VSF TR-11 - Signal Transport and Timing Considerations for Ground-Cloud-Cloud-Ground workflows

February 21, 2024

Video Services Forum (VSF)
Technical Recommendation TR-11
Signal Transport and Timing Considerations for Ground-Cloud-Cloud-Ground workflows

February 20, 2024 (draft publication for public comment)

Introduction (Informational)

This recommendation concerns the interchange and timing of signals to-, from-, and within an interconnected composition of media processes within a composable compute environment (either private or multi-tenant, i.e. “cloud”). In such systems, the time consumed performing a given process might be highly variable - we refer to these software-abstracted media processes here as “Time-Non-Linear” Workflow Steps.

Additionally, this recommendation concerns the transport and interchange of media signals between Time-Linear (traditional method) and Time-Non-Linear processing paradigms, in order to facilitate the integration of real-time production media signals into composable compute environments.

It builds on (and references) other work within the VSF on related subjects. A novel timing model is introduced for managing buffering and latency in the Time-Non-Linear processing paths.

About the Video Services Forum

The Video Services Forum, Inc. (www.videoservicesforum.org) is an international association dedicated to video transport technologies, interoperability, quality metrics and education. The VSF is composed of service providers, users and manufacturers. The organization’s activities include:

- Providing forums to identify issues involving the development, engineering, installation, testing and maintenance of audio and video services;
- Exchanging non-proprietary information to promote the development of video transport service technology and to foster resolution of issues common to the video services industry;
- Identification of video services applications and educational services utilizing video transport services;
- Promoting interoperability and encouraging technical standards for national and international standards bodies.

The VSF is an association incorporated under the Not For Profit Corporation Law of the State of New York. Membership is open to businesses, public sector organizations and individuals worldwide. For more information on the Video Services Forum or this document, please e-mail opsmgr@videoservicesforum.org.

About This Document - Draft Publication Status

This document is published as a draft for public comments. Comments are best made through the discussion mechanism on the VSF Github repository associated with this document https://github.com/vsf-tv/gccg-api/discussions.

The VSF expects to issue additional drafts and/or a finalized version of this specification in the future which addresses the comments received.

Conformance Notation

Normative text describes elements of the design that are indispensable or contain the conformance language keywords: "shall," "should," or "may."

Informative text is potentially helpful to the user but not indispensable and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except the Introduction and any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:”

The keywords "shall" and "shall not" indicate requirements strictly to be followed to conform to the document and from which no deviation is permitted.

The keywords "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword “reserved” indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword “forbidden” indicates “reserved” and in addition indicates that the provision will never be defined in the future.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

Unless otherwise specified, the order of precedence of the types of normative information in this document shall be as follows: Normative prose shall be the authoritative definition; Tables shall be next; followed by formal languages; then figures; and then any other language forms.

1. Scope

This Technical Recommendation defines practices for the interchange of television content between traditional Time-Linear Workflows and Time-Non-Linear Workflows, and the timing models associated with Time-Non-Linear Workflows necessary to achieve a bounded latency when returning signals to a Time-Linear environment. Transports, operating points and connection management methods are also recommended for common use cases.

2. Normative References

AMWA BCP-006-01: NMOS with JPEG XS v1.0.0
AMWA IS-04 NMOS Discovery and Registration Specification (Stable) v1.31
AMWA IS-05 NMOS Device Connection Management Specification (Stable) v1.1.1
VSF TR-07:2022 Transport of JPEG XS Video in MPEG-2 TS over IP
VSF TR-08:2022 Transport of JPEG XS Video in ST 2110-22
VSF TR-09:2022 Transport of ST 2110 Media Essences over Wide Area Networks - Data Plane
VSF TR-09:2022 Transport of ST 2110 Media Essences over Wide Area Networks - Control Plane

3. Informative References

AMWA BCP-007-01: NMOS with NDI

4. Motivational Section (Informative)

Production and broadcast workflows have traditionally been based around performing processes on audio, video and data signals in a synchronous (time linear) fashion, continually maintaining alignment and enabling various signals to be manipulated, transported and exchanged in a predictable manner. In reality, the only points at which media *must* be processed in a Time-Linear fashion are during capture (e.g camera sensors, audio sampling) or presentation to a viewer, either as part of the workflow (e.g for commentary, MCR control) or as the end consumer.

Synchronous processing of media effectively limits that processing to real-time: Each step in the workflow expects to receive a steady input signal, therefore any workflow step can only complete its job of outputting the current video or audio frame immediately prior to commencing work upon the next one. While this may be acceptable for a chain of dedicated appliances or hardware based implementations, modern software defined applications running within composable compute environments may be able to complete a process and hand the result off to the next Workflow Step well in advance of receiving the next portion of incoming media. During this time, those compute resources could be put to some other use until required again for the assigned media process.

In order to integrate with existing Time-Linear systems (i.e. send to, or extract media from the composable compute environment), there must be a Time-Linear transport system at either end of the Time-Non-Linear Workflow. This technical recommendation will promote the use of existing transport mechanisms where practical in order to maintain interoperability and enable transition of media to and from the Time-Non-Linear domain.

5. Ground-Cloud / Cloud-Ground Transport Recommendations

Transport of contribution signals from Ground to Cloud - and likewise transport of intermediate contributions from Cloud to Ground - involves a trade-off of latency, computational complexity and transport bandwidth. There is no single perfect solution for all applications, however this recommendation notes several existing methods appropriate to different use cases in order to establish some common interoperability points in the marketplace.

- H.264 low latency over MP2TS, via VSF TR-06 (RIST)
- JPEG XS as described in TR-07, via VSF TR-06 (RIST)
- JPEG XS as described in TR-08, managed as described in VSF TR-09

Other codecs and transports are also in common use. The only presumptions that the remainder of this recommendation makes about these transports is that media is delivered to the cloud edge in a Time-Linear manner (or in a manner which can be reconstructed into a Time-Linear presentation order) and likewise that media is transported away from the cloud edge in a Time-Linear manner for presentation.

