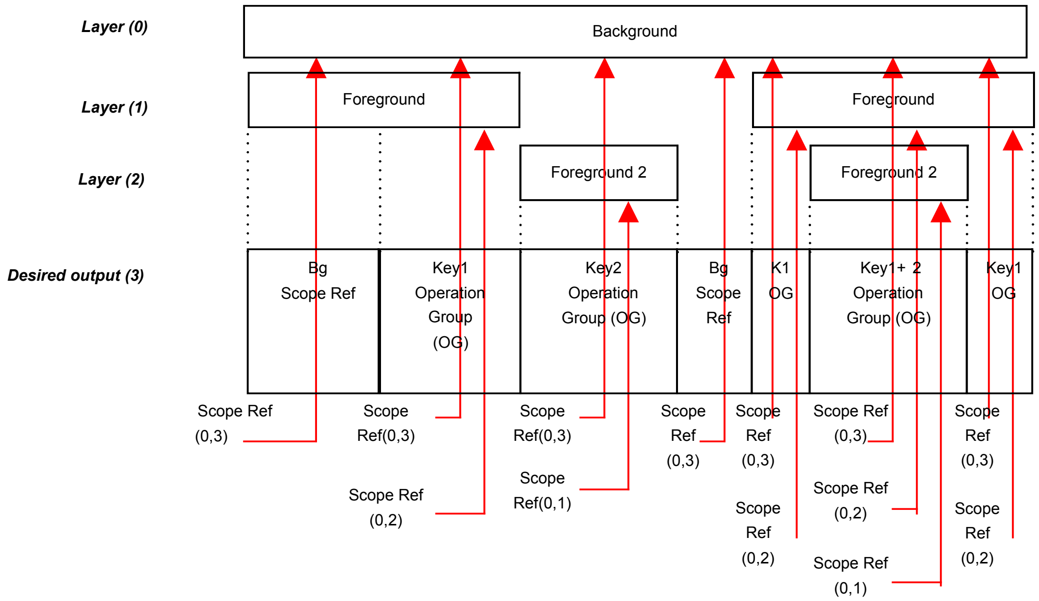


11.2.3 Multiple effects sharing common material

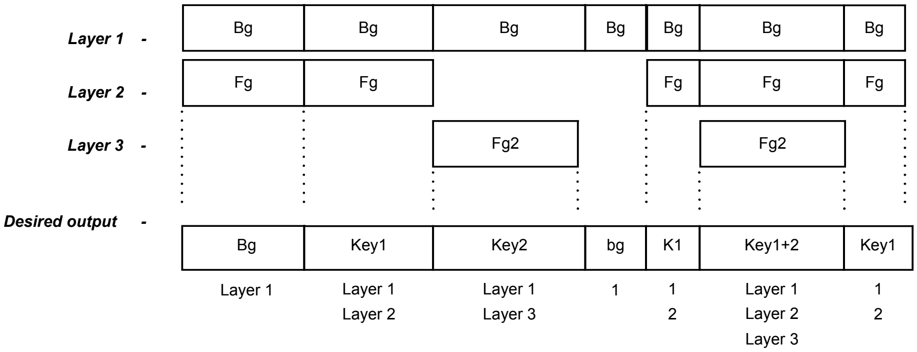
Multiple effects sharing common material shall be constructed within a NestedScope object. A continuous section of common material, used by several effects, shall be represented as a single Segment in the NestedScope. This material may be referenced from the output track of the NestedScope by ScopeReference objects, or OperationGroup objects with ScopeReferences as InputSegments.

This kind of representation maintains the fidelity of the timeline between applications, by avoiding unnecessary chopping-up of common material references as each effect changes to the next.

For example:

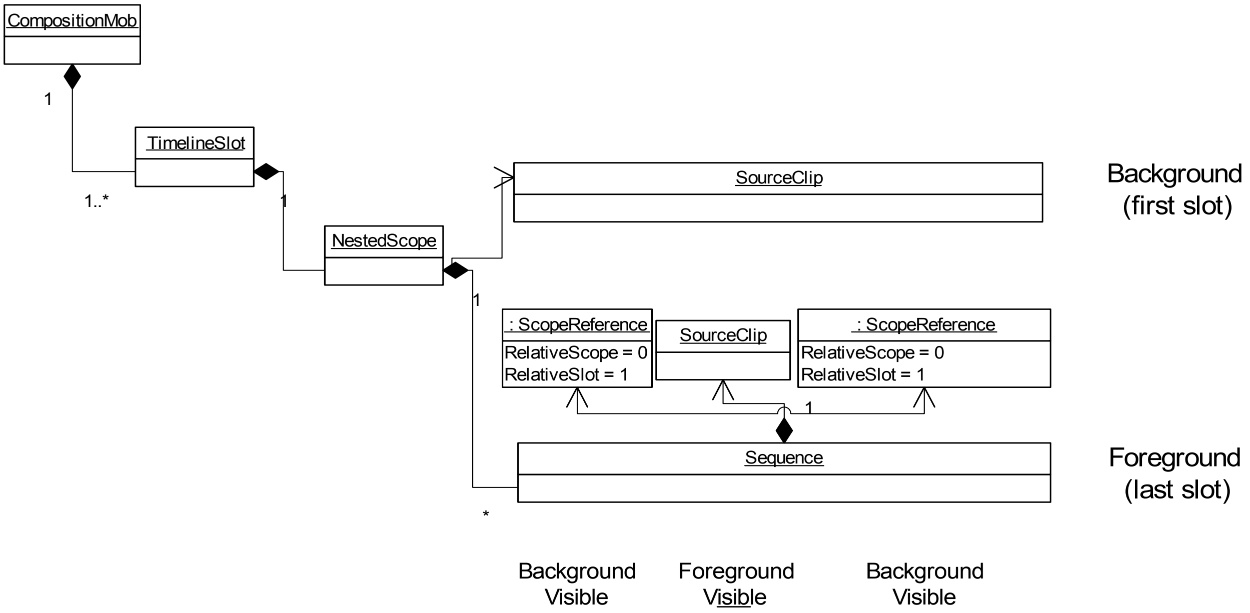


Informative note: An inferior representation of the same effects, in which the material is chopped-up across all tracks whenever a change in the desired output is required, is shown below for comparison. The Edit Protocol does not specify this representation.



11.3 Invoking effects on multiple layers

Effects on multiple layers shall be constructed within a NestedScope object. The layers shall be ordered background to foreground within the Slots of the NestedScope.



11.3.1 Fallback behavior

If the importing application does not support the required number of layers, a subset shall be imported into the application. This fallback behavior shall be logged. The user should be able to select which layers are imported.

11.4 Nested tracks

A nested group of tracks within a Segment of a CompositionMob shall be specified using a NestedScope object.

A NestedScope is used where a group of tracks are combined to form a single output. The output is typically formed by an effect or a reference to a nested track.

11.4.1 NestedScope definition

The NestedScope::Slots property contains an ordered vector of Segment objects. The last Segment defines the output of the NestedScope and the preceding Segments are the nested tracks.

11.4.2 ScopeReference definition

A ScopeReference references back to preceding tracks within a NestedScope or Mob, or outward to a containing NestedScope.

A ScopeReference shall reference an existing MobSlot or NestedScope track.

11.5 Effect parameter variation over time

Effect parameters shall be specified using ConstantValue or VaryingValue objects.

