ISO/IEC JTC 1/SC 29/WG 03 N0630

**ISO/IEC JTC 1/SC 29/WG 03  
MPEG Systems  
Convenorship: KATS (Korea, Republic of)**

**Document type:** Output Document

**Title:** WD of ISO/IEC 23009-1 5th edition AMD 2 EDRAP streaming and other extensions

**Status:** Approved

**Date of document:** 2022-07-22

**Source:** ISO/IEC JTC 1/SC 29/WG 03

**No. of pages:** 16 (with cover page)

**Email of Convenor:** young.L @ samsung . com

**Committee URL:** <https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg3>

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 03 MPEG SYSTEMS**

**ISO/IEC JTC 1/SC 29/WG 03 N0630**

**July 2022, Virtual**

|  |  |
| --- | --- |
| **Title** | **WD of ISO/IEC 23009-1 5th edition AMD 2 EDRAP streaming and other extensions** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **21759** |

# Change 1: EDRAP Streaming

*In clause 2, add the following reference:*

*ISO/IEC 14496‑12:2021 AMD1[[1]](#footnote-1), Information technology — Coding of audio-visual objects — Part 12: ISO base media file format, AMD 1 Improved brand documentation and other improvements [Ed. Note (YK): The latest draft text of this amendment as of Apr. 2022 is in WG 03 output document N0550.]*

*In subclause 3.2, add the following abbreviations:*

|  |  |
| --- | --- |
| EDRAP | extended dependent random access point |
| ESR | external stream Representation |
| MSR | main stream Representation |

*Add subclause 5.8.5.15 as follows:*

***5.8.5.15 MSR and ESR descriptors***

**5.8.5.15.1 General**

An Adaptation Set may have an **EssentialProperty** descriptor with @schemeIdUri equal to urn:mpeg:dash:msr:2022. This descriptor is referred to as the MSR descriptor. This descriptor may only be present in an Adaptation Set level and its presence indicates that each Representation in that Adaptation Set is an MSR, which carries a video track containing a track reference of type 'aest' as specified in ISO/IEC 14496 12:2021 AMD1.

An Adaptation Set may have an **EssentialProperty** descriptor with @schemeIdUri equal to urn:mpeg:dash:esr:2022. This descriptor is referred to as the ESR descriptor. This descriptor may only be present in an Adaptation Set level and its presence indicates that each Representation in the Adaptation Set is an ESR, which carries a video track referenced by a track reference of type 'aest' as specified in ISO/IEC 14496 12:2021 AMD1. An ESR is only intended to be consumed or played back together with its associated MSR.

Each ESR shall be associated with an MSR through the Representation-level attributes @associationId and @associationType in the MSR as follows: the @id of the associated ESR shall be referred to by a value contained in the attribute @associationId for which the corresponding value in the attribute @associationType is equal to 'aest'. Each MSR shall have an associated ESR.

For an MSR and an ESR associated with each other, the following applies:

* For each media sample with a particular presentation time in the ESR, there shall be a corresponding media sample with the same presentation time in the MSR.
* Each media sample in the MSR that has a corresponding ESR media sample is referred to as an EDRAP sample.
* The first byte position of each EDRAP sample in the MSR shall be the ISAU of a SAP, which enables playback of the media stream in the MSR provided that the corresponding ESR media sample is provided to the media decoder immediately before the EDRAP sample.
* Each EDRAP sample in the MSR shall be the first sample in a Segment or Subsegment (i.e., each EDRAP sample shall start a Segment or Subsegment).
* For each Segment or Subsegment in the MSR that starts with an EDRAP sample, there shall be a Segment in the ESR with the same earliest presentation time as the MSR Segment or Subsegment. This Segment in the ESR is referred to as the corresponding ESR Segment of the MSR Segment or Subsegment and vice versa.
* The concatenation of any Segment in the ESR and the corresponding MSR Segment or Subsegment (i.e., the MSR Segment or Subsegment having the same earliest presentation time as the ESR Segment) and all subsequent MSR Segments or Subsegments shall result in a conforming bitstream.
* For each MSR Segment or Subsegment that does not start with an EDRAP sample, there shall be no corresponding ESR Segment having the same earliest presentation time as the MSR Segment or Subsegment.