Origin Timestamps - notation of the original time of sampling such that Media Elements may be properly composed together during downstream production - remain a goal in the industry. This recommendation provides for the transport of origin time through the timing model definition below and also proposes mechanisms to infer origin time in cases where it is not available.

6. Workflow and Timing Model – Terms and Definitions

This recommendation develops a timing model for managing the time relationships of essence elements and managing the end-to-end latency of computational processes within a composable computing environment - including the sometimes necessary step of creating a time-linear output at the end of the Workflow. The timing model is based on the following definitions of terms:

- **Workflow Step:** A logical action or operation that handles or processes media.

- **Workflow:** Collection of Workflow Steps that make up a path along which media content will progress, either inside or outside of the composable compute environment

- **Media Element:** A (contiguous) group of video pixels (within a single frame, can be part of a frame (e.g. stripe) or whole image), or a (contiguous) group of audio samples, or a grouping of time-associated data. Different Workflow Steps may operate on Media Elements of differing sizes.

- **Media Flow:** A sequence of Media Elements belonging to the same media essence flow e.g. video, audio track which completely contain the essence

- **Time-Linear**: Processing of Media Elements - and transferring them to the next Workflow Step - at a constant rate, such that the progression of Media Elements through the Workflow Step occurs with evenly spaced intervals

- **Time-Non-Linear:** Processing each Media Element within a bounded amount of time, but with a minimum latency that might be substantially shorter than the bound - and transferring them to the next Workflow Step in a block manner when ready

- **Application-Ordered:** Media Elements are processed in the order received, but may be re-ordered within a Workflow Step prior to output. Each Media Element is timestamped such that presentation order can be re-created.

- **Local Timebase:** this is a timebase available to all of the Workflow Steps within the computing environment.

- **Flow Timebase:** this is the timebase of the Media Flow, which may differ from the Local Timebase. Different flows may have different Flow Timebases and each Flow Timebase is identified by a GMID (or a local-MAC based GMID). The frequency accuracy and wallclock relationship of the Flow Timebase is not specified, but is presumed to meet the requirements of the application.

- **Original Timebase:** this is the timebase of the flow as it enters the computing environment - in particular the Content Original Timestamp is a sample of this timebase.

- **Content Original Timestamp (COT):** this is a sample of the Original Timebase associated with each Media Element. This timestamp (and the GMID it is a sample of) remains associated with the related Media Element throughout the entire Time-Non-Linear Workflow process - unmodified. This timestamp may be accompanied by a discontinuity indicator which alerts when there is a discontinuous change to the timestamp at a given Media Element.

- **Local Arrival Timestamp (LAT)** – This is a sample of the Local Timebase within the Time-Non-Linear Workflow, stamped in a linear manner as the flow enters the non-linear domain. These local-entry-time stamps will proceed in regular (linear) increments (for fixed-rate Media Elements) but the increments might not be exactly the same as those measured in the Flow Timebase due to frequency drift between the Flow Timebase and the Local Timebase. (Note for MPEG-TS, the LAT needs to be based on the

Presentation Timestamp signaled and not the arrival timestamp. This accounts for the non-linear nature of the VBV in MPEG-TS.)

- **Local-Original-Offset** – for each Media Element, there is an offset between its Local Arrival Timestamp and its Content Original Timestamp. This offset may change slowly over time due to drift between the local and original timebases. The instantaneous value of this offset may be used to determine the local time of an intended action based on the original timebase value of the intended action. This value is not explicitly signaled, but can be derived whenever needed.

- **Synthetic Content** (clip playout or similar items that did not originate on the ground) must synthesize Content Origination Timestamps and (perhaps equal in value) Local Arrival Timestamps for use in downstream processing. These timestamps for the synthetic content could be referenced to the Local Timebase, or could be generated based on the timing provided in a Timing Reference Flow.

- **Timing Reference Flow** – a Media Flow with all the timestamps defined above, but which contains no (or trivial) video pixels, no (or trivial) audio, etc. The Timing Reference Flow can be consumed by a clip player or other Workflow Step to provide a source of frequency and time.

- **Tmin** - the minimum latency of the Workflow Step, including fully transferring the output(s) onto the fabric of the compute environment. This minimum includes any re-ordering delays required by the application.

- **T99** - the effective maximum latency of the Workflow Step, including fully transferring the output(s). The 99 denotes an expectation that the Workflow Step will fall within this maximum latency 99.9% of the time.

- **Temporal Conformance** – the action, short of full linearisation, of bringing a flow into a specified T99 envelope.

- **Production Timestamp(s)** - Media Elements may have one or more additional Production Timestamps which relate to events or times in the production. These timestamps might be used for relating different stored assets from past productions, or for other purposes out of the scope of this document.

7. Timing Model

7.1. General

The non-linear timing model in this document is designed to provide the information required to properly align various media signals to suit the production requirements, but also to properly convert these Time-Non-Linear signals into Time-Linear signals whilst minimizing accumulated latency and required delay buffering.