11.5.1 Interpolation for Time-varying parameters

Interpolation between ControlPoints within VaryingValue objects shall be specified as one of the following:

- ConstantInterpolator (step function)
- LinearInterpolator (equal amplitude)
- LogInterpolator
- PowerInterpolator (equal power)
- BSplineInterpolator

Informative note: The AAF reference implementation currently only implements linear and constant interpolators.

11.5.2 Fallback behavior

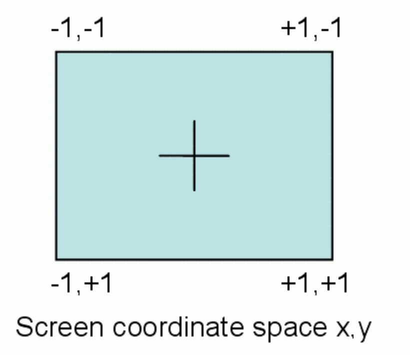
If the importing application does not support the required interpolation, it may fallback to linear interpolation. This fallback behavior shall be logged.

If the importing application does not support time-varying parameters, it may fallback to applying an average of the ControlPoint values. This fallback behavior shall be logged.

11.6 2D Effect parameter normalization

The position and size parameters for 2D effects defined in the Edit Protocol use a normalized interval from –1 to 1, as follows:

- The image has an origin 0,0 at its center
- The top left corner is (-1, -1)
- The top right corner is (1, -1)
- The bottom left corner is (-1, 1)
- The bottom right corner is (1, 1)



This kind of normalization is known as “VH normalization” because V and H axes are independently normalized to the range (-1, +1). The normalization is independent of the image aspect ratio (the aspect ratio is determined from the essence).

Effects that take multiple video inputs shall have all of their parameters VH normalized to the first video input unless explicitly stated otherwise in the effect definition.

Informative note: A traditional approach taken to 2D effect parameters in the past has been to describe them in “screen units”. For a 4:3 image the screen would be divided into 8 x 6 units with a center at 0,0 and ranging from -4 to +4 (left to right) and -3 to +3 (top to bottom). As an example, a movement of the image in the X plane of +4 units would move the center point of the image to the right edge of the screen. This process can be readily scaled up to handle 16:9 data where the screen now becomes -16 to +16 (left to right) and -9 to +9 (top to bottom) with the screen center at 0,0. However, to work with many different aspect ratios or arbitrarily sized images, a more general solution is required. Therefore the Edit Protocol uses the normalized interval from -1 to 1 rather than the traditional system of screen units.

Informative note: An alternative approach to parameter normalization is known as “V normalization”. This normalization uses a coordinate system where the pixels are square and the maximum X value corresponds to the aspect ratio of the image. For example a 4:3 image will have the coordinate range ($-4/3 < x < +4/3$, $-1 < y < +1$). The Edit Protocol introduces this term informatively but does not define any effects that use this normalization.

12 Effects Dictionary

12.1 Video Dissolve effect

A dissolve between overlapping video clips shall be specified using the Video Dissolve effect.

The Video Dissolve effect shall only be used by an OperationGroup within a Transition object.

12.1.1 Video Dissolve effect definition

OperationDefinition property	Value
Name	Video Dissolve
Description	Video Dissolve
Identification	OperationDef_VideoDissolve
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	2
InputSegments	Not specified. Input Segments are implicitly the Segment preceding and the Segment following the Transition object containing the OperationGroup. Segment “A” precedes the Transition object. Segment “B” follows the Transition object.
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_Level (optional; default is a VaryingValue object with two control points: Value 0 at time 0, and value 1 at time 1)

12.1.2 Level parameter definition

PropertyDefinition property	Value
Name	Level
Description	Level
Identification	ParameterDef_Level
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 Output pixel value P is computed by combining the input pixels A and B with the formula: P = (Level*B) + ((1-Level)*A)

12.1.3 Importing application

An importing application that supports video dissolves shall accept and present the Video Dissolve effect, following the precepts of Section 4.2.4.

12.2 SMPTE Video Wipe effect

Video Wipes are simple two-source visual effects defined by SMPTE 258M.

Informative note: There are many more wipe patterns defined by SMPTE 258M than are commonly used by video authoring applications.

Video Wipe effects that are defined within SMPTE 258M shall be specified using the SMPTE Video Wipe effect. The SMPTE Video Wipe effect shall only be used by an OperationGroup within a Transition object.

12.2.1 SMPTE Video Wipe effect definition

OperationDefinition property	Value
Name	SMPTE Video Wipe
Description	SMPTE Video Wipe
Identification	OperationDef_SMPTEVideoWipe
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	2
InputSegments	Not specified. Input Segments are implicitly the Segment preceding and the Segment following the Transition object containing the OperationGroup. Segment “A” precedes the Transition object. Segment “B” follows the Transition object.
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_SMPTEWipeNumber (required)
	ParameterDef_SMPTEReverse (optional; default value is False)
	ParameterDef_Level (optional; default is a VaryingValue object with two control points: Value 0 at time 0, and value 1 at time 1)

12.2.2 SMPTE Wipe Number parameter definition

PropertyDefinition property	Value
Name	SMPTE Wipe Number
Description	SMPTE Wipe Number as defined in Section 7.6.33 of SMPTE 258M
Identification	ParameterDef_SMPTEWipeNumber
Type	TypeID_Int32
DisplayUnits	Not specified

12.2.3 SMPTE Reverse parameter definition

PropertyDefinition property	Value
Name	SMPTE Reverse
Description	SMPTE Reverse
Identification	ParameterDef_SMPTEReverse
Type	TypeID_Boolean
DisplayUnits	Not specified

12.2.4 Level parameter definition

PropertyDefinition property	Value
Name	Level
Description	Level
Identification	ParameterDef_Level
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 (0 = 0% complete, 1=100% complete)

12.2.5 Importing application

An importing application that supports video wipes defined within SMPTE 258M shall accept and present the SMPTE Video Wipe effect, following the precepts of Section 4.2.4.

12.3 Video Speed Control effect

A video speed control effect, describing alterations to the playback speed of a clip including forwards and backwards directions, shall be specified using a Video Speed Control effect.

The Video Speed Control effect shall only be used by an OperationGroup not contained within a Transition object.

12.3.1 Video Speed Control effect definition

OperationDefinition property	Value
Name	Video Speed Control
Description	Video Speed Control
Identification	OperationDef_VideoSpeedControl
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	True
DegradeTo	Not specified
Bypass	1
ParametersDefined	ParameterDef_SpeedRatio (required)

12.3.2 Speed Ratio parameter definition

Speed Ratio defines the ratio of output length to input length.

For example, a ratio of 2/1 means that the output is twice as long as the input, appearing as a slow motion effect. A ratio of 1/2 would appear as fast motion since the input segment would be reduced to half its time. A negative value means that the frames are played in reverse order. A ratio of -1/1 would appear as backwards but at the normal frame rate. A ratio of -1/2 would appear as backward and fast motion.

A ratio value of 0 shall not be used.

Informative note: The Video Speed Control effect cannot be used to specify a video frame repeat. Instead, use the Video Repeat effect.

PropertyDefinition property	Value
Name	Speed Ratio
Description	Speed Ratio
Identification	ParameterDef_SpeedRatio
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-(2^31) to (2^31)-1, excluding 0

12.3.3 Importing application

An importing application that supports video speed control effects shall accept and present the Video Speed Control effect, following the precepts of Section 4.2.4.