**5.8.5.15.2 Example content preparation and client operations (informative)**

Below are example content preparation and client operations based on MSRs and their associated ESRs.

An example of content preparation operations is as follows:

1. A video content is encoded into one or more representations, each of which is of a particular spatial resolution, temporal resolution, and quality.
2. Each representation of the video content is represented by a pair of MSR and ESR associated with each other.
3. The MSRs of the video content are included in one Adaptation Set. The ESRs of the video content are included in another Adaptation Set.

An example of client operations is as follows:

1. A client gets the MPD of the Media Presentation, parses the MPD, selects an MSR.
2. When initializing a session or performing seeking, the client determines the starting presentation time from which the content is to be consumed, requests Segments or Subsegments of the MSR, starting from the Segment or Subsegment starting with a SAP and containing the sample having presentation time equal to (or earlier than but close enough to) the determined starting presentation time. For requesting Subsegments in a Segment, a Segment Index is requested beforehand to obtain information of the Subsegments and partial HTTP GET requests are used.
   1. If in the associated ESR there is a Segment having the same earliest presentation time as the starting MSR Segment or Subsegment, that ESR Segment is also requested, preferably before requesting of the starting MSR Segment or Subsegment. Otherwise, no Segment of the associated ESR is requested.
3. When switching to a different MSR, the client requests Segments or Subsegments of the switch-to MSR, starting from the first Segment or Subsegment having earliest presentation time greater than that of the last requested Segment or Subsegment of the switch-from MSR.
   1. If in the associated ESR there is a Segment having the same earliest presentation time as the starting Segment or Subsegment in the switch-to MSR, that ESR Segment is also requested, preferably before requesting of the starting Segment or Subsegment in the switch-to MSR. Otherwise, no Segment of the associated ESR is requested.
4. When continuously requesting and consuming subsequent Segments or Subsegments of an MSR after session initialization, seeking, or stream switching, no Segment of the associated ESR needs to be requested, including when requesting any subsequent MSR Segment or Subsegment starting with an EDRAP sample.

As can be seen from the above example client operations, the client needs to calculate the earliest presentation times of the MSR Segments and Subsegments as well as of the ESR Segments to figure out whether an MSR Segment or Subsegment has an associated ESR Segment.

# Change 2: Picture-in-picture support

*Add subclause 5.8.5.16 as follows:*

***5.8.5.16 Picture-in-picture descriptor***

A **SupplementalProperty** element with the @schemeIdUri attribute equal to urn:mpeg:dash:pinp:2022 is referred to as a picture-in-picture (PiP) descriptor.

At most one PiP descriptor may be present at Preselection level. The presence of a PiP descriptor in a Preselection indicates that the purpose of the Preselection is for providing a PiP experience.

PiP services offer the ability to include a video with a smaller spatial resolution within a video with a bigger spatial resolution. In this case, the different bitstreams/Representations of the main video are included in the Main Adaptation Set of the Preselection, and the different bitstreams/Representations of a supplementary video, also referred to as PiP video, are included a Partial Adaptation Set of the Preselection.

When a PiP descriptor is present in a Preselection, and the **picInPicInfo**@dataUnitsReplacable attribute is present and equal to 'true', the client may choose to replace the coded video data units representing the target PiP region in the main video with the corresponding coded video data units of the PiP video before sending to the video decoder. This way, separate decoding of the main video and the PiP video can be avoided. For a particular picture in the main video, the corresponding video data units of the PiP video are all the coded video data units in the decoding-time-synchronized sample in the supplemental video Representation.

The @value attribute of the PiP descriptor shall not be present. The PiP descriptor shall include a **picInPicInfo** element with its attributes as specified in Table 37.