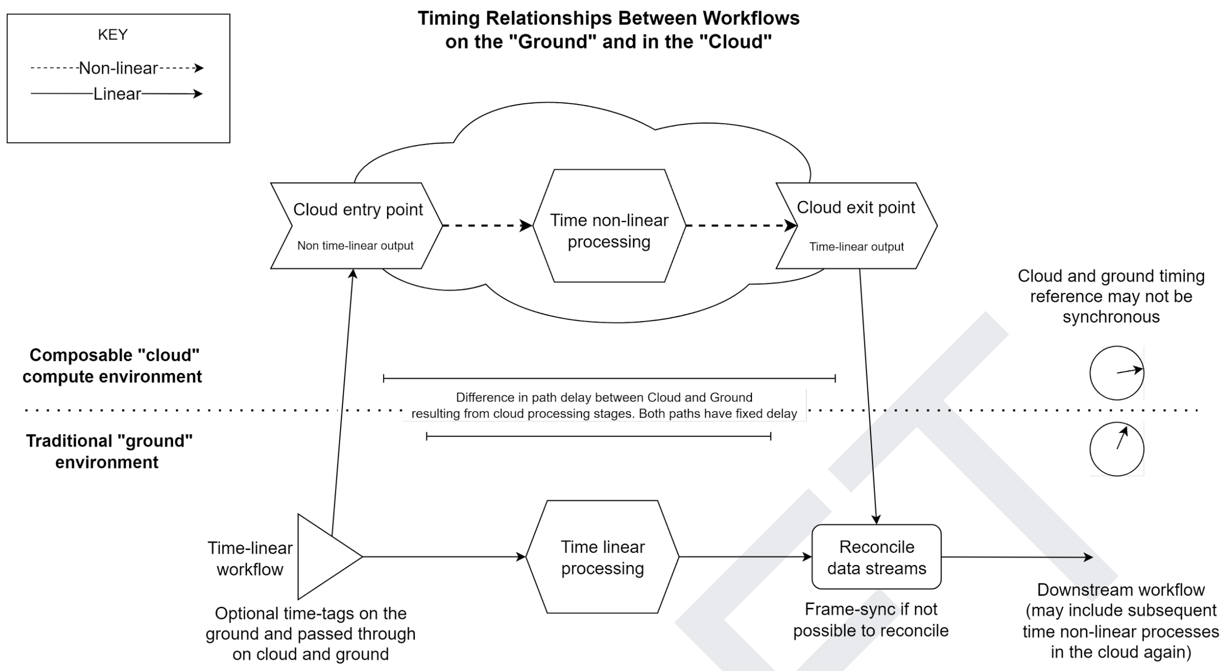
This model is based on a Push model. A Media Flow (of Media Elements) arrives at a Workflow Step based on its Flow Timebase. Different Media Flows may have different timebases, as identified by the GMID carried by the Media Elements. Timestamps of output Media Elements must comply with the advertised timing properties of the Workflow Step.

Timebase signaling in this recommendation is explicit. Even a Workflow Step with no inputs (like a clip player) must signal its output's timebase. This timebase can be referenced to the Local Timebase (LT) or the sender can accept an input Timing Reference Flow which gives it a sense of frequency and time for providing its output.

As an example, consider the storage-centric architecture concept: an ISO camera output goes up to the cloud for ingest and storage. Later a replay operator wants to roll a clip in a production. The ingest function will run as the clip arrives, encoding and writing to storage based on the Original Timebase. However, the playback function should be frequency referenced to the production's timebase. This is accomplished by explicit reference. A Timing Reference Flow can be sent to the playback engines to give them the production timebase.

7.2. Timing Relationship Between Ground and Cloud

Ground-based Workflows are typically based on broadcast plant practices - GPS-derived or otherwise traceable timebases are the goal, but not always the fact. The input flows arriving at the cloud entry point can signal the timebase they come from, but there is no presuming that those timebases are locked to each other or to any traceable source.
Within the cloud environment a service providing a Local Timebase is assumed to exist, giving a consistent sense of time to each of the Workflow Steps. The Local Timebase has no assumed relationship to the timebases of any of the incoming signals, except for a general expectation that all of these timebases are within 50PPM (Parts Per Million) of traceable time. This bound on the frequency difference allows for some simplifications in the handling of timestamps.


KEY
Non-linear
Linear
Timing Relationships Between Workflows on the "Ground" and in the "Cloud"
Cloud entry point
Non time-linear output
Time non-linear processing
Cloud exit point
Time-linear output
Cloud and ground timing reference may not be synchronous
Composable "cloud" compute environment
Difference in path delay between Cloud and Ground resulting from cloud processing stages. Both paths have fixed delay
Traditional "ground" environment
Time-linear workflow
Time linear processing
Reconcile data streams
Optional time-tags on the ground and passed through on cloud and ground
Frame-sync if not possible to reconcile
Downstream workflow (may include subsequent time non-linear processes in the cloud again)

7.3. Timing System Requirements for Time-Non-Linear Workflow Steps

In order to achieve the bounded latency objective of the Time-Non-Linear Workflow, it is required to define a timing model and apply it to the Workflow Steps and the Workflow as a whole.

Each Time-Non-Linear Workflow Step is characterized by certain timing parameters:

- **Tmin** - the minimum latency of the Workflow Step, including fully transferring the output(s) onto the fabric of the compute environment. This minimum includes any re-ordering delays required by the application.
- **T99** - the effective maximum latency of the Workflow Step, including fully transferring the output(s). The 99 denotes an expectation that the Workflow Step will fall within this maximum latency 99.9% of the time.

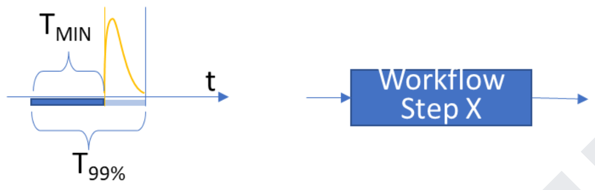
Both of the above are measured from the time when all of the required input(s) have fully arrived. These metrics must be true to the current running platform and configuration and are advertised by the Workflow Step. For any specific Workflow Step *instance* these values are expected to be quasi-constant (for a given set of advertised capabilities).

A Time-Non-Linear Workflow path is composed of a sequence of interconnected steps. The path delay resulting from the sequence of Workflow Steps can be defined as follows:

- The **Tmin** of a Workflow path is the summation of the **Tmin**s along the path
- The **T99** of a Workflow path is the summation of the **T99**s along the path

The topology of the Workflow path is constant (see discussion of switching below). If a step within a path has more than one input, the worst-case parameters of those inputs are used in the path summation above. In addition to advertising its individual **Tmin** and **T99** contributions, each Workflow Step shall advertise the accumulated **Tmin** and **T99** of its output