12.4 Video Repeat effect

A video frame repeat effect, describing a clip that has a single essence frame as their original source, shall be specified using a Video Repeat effect.

The Video Repeat effect shall only be used by an OperationGroup not contained within a Transition object.

12.4.1 Video Repeat effect definition

OperationDefinition property	Value
Name	Video Repeat
Description	Video Repeat
Identification	OperationDef_VideoRepeat
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	A single-frame reference to the source material
IsTimeWarp	True
DegradeTo	Not specified
Bypass	1
ParametersDefined	None

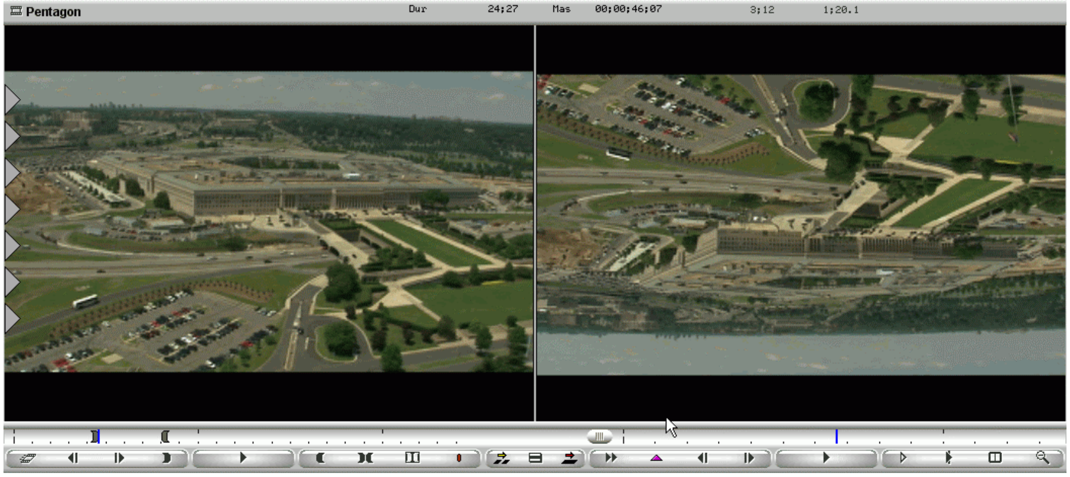
An alternative means of specifying this effect is defined by AAF, in which a SourceClip in a TimelineMobSlot references a still frame in a StaticMobSlot. The Edit Protocol deprecates the use of this alternative representation. An exporting application shall not use this alternative representation and an importing application need not accept it.

12.4.2 Importing application

An importing application that supports video repeat effects shall accept and present the Video Repeat effect, following the precepts of Section 4.2.4.

12.5 Video Flip effect

A video flip effect changes the vertical orientation of the image. The example below shows the result of the video flip effect, with the original source on the left and the processed result on the right.



A video flip effect shall be specified using the Video Flip effect.

The Video Flip effect shall only be used by an OperationGroup not contained within a Transition object.

12.5.1 Video Flip effect definition

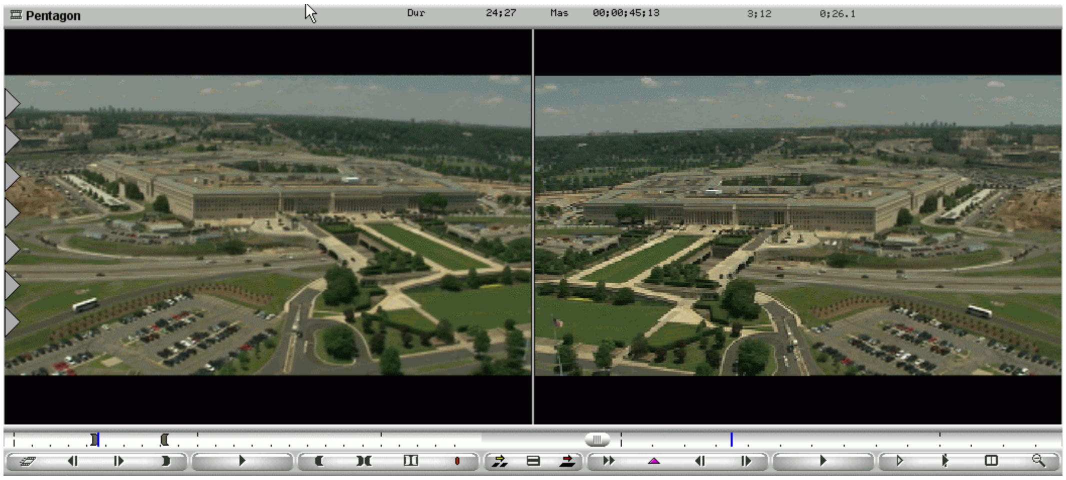
OperationDefinition property	Value
Name	Video Flip
Description	Video Flip
Identification	OperationDef_Flip
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1
ParametersDefined	None

12.5.2 Importing application

An importing application that supports video flip effects shall accept and present the Video Flip effect, following the precepts of Section 4.2.4.

12.6 Video Flop effect

A video flop effect changes the horizontal orientation of the image. The example below shows the result of the video flop effect, with the original source on the left and the processed result on the right.



A video flop effect shall be specified using the Video Flop effect.

The Video Flop effect shall only be used by an OperationGroup not contained within a Transition object.

12.6.1 Video Flop effect definition

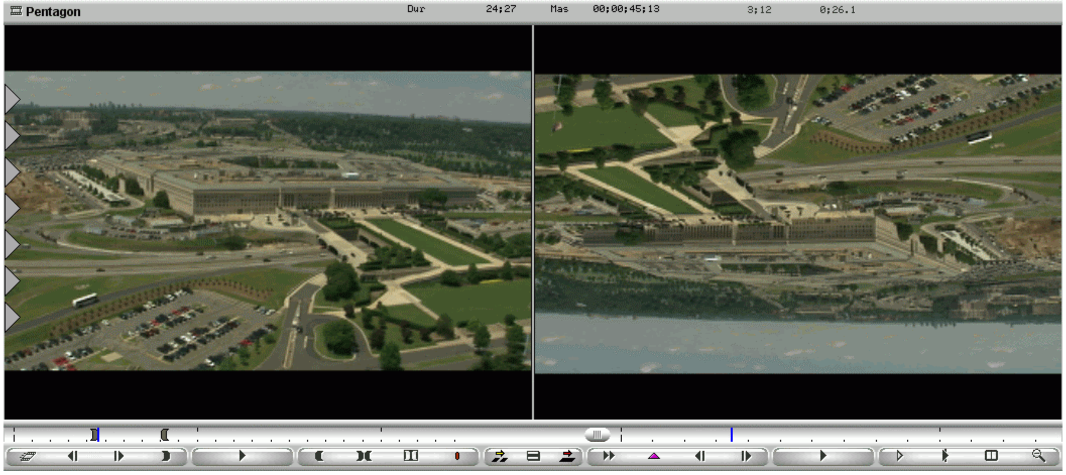
OperationDefinition property	Value
Name	Video Flop
Description	Video Flop
Identification	OperationDef_Flop
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1
ParametersDefined	None

12.6.2 Importing application

An importing application that supports video flop effects shall accept and present the Video Flop effect, following the precepts of Section 4.2.4.