**Table 37 — Semantics of picInPicInfo element**

| **Element or Attribute Name** | | | | **Use** | **Description** |
| --- | --- | --- | --- | --- | --- |
|  |  | **PicInpicInfo** | |  |  |
|  |  |  | @dataUnitsReplacable | OD  Default: 'false' | specifies whether the coded video data units representing the target PiP region in the main video can be replaced by the corresponding video data units of the PiP video.  When @dataUnitsReplacable is equal to 'true', the client may choose to replace the coded video data units representing the target PiP region in the main video with the corresponding coded video data units of the PiP video before sending to the video decoder for decoding. |
|  |  |  | @regionIds | O | specifies the IDs of the coded video data units representing the target PiP region, as a white space separated list.  When @dataUnitsReplacable is equal to 'false', this attribute shall not be present.  The concrete semantics of the region IDs need to be explicitly specified for specific video codecs. |
| **Key**  For attributes: M=mandatory, O=Optional, OD=optional with default value, CM=conditionally mandatory  For elements: <minOccurs>..<maxOccurs> (N=unbounded)  Elements are **bold**; attributes are non-bold and preceded with an @. | | | | | |

The XML syntax of the **PicInpicInfo** element is as follows:

<xs:complexType name="PicInpicInfoType">

<xs:annotation>

<xs:documentation xml:lang="en">

**Picnpic**

</xs:documentation>

</xs:annotation>

<xs:attribute name="dataUnitReplacable" type="xs:boolean" default="false" />

<xs:attribute name="regionIds" type="StringNoWhitespaceType"/>

<xs:anyAttribute namespace="##other" processContents="lax"/>

</xs:complexType>

Editors Note: An alternative solution for picture-in-picture support in DASH is to use the 'pip' value for **Role**, as well as to use **ContentComponent** along with **Role** and @tag to signal the subpicture IDs or any other IDs needed, as described in m58924, for which some discussions of the details are available herein: <http://mpegx.int-evry.fr/software/MPEG/Systems/DASH/spec/-/issues/265>.

# Change 3: Adding dispatch mode and status to Event

*Replace Table 38s and 39 with the following tables:*

Table 38 — Event Stream Semantics

| **Element or Attribute Name** | | | **Use** | **Description** |
| --- | --- | --- | --- | --- |
|  | EventStream | |  | specifies event Stream |
|  |  | @xlink:href | O | specifies a reference to an external EventStream element |
|  |  | @xlink:actuate | OD  default: onRequest | specifies the processing instructions, which can be either "onLoad" or "onRequest".  This attribute shall not be present if the @xlink:href attribute is not present. |
|  |  | @schemeIdUri | M | identifies the message scheme. The string may use URN or URL syntax. When a URL is used, it is recommended to also contain a month-date in the form mmyyyy; the assignment of the URL must have been authorized by the owner of the domain name in that URL on or very close to that date. A URL may resolve to an Internet location, and a location that does resolve may store a specification of the message scheme. |
|  |  | @value | O | specifies the value for the event stream element. The value space and semantics must be defined by the owners of the scheme identified in the @schemeIdUri attribute. |
|  |  | @timescale | O | specifies the timescale in units per seconds to be used for the derivation of different real-time duration values in the Event elements.  If not present on any level, it shall be set to 1. |
|  |  | @presentationTimeOffset | OD  Default: 0 | specifies the presentation time offset of this Event Stream that aligns with the start of the Period. Any Event contained in this Event Stream is mapped to the Period timeline by using the Event presentation time subtracted by the value of the presentation time offset.  This adjustment shall not be applied to Inband event message streams..  The value of the presentation time offset in seconds is the division of the value of this attribute and the value of the @timescale attribute. |
|  |  | Event | 0 ... N | specifies one event. For details see Table 39.  Events in Event Streams shall be ordered such that their presentation time is non-decreasing. |
| **Key**  For attributes: M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory  For elements: <minOccurs>...<maxOccurs> (N=unbounded)  Elements are bold; attributes are non-bold and preceded with an @. | | | | |