Advertised Tmin and T99 of a Workflow Step

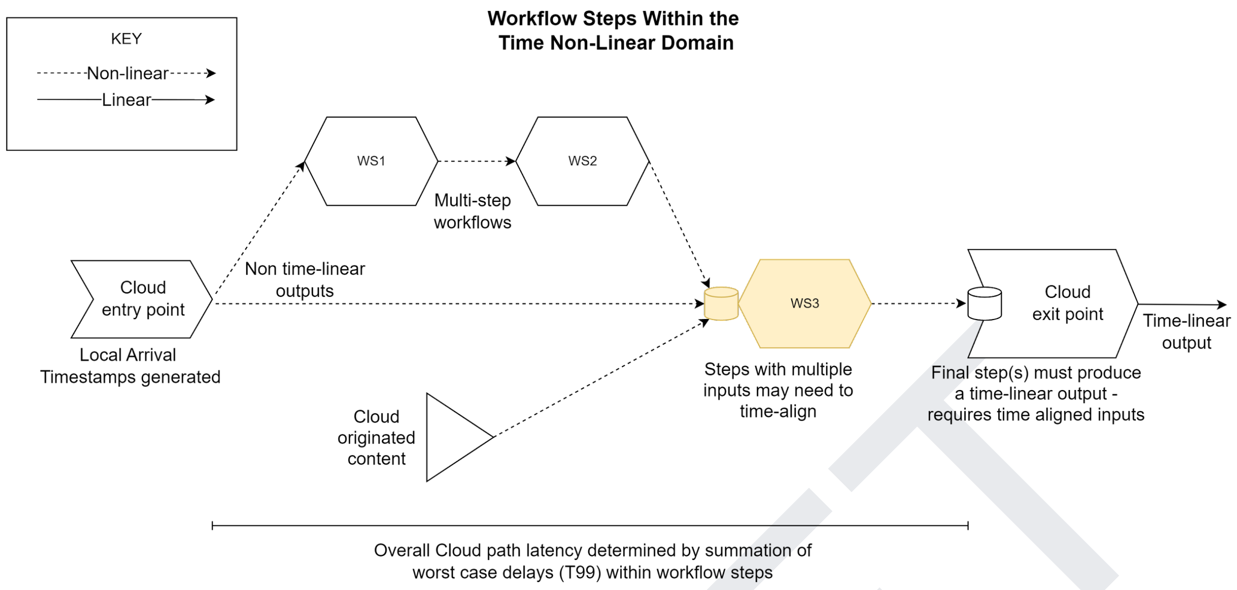


7.4. Routing: Workflow Steps with Multiple and/or Changeable Inputs

A Workflow Step with multiple inputs needs to reconcile the timing between those input signals. Moreover, a Workflow Step with changeable (routable) inputs must deal with differences in input timing before and after a signal is switched.At all times the Workflow Step must also calculate and advertise its prevailing **Tmin** and **T99** values:

- A “**Simple Input**” to a Workflow Step inherits the accumulated Tmin and T99 of its input signal. If a different signal is substituted into this input, the accumulated **Tmin** and **T99** of the new input signal may affect the revised **Tmin** and **T99** of the Workflow Step outputs, and may disrupt the workflow. Simple inputs are not expected to switch cleanly between signals.

- A “**Planned Switching Input**” to a Workflow Step allows for a *restricted amount* of variation in the Tmin and T99 of the “previous” and “next” signals which may be switched at runtime, while preserving a common output timing model.
    - The budget for variation of Tmin and T99 is a configuration parameter
    - The “previous” and “next” signals must share a common Flow Timebase
    - The Planned Switching Input is constructed to facilitate switching while creating a continuous signal - in particular it may include facilities to acquire the “next” input signal before dropping the “previous” one, or it may employ other methods (frame repeats, etc) to maintain a continuous signal during the transition.
    - The **Tmin** and **T99** of this Workflow Step is calculated based on this variation parameter, and if the input variation is within the configured variation budget, then the Tmin and T99 of the output will not change.


Workflow Steps Within the Time Non-Linear Domain
KEY
Non-linear
Linear
WS1
WS2
Multi-step workflows
Cloud entry point
Local Arrival Timestamps generated
Non time-linear outputs
WS3
Steps with multiple inputs may need to time-align
Cloud exit point
Time-linear output
Final step(s) must produce a time-linear output - requires time aligned inputs
Cloud originated content
Overall Cloud path latency determined by summation of worst case delays (T99) within workflow steps

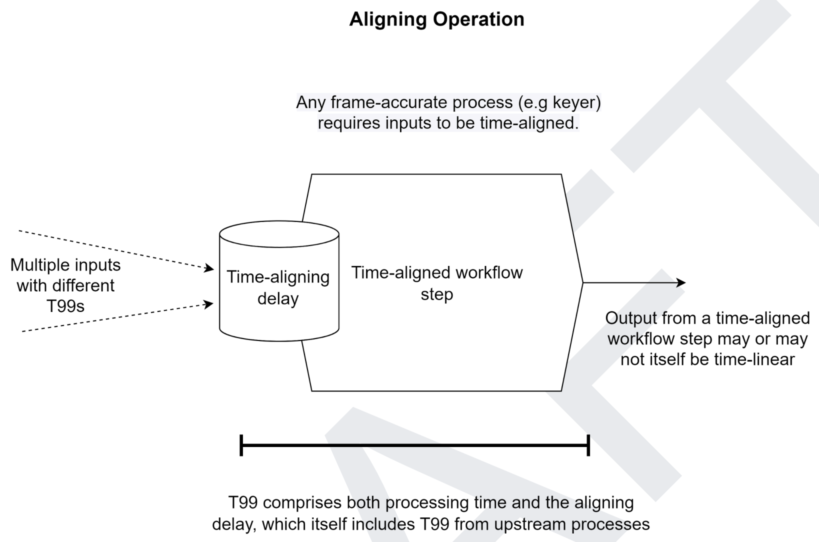
Workflow Steps may use a combination of timing operations in order to reconcile, preserve or recover the desired timing for the signals to be processed and likewise for the output signal(s) that will result from the Workflow Step.

7.5. Timing Operations

A **Restamping Operation** occurs when a Workflow Step composes a new output signal (perhaps derived from one or more inputs) and for production reasons stamps it with current time as the Content Origination Timestamp (and similarly restamps the Local Arrival Timestamp).