12.7 Video Flip-Flop effect

A video flip-flop effect changes the vertical and horizontal orientation of the image, equivalent to a 180 degree rotation of the image. The example below shows the result of the video flip-flop effect, with the original source on the left and the processed result on the right.



A video flip-flop effect shall be specified using the Video Flip-Flop effect.

The Video Flop-Flop effect shall only be used by an OperationGroup not contained within a Transition object.

12.7.1 Video Flip-Flop effect definition

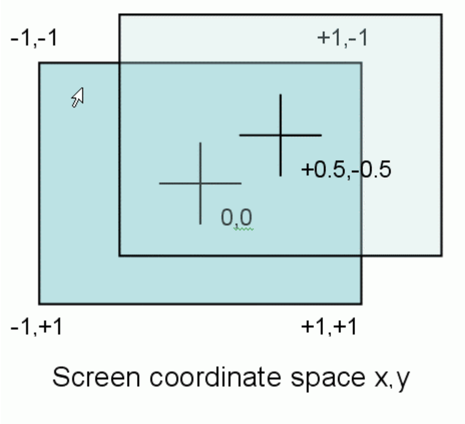
OperationDefinition property	Value
Name	Video Flip Flop
Description	Video Flip Flop
Identification	OperationDef_FlipFlop
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1
ParametersDefined	None

12.7.2 Importing application

An importing application that supports video flip-flop effects shall accept and present the Video Flip-Flop effect, following the precepts of Section 4.2.4.

12.8 Video Position effect

An image move shall be specified using the Video Position effect.



The Video Position effect shall only be used by an OperationGroup not contained within a Transition object.

12.8.1 Video Position effect definition

OperationDefinition property	Value
Name	Video Position
Description	Video Position
Identification	OperationDef_VideoPosition
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_PositionOffsetX (optional; default is 0)
	ParameterDef_PositionOffsetY (optional; default is 0)

12.8.2 Position Offset X parameter definition

PropertyDefinition property	Value
Name	Position OffsetX
Description	Position OffsetX
Identification	ParameterDef_PositionOffsetX
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-(2^31) to (2^31)-1 (-1 offsets image center to left of image, +1 offsets image center to right of image)

12.8.3 Position Offset Y parameter definition

PropertyDefinition property	Value

PropertyDefinition property	Value
Name	Position OffsetY
Description	Position OffsetY
Identification	ParameterDef_OffsetY
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-(2^31) to (2^31)-1 (-1 offsets image center to top of image, +1 offsets image center to bottom of image)

12.8.4 Importing application

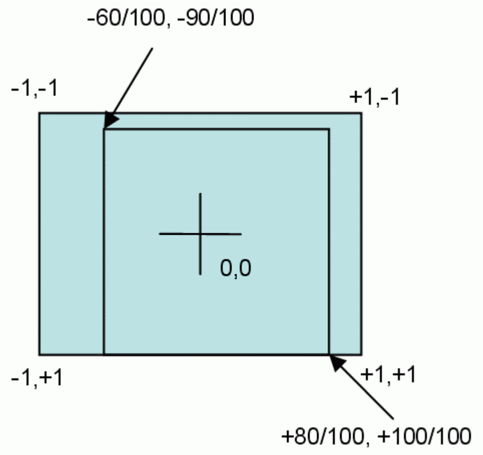
An importing application that supports image positioning shall accept and present the Video Position effect, following the precepts of Section 4.2.4.

12.9 Video Crop effect

To remove unwanted image information it is often necessary to crop the image. An image crop shall be specified using the Video Crop effect.

Crop parameters are specified for the left, right, top and bottom. These values are expressed as a fraction of the original image coordinate space.

In the example below the image is cropped to the reduced size represented by the fractional top left and bottom right coordinates.



The Video Crop effect shall only be used by an OperationGroup not contained within a Transition object.

12.9.1 Video Crop effect definition

OperationDefinition property	Value
Name	Video Crop
Description	Video Crop
Identification	OperationDef_VideoCrop
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1

OperationDefinition property	Value
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_CropLeft (optional; default is -1)
	ParameterDef_CropRight (optional; default is 1)
	ParameterDef_CropTop (optional; default is -1)
	ParameterDef_CropBottom (optional; default is 1)

12.9.2 Crop Left parameter definition

PropertyDefinition property	Value
Name	CropLeft
Description	CropLeft
Identification	ParameterDef_CropLeft
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is left of image, +1 is right of image)

12.9.3 Crop Right parameter definition

PropertyDefinition property	Value
Name	CropRight
Description	CropRight
Identification	ParameterDef_CropRight
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is left of image, +1 is right of image)

12.9.4 Crop Top parameter definition

PropertyDefinition property	Value
Name	CropTop
Description	CropTop
Identification	ParameterDef_CropTop
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1

PropertyDefinition property	Value
	(-1 is top of image, +1 is bottom of image)

12.9.5 Crop Bottom parameter definition

PropertyDefinition property	Value
Name	CropBottom
Description	CropBottom
Identification	ParameterDef_CropBottom
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is top of image, +1 is bottom of image)

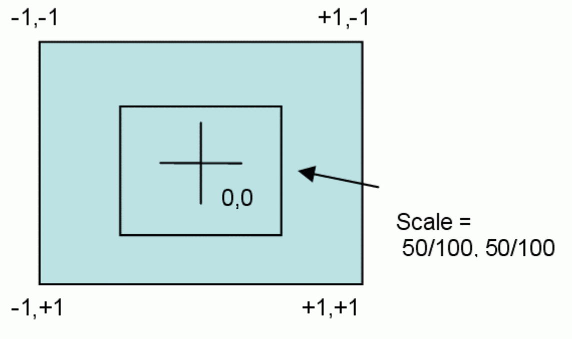
12.9.6 Importing application

An importing application that supports image cropping shall accept and present the Video Crop effect, following the precepts of Section 4.2.4.

12.10 Video Scale effect

Image scaling shall be specified using the Video Scale effect.

The required width and height of the image can be represented as fractions of the original image width and height with the result centered about 0,0.



The Video Scale effect shall only be used by an OperationGroup not contained within a Transition object.

12.10.1 Video Scale effect definition

OperationDefinition property	Value
Name	Video Scale
Description	Video Scale
Identification	OperationDef_VideoScale
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1

OperationDefinition property	Value
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_ScaleX (optional; default is 1)
	ParameterDef_ScaleY (optional; default is 1)

12.10.2 Scale X parameter definition

PropertyDefinition property	Value
Name	ScaleX
Description	ScaleX
Identification	ParameterDef_ScaleX
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to (2^31)-1 (1 is original size)

12.10.3 Scale Y parameter definition

PropertyDefinition property	Value
Name	ScaleY
Description	ScaleY
Identification	ParameterDef_ScaleY
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to (2^31)-1 (1 is original size)

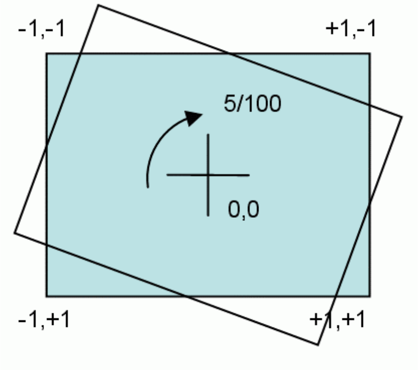
12.10.4 Importing application

An importing application that supports image scaling shall accept and present the Video Scale effect, following the precepts of Section 4.2.4.