Table 39 — Event Semantics

| **Element or Attribute Name** | | | | | **Use** | **Description** |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Event | |  | specifies an Event and contains the message of the event. The content of this element depends on the event scheme. The contents shall be either:   * A string, optionally encoded as specified by @contentEncoding * XML content using elements external to the MPD namespace   For new event schemes string content should be used, making use of Base 64 encoding if needed.  NOTE The schema allows “mixed” content within this element however only string data or XML elements are permitted by the above options, not a combination. |
|  |  |  |  | @presentationTime | OD default: 0 | specifies the presentation time of the event relative to the start of the Period taking into account the @presentationTimeOffset of the Event Stream, if present.  The value of the presentation time in seconds is the division of the value of this attribute and the value of the @timescale attribute.  If not present, the value of the presentation time is 0. |
|  |  |  |  | @duration | O | specifies the presentation duration of the Event.  The value of the duration in seconds is the division of the value of this attribute and the value of the  @timescale attribute.  The interpretation of the value of this attribute is defined by the scheme owner.  If not present, the value of the duration is unknown. |
|  |  |  |  | @id | O | specifies an identifier for this instance of the event. Events with equivalent content and attribute values in the Event element shall have the same value for this attribute.  The scope of the @id for each Event is with the same @schemeIdURI and @value pair. |
|  |  |  |  | @status | O  default: none | specifies the status of event:   * none: no specific status * update: the event is an update of another event with identical values of @schemeIdUr, @value, and @id |
|  |  |  |  | @contentEncoding | O | specifies whether the information in the body and the information in the @messageData is encoded.  If present, the following value is possible:   * base64 the content is encoded as described in IETF RFC 4648 prior to adding it to the field.   If this attribute is present, the DASH Client is expected to decode the message data and only provide the decoded message to the application. |
|  |  |  |  | @messageData | O | specifies the value for the event stream element. The value space and semantics must be defined by the owners of the scheme identified in the @schemeIdUri attribute.  NOTE the use of the message data is discouraged by content authors, it is only maintained for the purpose of backward-compatibility. Including the message in the Event element is recommended in preference to using this attribute. This attribute is expected to be deprecated in the future editions of this document. |
| **Key**  For attributes: M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory  For elements: <minOccurs>...<maxOccurs> (N=unbounded)  Elements are bold; attributes are non-bold and preceded with an @. | | | | | | |

An **Event** with @status=’update’ is the updated instance of an event with identical @schemeIdUri, @value, and @id attributes that may have been previously processed by the DASH client. The DASH client may replace the previous event with the updated instance if the previous event has not been dispatched yet. An Event with @status=’update’ may differ from the previous event except in the following attributes: @schemeIdUri, @value, and @id.