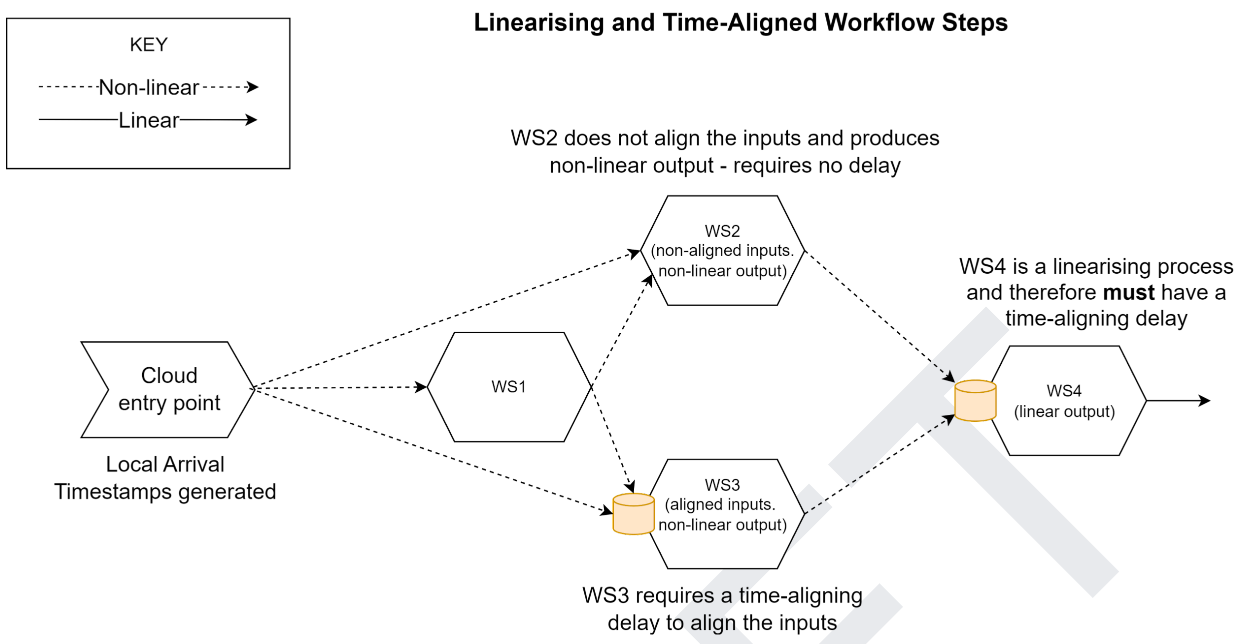
A **Retiming Operation** converts a flow of Media Elements into a new flow with a different timebase (e.g by frame skip/repeat, audio resampling, or other similar techniques). Often this would involve conversion from the origin timebase to the local timebase, or from the Local Timebase to match the origin timebase of another signal. Note that this operation does not necessarily require full linearization, however it may require Temporal Conformance to meet a specified T99 value at the output, such that it can meet the requirements of a Planned Switching input downstream. The output of this operation has new Local Arrival Timestamps (synthetically created) which reflect when the signal would have arrived if it had been in the new timebase on its arrival. The Content Original Timestamps should be retained through this operation as faithfully as possible, but accounting for any frame skips/repeats or additional/removed audio samples, with an emphasis on continuity of the output signal to limit downstream disruptions.

An **Aligning Operation** is required where a Workflow Step combines more than one input flow (like a graphic keyer or video transition or audio mixer) and needs to apply the operation to the contemporaneous Media Elements from each input. Alignment requires that the input flows share the same Flow Timebase, which may be accomplished through a Retiming Operation upstream. The Aligning Operation involves waiting until the required Media Elements have all arrived, but does not imply a Time-Linear output will be produced. Alignment can take place as soon as the required Media Elements are available, meaning the Workflow Step can complete its processing ahead of schedule. The aligning delay must be sufficient to accommodate the worst-case difference in T99 between the inputs.


Aligning Operation
Any frame-accurate process (e.g keyer) requires inputs to be time-aligned.
Multiple inputs with different T99s
Time-aligning delay
Time-aligned workflow step
Output from a time-aligned workflow step may or may not itself be time-linear
T99 comprises both processing time and the aligning delay, which itself includes T99 from upstream processes

A **Linearizing Operation** takes a Time-Non-Linear input signal and creates a Time-Linear output signal (internally). A delay (buffer) is required in order to ensure the continuous (smooth) availability of signal into the Time-Linear domain.

- The linearizing operation requires determination of a point-in-time for removing each Media Element from the linearising buffer and these removals must proceed in a Time-Linear manner by definition.
- A successful linearising operation requires that the media item is always available when it is time to remove it from the buffer
- The minimum delay of the Linearizing operation is determined by the T99 accumulated along its input paths, plus an accommodation for internal processing and additional production delays.


Linearising and Time-Aligned Workflow Steps
KEY
Non-linear
Linear
WS2 does not align the inputs and produces non-linear output - requires no delay
WS2 (non-aligned inputs. non-linear output)
WS4 is a linearising process and therefore **must** have a time-aligning delay
Cloud entry point
WS1
WS4 (linear output)
Local Arrival Timestamps generated
WS3 (aligned inputs. non-linear output)
WS3 requires a time-aligning delay to align the inputs

7.6. Time-Aligning and Linearising Workflow Steps

A Non-Linearising Workflow Step shall output each Media Element as soon as the Media Element is available for output (but never sooner than the Local Arrival Timestamp - clip players in particular cannot play ahead).

Any Time-aligning (or Linearising) Workflow Step must consider the timing properties of its inputs in order to determine the required aligning delay:

Max_Tmin = MAX(Tmin_1, … Tmin_N)
Max_T99 = MAX(T99_1, … T99_N)

where N is the number of inputs to the Workflow Step.

A Time-aligning Workflow Step cannot output a Media Element until all required input Media Elements have arrived (this will be at least Max_Tmin) and been processed. It can then output each Media Element as soon as it is ready to be output. If a Media Element has not arrived on time (within its T99 window) the time-aligning Workflow Step shall take an implementation-specific action attempting to preserve the output signal cadence and timing integrity.