12.11 Video Rotate effect

Image rotation shall be specified using the Video Rotate effect.

The rotation of the image about its center is represented by a single parameter scaled 0 to 1 for one complete clockwise rotation.



The Video Rotate effect shall only be used by an OperationGroup not contained within a Transition object.

12.11.1 Video Rotate effect definition

OperationDefinition property	Value
Name	Video Rotate
Description	Video Rotate
Identification	OperationDef_VideoRotate
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_Rotation (required)

12.11.2 Rotation parameter definition

PropertyDefinition property	Value
Name	Rotation
Description	Rotation
Identification	ParameterDef_Rotation
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 (0 is no rotation, 1 is one full clockwise rotation)

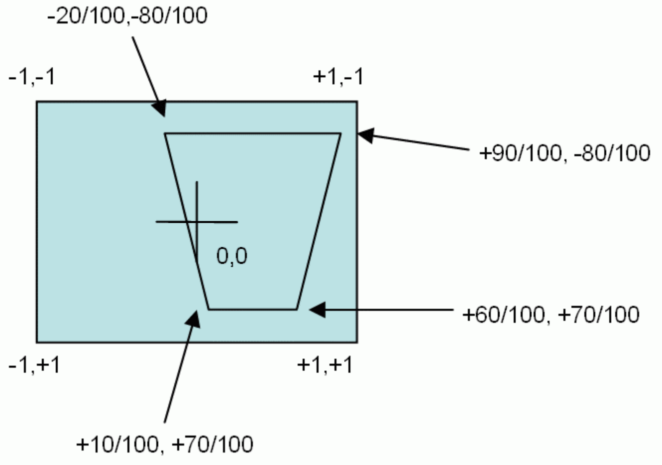
12.11.3 Importing application

An importing application that supports image rotation shall accept and present the Video Rotate effect, following the precepts of Section 4.2.4.

12.12 Video Corner Pinning effect

Image corner pinning shall be specified using the Video Corner Pinning effect.

Image corner pinning uses four parameters which represent where the corners of the original image have moved to.



In the Figure, the four corners have been moved to new positions. The origin coordinates are the original corners of the image specifically not taking into account any crop that might be applied.

The Video Corner Pinning effect shall only be used by an OperationGroup not contained within a Transition object.

12.12.1 Video Corner Pinning effect definition

OperationDefinition property	Value
Name	Video Corner Pinning
Description	Video Corner Pinning
Identification	OperationDef_VideoCornerPinning
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_PinTopLeftX (optional; default is -1)
	ParameterDef_PinTopLeftY (optional; default is -1)
	ParameterDef_PinTopRightX (optional; default is 1)
	ParameterDef_PinTopRightY (optional; default is -1)
	ParameterDef_PinBottomLeftX (optional; default is -1)
	ParameterDef_PinBottomLeftY (optional; default is 1)
	ParameterDef_PinBottomRightX (optional; default is 1)
	ParameterDef_PinBottomRightY (optional; default is 1)

12.12.2 Pin Top Left X parameter definition

PropertyDefinition property	Value
Name	Pin Top Left X
Description	Pin Top Left X
Identification	ParameterDef_PinTopLeftX
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is left of image, 1 is right of image)

12.12.3 Pin Top Left Y parameter definition

PropertyDefinition property	Value
Name	Pin Top Left Y
Description	Pin Top Left Y
Identification	ParameterDef_PinTopLeftY
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is top of image, 1 is bottom of image)

12.12.4 Pin Top Right X parameter definition

PropertyDefinition property	Value
Name	Pin Top Right X
Description	Pin Top Right X
Identification	ParameterDef_PinTopRightX
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is left of image, 1 is right of image)

12.12.5 Pin Top Right Y parameter definition

PropertyDefinition property	Value
Name	Pin Top Right Y
Description	Pin Top Right Y
Identification	ParameterDef_PinTopRightY
Type	TypeID_Rational

PropertyDefinition property	Value
DisplayUnits	Not specified
Range	-1 to 1 (-1 is top of image, 1 is bottom of image)

12.12.6 Pin Bottom Left X parameter definition

PropertyDefinition property	Value
Name	Pin Bottom Left X
Description	Pin Bottom Left X
Identification	ParameterDef_PinBottomLeftX
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is left of image, 1 is right of image)

12.12.7 Pin Bottom Left Y parameter definition

PropertyDefinition property	Value
Name	Pin Bottom Left Y
Description	Pin Bottom Left Y
Identification	ParameterDef_PinBottomLeftY
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is top of image, 1 is bottom of image)

12.12.8 Pin Bottom Right X parameter definition

PropertyDefinition property	Value
Name	Pin Bottom Right X
Description	Pin Bottom Right X
Identification	ParameterDef_PinBottomRightX
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is left of image, 1 is right of image)

12.12.9 Pin Bottom Right Y parameter definition

PropertyDefinition property	Value

PropertyDefinition property	Value
Name	Pin Bottom Right Y
Description	Pin Bottom Right Y
Identification	ParameterDef_PinBottomRightY
Type	TypeID_Rational
DisplayUnits	Not specified
Range	-1 to 1 (-1 is top of image, 1 is bottom of image)

12.12.10 Importing application

An importing application that supports image corner pinning shall accept and present the Video Corner Pinning effect, following the precepts of Section 4.2.4.

12.13 Alpha with Video Key effect

Alpha Transparency provides a mechanism for titles and travelling mattes.

An alpha key effect in which the alpha channel is within the foreground video essence shall be specified using the Alpha With Video Key effect.

The alpha channel shall be contained within a channel of the foreground video essence, using either RGBA or YUVA. The essence descriptor shall be an RGBADescriptor object (or a sub-class) for RGBA, or a CDCIDescriptor object (or a sub-class) for CDCI. The alpha channel shall be indicated by the essence descriptor.

The DigitalImageDescriptor::AlphaTransparency property shall be present and specify whether the minimum Alpha value or the maximum Alpha value indicates complete transparency. The ParameterDef_AlphaKeyInvertAlpha parameter indicates whether the transparency indicated by the DigitalImageDescriptor::AlphaTransparency property should be inverted.

In the NestedScope illustration of Section 11.2.3, an Alpha with Video Key effect would be Key 1 or Key 2. In this case the NestedScope contains two layers, although those layers do not need to be adjacent.

The Alpha with Video Key effect shall only be used by an OperationGroup not contained within a Transition object.

12.13.1 Alpha with Video Key effect definition

OperationDefinition property	Value
Name	Alpha With Video Key effect
Description	Alpha With Video Key effect
Identification	OperationDef_VideoAlphaWithinVideoKey
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	2
InputSegments	In the following order: Background Segment Foreground Segment
IsTimeWarp	False
DegradeTo	Not specified

OperationDefinition property	Value
Bypass	1 (background)
ParametersDefined	ParameterDef_AlphaKeyInvertAlpha (optional; default False)

12.13.2 Alpha with Video Key Invert Alpha parameter definition

PropertyDefinition property	Value
Name	InvertAlpha
Description	InvertAlpha
Identification	ParameterDef_AlphaKeyInvertAlpha
Type	TypeID_Boolean
DisplayUnits	Not specified
Range	True or False

12.13.3 Importing application

An importing application that supports alpha keying shall accept and present the Alpha With Video Key effect, following the precepts of Section 4.2.4.

12.14 Separate-Alpha Key effect

Alpha Transparency provides a mechanism for titles and travelling mattes.

An alpha key effect in which the alpha channel essence is separate from the video foreground and background essence shall be specified using the Separate-Alpha Key effect.

The alpha channel shall be contained within a channel of separate essence, using either RGBA or YUVA. The essence descriptor shall be an RGBADescriptor object (or a sub-class) for RGBA, or a CDCIDescriptor object (or a sub-class) for CDCI. Where the alpha channel is indicated by the essence descriptor, that channel shall be used as the alpha channel. Where the alpha channel is not indicated by the essence descriptor, the luminance value of the essence shall be used as the alpha channel.

The DigitalImageDescriptor::AlphaTransparency property shall be present and specify whether the minimum Alpha value or the maximum Alpha value indicates complete transparency. The ParameterDef_AlphaKeyInvertAlpha parameter indicates whether the transparency indicated by the DigitalImageDescriptor::AlphaTransparency property should be inverted.

In the NestedScope illustration of Section 11.2.3, a Separate Alpha Key effect would be Key 1+2. In this case the NestedScope contains three layers, although those layers do not need to be adjacent. Layer 2 is the alpha, layer 1 is the foreground and layer 0 is the background.

The Separate Alpha Key effect shall only be used by an OperationGroup not contained within a Transition object.

12.14.1 Separate Alpha Key effect definition

OperationDefinition property	Value
Name	Separate Alpha Key effect
Description	Separate Alpha Key effect
Identification	OperationDef_VideoSeparateAlphaKey
DataDefinition	DataDef_Picture or DataDef_LegacyPicture

OperationDefinition property	Value
Category	Not specified
NumberInputs	2
InputSegments	In the following order: Background Segment Foreground Segment Alpha Segment
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1 (background)
ParametersDefined	ParameterDef_AlphaKeyInvertAlpha (optional; default False)

12.14.2 Alpha with Video Key Invert Alpha parameter definition

PropertyDefinition property	Value
Name	InvertAlpha
Description	InvertAlpha
Identification	ParameterDef_AlphaKeyInvertAlpha
Type	TypeID_Boolean
DisplayUnits	Not specified
Range	True or False

12.14.3 Importing application

An importing application that supports alpha keying shall accept and present the Separate Alpha Key effect, following the precepts of Section 4.2.4.

12.15 Luminance Key effect

A luminance key effect shall be specified using the Luminance Key effect. The luminance is determined from the Y or the RGB channels.

In the NestedScope illustration of Section 11.2.3, a Luminance Key effect would be Key 1 or Key 2. In this case the NestedScope contains two layers, although those layers do not need to be adjacent.

The Luminance Key effect shall only be used by an OperationGroup not contained within a Transition object.

12.15.1 Luminance Key effect definition

OperationDefinition property	Value
Name	Luminance Key
Description	Luminance Key
Identification	OperationDef_VideoLuminanceKey
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	2
InputSegments	In the following order:

OperationDefinition property	Value
	Background Segment Foreground Segment
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1 (background)
ParametersDefined	ParameterDef_LumKeyLevel (required) ParameterDef_LumKeyClip (required)

12.15.2 Luminance Key Level parameter definition

PropertyDefinition property	Value
Name	Luminance Key Level
Description	Luminance Key Level
Identification	ParameterDef_LumKeyLevel
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to (2^31)-1 (0 is black, 1 is white)

12.15.3 Luminance Key Clip parameter definition

PropertyDefinition property	Value
Name	Luminance Key Clip
Description	Luminance Key Clip
Identification	ParameterDef_LumKeyClip
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 (0 is black, 1 is white)

12.15.4 Importing application

An importing application that supports luminance keying shall accept and present the Luminance Key effect, following the precepts of Section 4.2.4.

12.16 Chroma Key effect

A chroma key effect shall be specified using the Chroma Key effect.

Informative note: The Edit Protocol does not specify any parameters for the Chroma Key effect—the exporting implementation may define and use its own parameters. The Edit Protocol specifies only the identity of the effect and the ordering of the input segments.

In the NestedScope illustration of Section 11.2.3, a Chroma Key effect would be Key 1 or Key 2. In this case the NestedScope contains two layers, although those layers do not need to be adjacent.

The Chroma Key effect shall only be used by an OperationGroup not contained within a Transition object.

12.16.1 Chroma Key effect definition

OperationDefinition property	Value
Name	Chroma Key
Description	Chroma Key
Identification	OperationDef_VideoChromaKey
DataDefinition	DataDef_Picture or DataDef_LegacyPicture
Category	Not specified
NumberInputs	2
InputSegments	In the following order: Background Segment Foreground Segment
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1 (background)
ParametersDefined	Not specified

12.16.2 Importing application

An importing application that supports chroma keying shall accept and present the Chroma Key effect, following the precepts of Section 4.2.4.

12.17 Audio Gain effect

An audio gain effect, specifying how the level of an audio signal is amplified or attenuated over time, shall be specified using the Mono Audio Gain effect.

The Mono Audio Gain effect shall only be used by an OperationGroup not contained within a Transition object.

12.17.1 Audio Gain effect definition

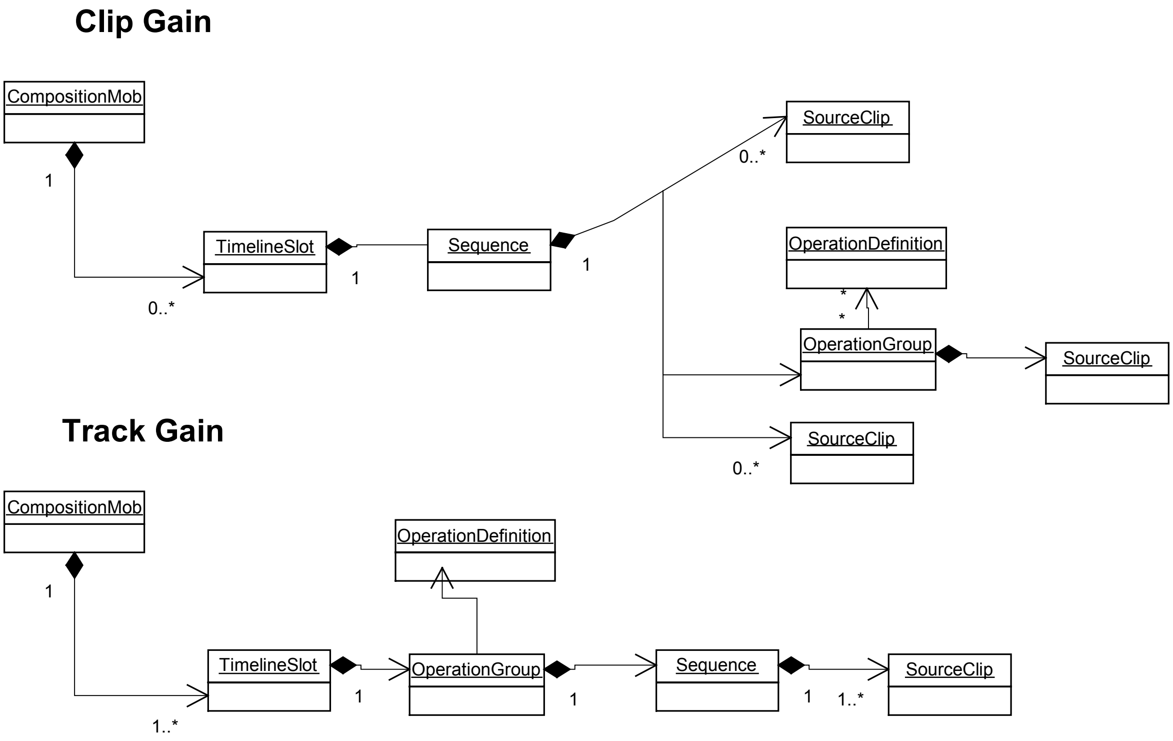
OperationDefinition property	Value
Name	Mono Audio Gain
Description	Mono Audio Gain
Identification	OperationDef_MonoAudioGain
DataDefinition	DataDef_Sound or DataDef_LegacySound
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1
ParametersDefined	ParameterDef_Amplitude (required)