A Linearising Workflow Step buffers its input Media Elements in order to create a regular cadence of output. In particular, it shall wait to output each Media Element at:

T_drain = Local_arrival_timestamp + Max_T99

If a Media Element has not arrived on time (within its T99 window) the linearizing Workflow Step shall take an implementation-specific action attempting to preserve the output signal cadence and timing integrity.

All timestamps in the definitions above are in cloud time and it is assumed that there is only modest (negligible) slippage between Original Timebases and Local Timebases for the purposes of these definitions. In addition, Linearizing workflows steps shall also allow specification (through the management API) of an additional timing margin to accommodate network-induced delivery delay and accumulated timing error.

7.7. Entry Aspects - from Linear Transport to Non-Linear processing

When a flow arrives at the edge of the cloud, the model must determine a Content Original Timestamp (COT) for each Media Element. While as an industry we aspire to transport origin timestamps with the essence, in practice the model must have a consistent way to estimate the COT if they are not present in the transport.

- In the case that origin timing is not available in the arriving signal, let the COT be derived as a linear “sample” of the estimated Original Timebase, modeling the slew rate (frequency difference) between Original Timebase and the Local Timebase from the Local Timebase samples at each Media Element arrival event (emulating a constant frame rate and accounting for any VBV-like variations and long-gop re-ordering). This type of PCR-aware data-directed time estimation can be fairly accurate, and the error is bounded by the 50ppm assumption above. These timestamps can be modified by an operationally-determined constant which reflects the path delays upstream.

- In the case of 2110-xx incoming, if the timestamps have the meanings ascribed in ST 2110-10:2022, we can use the RTP timestamp to infer the origination time. These timestamps may again need to be offset by a constant determined by some knowledge of the accumulated delays.

- In some cases a timecode signal (ATC in ST2038 or ST2110-40, SEI in h.264, etc.) is available within the incoming signal. In these cases, that timing information modified by a production-determined constant, could be used as the basis for the origin timestamps.

8. The GCCG API Design

8.1. General

This Technical Recommendation includes by reference an API definition on the VSF github repository https://github.com/vsf-tv/gccg-api which provides a consistent way for applications to be built in a manner independent of the target computing environment's implementation of this recommendation. It is assumed that *within a computing environment* (within a single cloud provider for example) Workflow Steps using the API can send and receive Media Flows to/from each other.

Connections between Workflow Steps are managed using the Connection Management API defined in AMWA IS-05, with Transport Parameters specified by the underlying implementation (may be different in different computing environments) and Media Parameters as defined below.

Certain parameters of a Sender (and its related Flow) are conveyed to the Receiver and are fixed for life of the connection

- Tmin, T99 parameters of the signal being transported (can change but the change is disruptive to the Workflow timing while the Workflow adapts.)
- Transport Parameters of the Sender (required for the connection management interface)
- Structural aspects of the Source and Flow, including which types of content *could* be present in the multiplex flow (whats potentially in the multiplex, even if a given transfer does not include it all) and what transfer format is used (video packing, etc)

In addition to the Connection-level parameters above, each Media Element conveys certain Data-Package-Level information alongside the media information of the Media Element. Should there be any inconsistency between the information conveyed in the flow, in comparison to the data conveyed in the connection request, the data from the flow shall be considered correct.

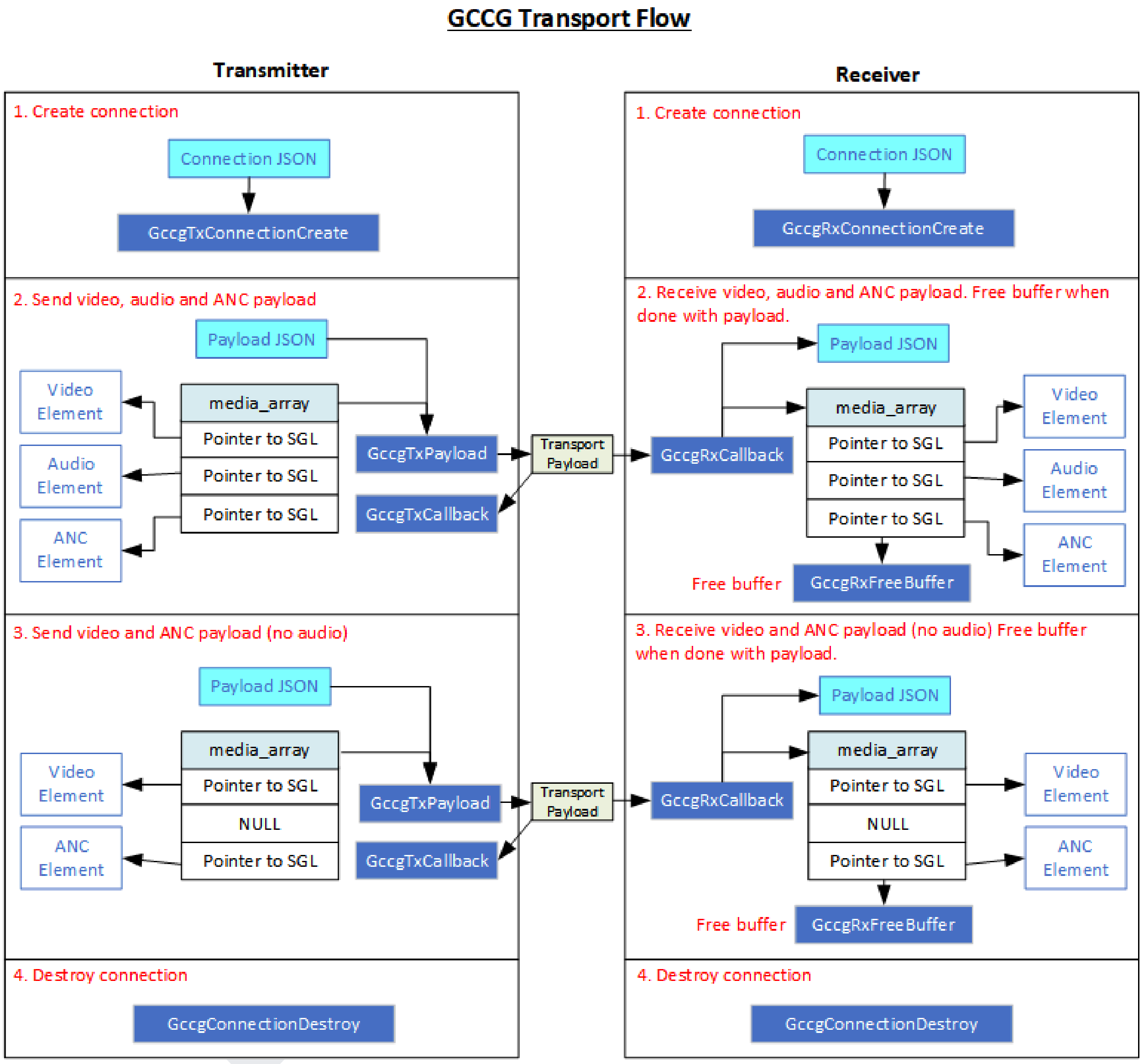
- Header information (GMID,Tmin, T99)
- Timestamp information (COT, LAT)
- Structural information about the multiplex (which elements are present)
- Multiplex of some combination of Video + audio + data
    - Optional Video (in the specified packing format)
    - Optional Audio (in the specified packing format)
    - Optional ANC (in the specified packing format)