12.17.2 Amplitude parameter definition

PropertyDefinition property	Value
Name	Amplitude multiplier
Description	Amplitude multiplier
Identification	ParameterDef_Amplitude
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to (2^31)-1, unity gain is 1 This multiplier is scaled linearly. To convert this amplitude multiplier to dB: value in dB = 20 $\log_{10}$ (amplitude multiplier)

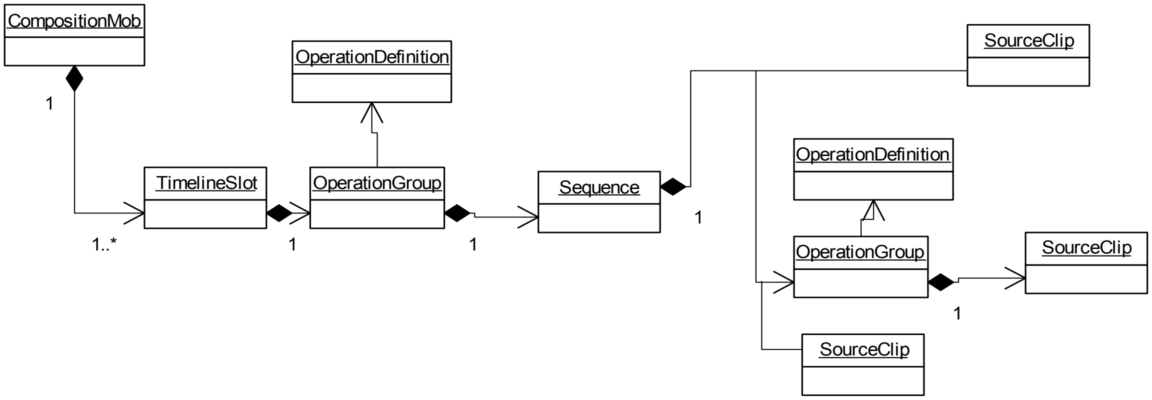
12.17.3 Clip Gain and Track Gain

The amplitude multiplier parameter specifies the gain value as a multiplier. Audio gain may be applied to an individual clip and/or a track:



12.17.4 Serializing Gain

Both clip and track gain may apply to a clip:



The resulting gain shall be calculated as follows:

1. Multiply each audio sample by the clip gain amplitude multiplier (if any is specified)
2. Multiply the resulting value by the track gain amplitude multiplier (if any is specified)
3. Apply the appropriate saturation (clipping) algorithm to the result computed in step 2.

12.17.5 Importing application

An importing application that supports audio gain effects shall accept and present the Mono Audio Gain effect, following the precepts of Section 4.2.4.

12.18 Audio Pan effect

An audio pan effect, specifying the left/right balance of mono audio sources in a stereo sound field, shall be specified using the Mono Audio Pan effect.

A Mono Audio Pan effect shall only be applied to audio tracks that refer to mono essence.

Audio pan effects shall not be nested. The top-level pan value shall be applied.

The Mono Audio Pan effect shall only be used by an OperationGroup not contained within a Transition object.

12.18.1 Audio Pan effect definition

OperationDefinition property	Value
Name	Mono Audio Pan
Description	Mono Audio Pan
Identification	OperationDef_MonoAudioPan
DataDefinition	DataDef_Sound or DataDef_LegacySound
Category	Not specified
NumberInputs	1
InputSegments	The essence to filter
IsTimeWarp	False
DegradeTo	Not specified
Bypass	1
ParametersDefined	ParameterDef_Pan (required)

12.18.2 Pan parameter definition

PropertyDefinition property	Value
Name	Pan value
Description	Pan value
Identification	ParameterDef_Pan
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 (0=full left, 1/2=half left & half right, 1=full right)

12.18.3 Track-based pan

Pan automation shall be track-based. If an application has a different native representation (e.g., clip-based pan), it shall convert to and from its native representation when exporting and importing the composition.

12.18.4 Importing application

An importing application that supports audio pan effects shall accept and present the Mono Audio Pan effect, following the precepts of Section 4.2.4.

12.19 Single-Parameter Audio Dissolve effect

A dissolve between overlapping audio clips, in which the fade out of the outgoing "A" segment is the mirror image in time of the fade in of the incoming "B" segment, shall be specified using the Single-Parameter Mono Audio Dissolve effect.

The Single-Parameter Mono Audio Dissolve effect shall only be used by an OperationGroup within a Transition object.

12.19.1 Audio Dissolve effect definition

OperationDefinition property	Value
Name	Mono Audio Dissolve
Description	Single-Parameter Mono Audio Dissolve
Identification	OperationDef_MonoAudioDissolve
DataDefinition	DataDef_Sound or DataDef_LegacySound
Category	Not specified
NumberInputs	2
InputSegments	Not specified. Input Segments are implicitly the Segment preceding and the Segment following the Transition object containing the OperationGroup. Segment "A" (outgoing) precedes the Transition object. Segment "B" (incoming) follows the Transition object.
IsTimeWarp	False
DegradeTo	Not specified
Bypass	Not specified
ParametersDefined	ParameterDef_Level (optional; default is a VaryingValue object with two control points: Value 0 at time 0, and value 1 at time 1)

12.19.2 Level parameter definition

PropertyDefinition property	Value
Name	Level
Description	Level
Identification	ParameterDef_Level
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 Output sample value P is computed by combining the input samples A and B with the formula: $P_{(t)} = (Level_{(t)}*B_{(t)}) + (Level_{(1-t)}*A_{(t)})$ where t=0 is the start of the transition, t=1 is the end of the transition Level is scaled linearly. To convert Level to dB: value in dB = 20 $\log_{10}$ (Level)

12.19.3 Importing application

An importing application that supports audio dissolves, in which the fade out of the outgoing “A” segment is the mirror image in time of the fade in of the incoming “B” segment, shall accept and present the Single-Parameter Mono Audio Dissolve effect, following the precepts of Section 4.2.4.

12.20 Two-Parameter Audio Dissolve effect

A dissolve between overlapping audio clips, in which the levels of the incoming “B” segment and outgoing “A” segment are independently defined, shall be specified using the Two-Parameter Mono Audio Dissolve effect.

Informative note: Such dissolves are commonly used by digital audio workstations.

The Two-Parameter Mono Audio Dissolve effect shall only be used by an OperationGroup within a Transition object.

12.20.1 Two-Parameter Audio Dissolve effect definition

OperationDefinition property	Value
Name	Two-Parameter Mono Audio Dissolve
Description	Two-Parameter Mono Audio Dissolve
Identification	OperationDef_TwoParameterMonoAudioDissolve
DataDefinition	DataDef_Sound or DataDef_LegacySound
Category	Not specified
NumberInputs	2
InputSegments	Not specified. Input Segments are implicitly the Segment preceding and the Segment following the Transition object containing the OperationGroup. Segment “A” (outgoing) precedes the Transition object. Segment “B” (incoming) follows the Transition object.
IsTimeWarp	False
DegradeTo	Not specified

OperationDefinition property	Value
Bypass	Not specified
ParametersDefined	ParameterDef_OutgoingLevel (optional; default is a VaryingValue object with two control points: Value 1 at time 0, and value 0 at time 1)
	ParameterDef_IncomingLevel (optional; default is a VaryingValue object with two control points: Value 0 at time 0, and value 1 at time 1)

12.20.2 Outgoing Level parameter definition

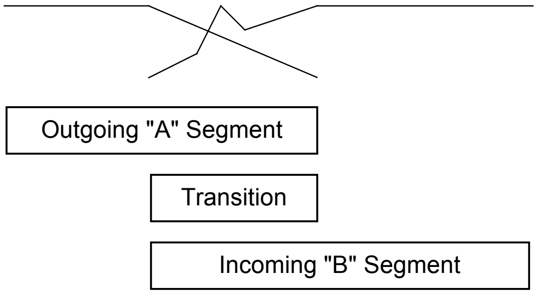
PropertyDefinition property	Value
Name	Outgoing Level
Description	Outgoing Level
Identification	ParameterDef_OutgoingLevel
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 Output sample value P is computed by combining the input samples A and B with the formula: P = (IncomingLevel*B) + (OutgoingLevel*A) Outgoing Level is scaled linearly. To convert Outgoing Level to dB: value in dB = 20 $\log_{10}$ (Outgoing Level)

12.20.3 Incoming Level parameter definition

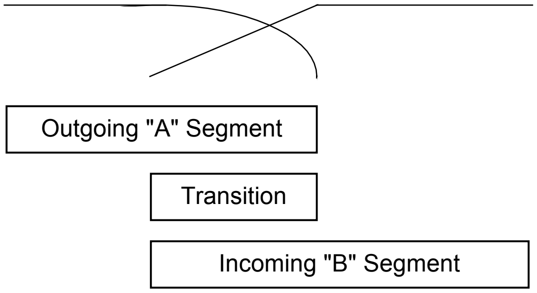
PropertyDefinition property	Value
Name	Incoming Level
Description	Incoming Level
Identification	ParameterDef_IncomingLevel
Type	TypeID_Rational
DisplayUnits	Not specified
Range	0 to 1 Output sample value P is computed by combining the input samples A and B with the formula: P = (IncomingLevel*B) + (OutgoingLevel*A) Incoming Level is scaled linearly. To convert Incoming Level to dB: value in dB = 20 $\log_{10}$ (Incoming Level)

12.20.4 Examples of Two-Parameter Audio Dissolve effect

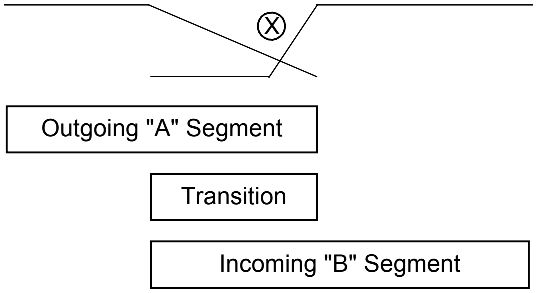
The following examples illustrate the use of the Two-Parameter Audio Dissolve effect.



Transition length: 4000 edit units
Fade Out: 4000 edit units, 2 points, linear
Fade In: 4000 edit units, 5 points, linear



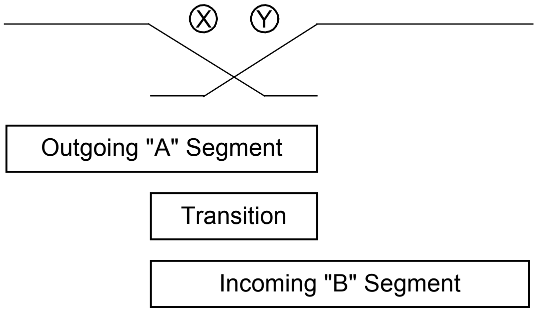
Transition length: 4000 edit units
Fade Out: 4000 edit units, 2 points, equal power
Fade In: 4000 edit units, 2 points, linear



Transition length: 4000 edit units
Fade Out: 4000 edit units, 2 points, linear
Fade In: 4000 edit units, 3 points, linear

Note: Although the incoming segment is first faded up only at point X, the incoming segment must be specified from the beginning of the Transition.

Informative note: If point X is the beginning of the referenced material, then specifying the incoming segment from the beginning of the Transition will create a SourceClip object that, until point X, references non-existent material. This exception to the usual requirement that a SourceClip only reference existing material is allowed in the Edit Protocol in order to support dissolves of this kind using the Transition class, and is the reason for the normative provision of Section 6.2.1.



Transition length: 4000 edit units
Fade Out: 4000 edit units, 3 points, linear
Fade In: 4000 edit units, 3 points, linear

Note: As in the previous example, although the incoming segment is first faded up only at point X, the incoming segment must be specified from the beginning of the Transition. Similarly, although the outgoing segment is completely faded down at point Y, the outgoing segment must be specified to the end of the Transition.

Informative note: If point Y is the end of the referenced material, then specifying the outgoing segment to the end of the Transition will create a SourceClip object that, after point Y, references non-existent material. This exception to the usual requirement that a SourceClip only reference existent material is allowed in the Edit Protocol in order to support dissolves of this kind using the Transition class, and is the reason for the normative provision of Section 6.2.1.

12.20.5 Importing application

If the SourceClip objects in a Two-Parameter Audio Dissolve effect reference non-existent material, an importing application shall treat this as a reference to Filler, and present as black picture or silent audio.

An importing application that supports audio dissolves, in which the levels of the incoming “B” segment and outgoing “A” segment are independently specified, shall accept and present the Two-Parameter Mono Audio Dissolve effect, following the precepts of Section 4.2.4.

12.20.6 Fallback behavior

If the importing application does not support audio dissolves in which the levels of the incoming “B” segment and outgoing “A” segment are independently specified, then it shall fallback to a dissolve in which the level of the incoming “B” segment is the complement of the level of the outgoing “A” segment, following the precepts of Section 4.2.4. This fallback shall be logged.

13 Optional properties that have unknown values

When the value of an optional property is unknown, the exporting application shall omit that property.

14 Structured Storage File Encoding

For Structured Storage encoding of the AAF file, the exporting application shall use a sector size of 4096 bytes. An importing application shall accept sector sizes of 512 or 4096 bytes.

15 Protocol Label

Header::OperationalPattern should be equal to OpEditProtocol.

16 Bibliography

1. Advanced Authoring Format, http://www.aafassociation.org