The authoritative API specification is found at https://github.com/vsf-tv/gccg-api/tree/main while informative annex A of this recommendation is provided for convenience.

In addition to the send and receive hooks of the API, the application also requires calls and responses for integration of the application and support of the NMOS interactions.

- On a sender, the API must return to the application the (environment-specific) transport parameters JSON object to be incorporated into the connection management NMOS advertisement.
- On a receiver, the API needs to accept an environment-specific JSON object of transport parameters coming from the IS-05 connection management API exposure.

- On a sender, the typical egress latency of the signal (the egress contribution to the Tmin) must be made available to the application



8.2. Media Content Structure at the API

The underlying transport mechanism between Media Workflow Steps is specific to the environment, and could be as simple as a buffer transfer over RDMA or similar memory-transfer techniques. In order to promote portability of the API, the Content Structure (the exact definition of the buffer transferred) is defined explicitly in the Github repository and summarized below:

- Media Element header including COT, LAT, GMID, Tmin, T99, and structural information (which essences are present, etc)
- Video Buffer:
  - A header describing
    - origin (within the frame) and dimensions of the pixel array that follows (for example (0,0) and (1920,1080) for a whole 1080p video frame). For interlaced signals, the buffer is composed of two fields, and the conferred dimensions are relative to the current field.
    - Format Information (uncompressed, compressed, how compressed)
    - Structural information of the video (sampling, bit depth, etc). This information leverages the FMTP parameters defined in SMPTE ST2110-20 for video signals, and thus inherits its minor constraints on image size and sampling structures.
    - Interpretive information of the pixel data (colorimetry, TCS, RANGE, etc) are also conveyed in a parametric manner inherited from ST 2110-20.
  - Format-specific video blob
    - Uncompressed case (default) A concatenation of pgroup structures as defined in SMPTE ST 2110-20, covering the prescribed rectangular space.
    - Compressed cases: IANA types are used to indicate the content format. video/raw for uncompressed, video/jxsv for jpeg-xs, etc.
- Audio Buffer
  - A header describing
    - The total number of channels being transported (fixed for life of connection)
    - The packaging of the channels (ST2110-31 style AM824, or 32-bit PCM)
    - The number of active channels within that total (could vary at runtime, but cannot exceed the total number being transported)
    - Channel order information (2110-30 channel ordering convention syntax)
    - Sampling rate (fixed for life of connection)
    - the number of samples included for each channel (could vary slightly due to fractional frame rates), and the original bit depth of the samples
  - A buffer of 32-bit unsigned integers, with one audio sample MSB-justified into each word. The buffer will be N-total-channels * N-samples in size. In the case of audio, the timestamps refer to the time of the first sample of the buffer.
- ANC data
  - A header describing the contents
    - The number ANC packets being transported
    - If there is no ANC data to be transmitted in a given period, the header shall still be sent in a timely manner indicating a count of zero.
    - The type of video (interlaced or progressive) and the field if interlaced
  - A packet header based on the packing model of RFC 8331
    - Whether the ANC data corresponds to the luma (Y) data channel or not

        - The interface line number of the ANC data (in cases where legacy location is not required, users are encouraged to use the location-free indicators specified in RFC8331)
        - DID and SDID
        - The number of data words for each ANC element
        - Note that in this specification we have removed the horizontal offset and stream number which are present in the RFC.
    - Packet data based on the packing model of RFC 8331

8.3. Regarding NMOS Integration

The application shall expose an IS-04 Node API and IS-05 connection management API. Within the IS-04 advertisement of resources, Receivers should declare their capabilities using the methods described in BCP-004-01. If an incoming Media Flow is outside the range of advertised capabilities, the Workflow Step can fail to receive it.

From an NMOS perspective, this API generates and consumes flows of format “urn:x-nmos:format:mux” – multiplexed flows which may include video, audio, and ANC data. Even if a given flow only contains a single component, in this specification it will be declared as a multiplex flow because the structure could contain multiple essence types. Note that the grain rate of multiplexed flows in NMOS is by rule the grain rate of the video, even if there might be no video present. In general the method of NMOS integration for this API is consistent with that proposed in AMWA BCP-007-1 where possible.

Each different implementation must register a URN for its specific transport format, used in the “transport” key of the IS-04 sender and receiver declarations. The underlying implementation also defines a transport parameters structure specific to the implementation - this structure is indicated by the transport URN. In order to isolate the application from differing transport formats, the transport parameters structure shall be conveyed as a JSON-formatted object through the API and exposed in the related NMOS IS-05 sender and receiver structures. It is the responsibility of the implementation provider to register a JSON schema and URN for its transport parameters object format in the AMWA NMOS Parameter Register.

The API defines a JSON structure based on the FMTP parameters defined in SMPTE ST 2110-20, 2110-30, and 2110-40. This JSON structure shall be used in the /transportfile endpoint for IS-05 senders, and within the transport_file object for IS-05 receiver /staged and /active endpoints.

Since the transport is a multiplexed transport, the JSON structure includes an array of JSON objects, where each object documents a specific essence element. The ordering of the items in the array is significant. On IS-05 receivers within the transport_file structure, the type "application/x-vsf-gccg" will be used to indicate that the "data" item is the JSON structure above. The JSON structure also includes the GMID, Tmin, and T99 parameters indicating the timing of the related essences.

9. Profiles and Levels

9.1. Profiles

Profiles and Levels are a market-oriented construct to provide vendors with an ability to simply indicate that a product supports a certain range of capabilities. Note that these are similar in concept to the classic MPEG profiles and levels, but are not strictly aligned to them. Support of a specific level *does not* imply support for levels below it – products should list explicitly the profile and level combinations they support in order to avoid misunderstandings in the field.

- Main Profile
    - Video
        - Transfers in pgroup structures as in 2110
        - Support 4:2:2/10
        - Support 4:2:2/8 – yes (no objections)
        - Support 4:2:0/8 – yes (no objections)
        - Frame sizes and rates according to level
    - Audio
        - Up to 16 channels, 24bits 48khz,
        - Normally packaged as PCM samples (as noted above)
        - Optionally packed as in 2110-31 including status bits (need ref)
    - ANC
        - Up to 1748 10-bit words per frame
- DCI Profile
    - Video 4:2:2/10 or 4:4:4/12 (are there others in practice?)
    - Audio up to 32 channels, 24 bits, 48khz
    - ANC (as in main profile)

9.2. Levels

- **Level 0 (@main profile)**
    - Name: SD
    - Pixel Rate (720 x 480 x 30Hz x 20BPP) = 207360000 bits/sec max
    - Pixel Rate (720 x 576 x 25Hz x 20BPP) = 207360000 bits/sec max
    - Max Image Size: 720(w) x 576(h)
    - Max Frame Rate: 30 (60 interlaced fields)
    - Max Bit Depth: 10
    - Max Bits/Image Sample: 20

- **Level 1-HDA (@main profile)**

- Name: HDA
- Pixel Rate (1920 x 1080 x 30Hz x 20BPP) = 1,244,160,000
- Max Image Size: 1920x1080
- Max Frame Rate: 30
- 2:1 Interlacing is supported
- Max Bit Depth: 10
- Max Bits/Image Sample: 20

- **Level 1-HDB (@main profile)**
  - Name: HDB
  - Pixel Rate (1280 x 720 x 60Hz x 20BPP) = 1,105,920
  - Max Image Size: 1280x720
  - Max Frame Rate: 60
  - Interlacing is not supported
  - Max Bit Depth: 10
  - Max Bits/Image Sample: 20

- **Level 1-HD – indicates support for both HDA and HDB**

- **Level 2 - 1080p (@main profile)**
  - Name: 1080p
  - Pixel Rate (1920 x 1080 x 60Hz x 20BPP) = 2,488,320,000
  - Max Image Size: 1920x1080
  - Max Frame Rate: 60
  - Max Bit Depth: 10
  - Max Bits/Image Sample: 20

- **Level 2+ - 1080p (@main profile)**
  - Name: 1080p+
  - Pixel Rate (1920 x 1080 x 60Hz x 36BPP) = 4,478,976,000
  - Max Image Size: 1920x1080
  - Max Frame Rate: 60
  - Max Bit Depth: 12
  - Max Bits/Image Sample: 36 (4:4:4)

- **Level 3 UHD (@main profile)**
  - Name: UHD
  - Pixel Rate: (3840 x 2160 x 60 x 20BPP) = 9,952,280,000
  - Max Image Size: 3840x2160
  - Max Frame Rate: 60
  - Max Bit Depth: 10
  - Max Bits/Image Sample: 20 (4:2:2)

- **Level 3+ UHD (@plus profile)**

- Name: UHD+
- Pixel Rate: (3840 x 2160 x 60 x 36BPP) = 17,915,904,000
- Max Image Size: 3840x2160
- Max Frame Rate: 60
- Max Bit Depth: 12
- Max Bits/Image Sample: 36 (4:4:4)

- **Level 3C UHD (@DCI Profile)**
  - Name: UHD+DCI
  - Pixel Rate: (4096 x 2160 x 24 x 36BPP) = 7,644,119,040
  - Max Image Size: 4096x2160
  - Max Frame Rate: TBD
  - Max Bit Depth: 12
  - Max Bits/Image Sample: 444@12

- **Level 4 UHD2 (@main profile)**
  - Name: UHD2
  - Pixel Rate: (7680 x 4320 x 60 x 20BPP) = 39,813,120,000
  - Max Image Size: 7680x4320
  - Max Frame Rate: 60
  - Max Bit Depth: 10
  - Max Bits/Image Sample: 20

Note that the Profiles and Levels above are not limitations, they are simply marketing-useful terms to describe minimum feature sets of products in the marketplace. Products which surpass these profiles and levels are welcome in the marketplace. For example, High-Frame-Rate (HFR) and unusual aspect ratios are supported in this specification, but may be beyond the profiles and levels described above.

10. Platform Expectations/Requirements (what can the application expect to find in the environment)

In addition to the API described above, applications can expect to find a time service which exposes a TAI-enumerated timebase whose frequency is within 50ppm accuracy relative to traceable time. The phase accuracy of the timebase relative to traceable time shall be within 100ms (plus or minus). Time service implementations should ensure that successive reads of the timebase increase under all circumstances (time does not go backwards) except under exceptional circumstances. Irrespective of the above, application developers are encouraged to be aware of the exceptional circumstances that might occur during startup or re-locking events.