*Replace 5.10 with the following:*

<xs:complexType name="EventStreamType">

<xs:annotation>

<xs:documentation xml:lang="en">

**Event Stream**

</xs:documentation>

</xs:annotation>

<xs:sequence>

<xs:element name="Event" type="EventType" minOccurs="0" maxOccurs="unbounded"/>

<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

<xs:attribute ref="xlink:href"/>

<xs:attribute ref="xlink:actuate" default="onRequest"/>

<xs:attribute ref="xlink:type" fixed="simple"/>

<xs:attribute ref="xlink:show" fixed="embed"/>

<xs:attribute name="schemeIdUri" type="xs:anyURI" use="required"/>

<xs:attribute name="value" type="xs:string"/>

<xs:attribute name="timescale" type="xs:unsignedInt"/>

<xs:attribute name="presentationTimeOffset" type="xs:unsignedLong" default="0"/>

</xs:complexType>

<xs:complexType name="EventType" mixed="true">

<xs:annotation>

<xs:documentation xml:lang="en">

**Event**

</xs:documentation>

</xs:annotation>

<xs:sequence>

<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>

</xs:sequence>

<xs:attribute name="presentationTime" type="xs:unsignedLong" default="0"/>

<xs:attribute name="duration" type="xs:unsignedLong"/>

<xs:attribute name="id" type="xs:unsignedInt"/>

<xs:attribute name="status" type="xs:StatusType"/>

<xs:attribute name="contentEncoding" type="ContentEncodingType"/>

<xs:attribute name="messageData" type="xs:string">

<xs:annotation>

<xs:documentation xml:lang="en">

**Deprecated in favor of carrying the message information in the**

**value space of the event**

</xs:documentation>

</xs:annotation>

</xs:attribute>

<xs:anyAttribute namespace="##other" processContents="lax"/>

</xs:complexType>

<xs:simpleType name="StatusType">

<xs:annotation>

<xs:documentation xml:lang="en">

**Event Status**

</xs:documentation>

</xs:annotation>

<xs:restriction base="xs:string">

<xs:enumeration value="none"/>

<xs:enumeration value="update"/>

</xs:restriction>

</xs:simpleType>

<xs:simpleType name="ContentEncodingType">

<xs:annotation>

<xs:documentation xml:lang="en">

**Event Coding**

</xs:documentation>

</xs:annotation>

<xs:restriction base="xs:string">

<xs:enumeration value="base64"/>

</xs:restriction>

</xs:simpleType>

*Replace 5.10.3.3.3 with the following:*

##### Syntax

aligned(8) class DASHEventMessageBox extends FullBox('emsg', version, flags){

if (version==0) {

string scheme\_id\_uri;

string value;

unsigned int(32) timescale;

unsigned int(32) presentation\_time\_delta;

unsigned int(32) event\_duration;

unsigned int(32) id;

} else if (version==1) {

unsigned int(32) timescale;

unsigned int(64) presentation\_time;

unsigned int(32) event\_duration;

unsigned int(32) id;

string scheme\_id\_uri;

string value;

}

unsigned int(8) message\_data[];

}

*Replace 5.10.3.3.4 with the following:*

— scheme\_id\_uri: is a null-terminated ('C') string in UTF-8 characters that identifies the message scheme. The semantics and syntax of the message\_data[] are defined by the owner of the scheme identified. The string may use URN or URL syntax. When a URL is used, it is recommended to also contain a month-date in the form mmyyyy; the assignment of the URL must have been authorized by the owner of the domain name in that URL on or very close to that date. A URL may resolve to an Internet location, and a location that does resolve may store a specification of the message scheme.

— value: is a null-terminated ('C') string in UTF-8 characters that specifies the value for the event. The value space and semantics must be defined by the owners of the scheme identified in the scheme\_id\_uri field.

— timescale provides the timescale, in ticks per second, for the event duration and presentation\_time\_delta or presentation\_time fields. The value should be identical to the timescale of a track contained in the carrying Segment. Furthermore, the value should be identical for all events in one Event Stream.

— presentation\_time\_delta provides the Media Presentation time delta of the media presentation time of the event and the earliest presentation time in this segment. If the segment index is present, then the earliest presentation time is determined by the field earliest\_presentation\_time of the first 'sidx' box. If the segment index is not present, the earliest presentation time is determined as the earliest presentation time of any access unit in the media segment. The timescale is provided in the timescale field.

— presentation\_time provides the Media Presentation time of the event measured on the Movie timeline, in the timescale provided in the timescale field, and adjusted by **InbandEventStream**@presentationTimeOffset, in the time scale provided by **InbandEventStream**@timescale; the value shall not be less than the earliest presentation time of the carrying Segment.

— event\_duration provides the duration of event in media presentation time. The timescale is indicated in the timescale field. The value 0xFFFFFFFF indicates an unknown duration. The interpretation of this value must be defined by the owner of the event scheme.

— id: a field identifying this instance of the message. The scope of this identifier for each event is with the same scheme\_id\_uri and value pair. Messages with the same id within the scope of the same scheme\_id\_uri and value pair are equivalent , i.e. processing of any one event message box with the same id is sufficient.

— message\_data: body of the message, which fills the remainder of the message box. This may be empty depending on the above information. The syntax and semantics of this field must be defined by the owner of the scheme identified in the scheme\_id\_uri field.

The flags field is specified as follows:

* (flags & 1) equal to 1 indicates that the esmg is an update of another esmg with identical values of scheme\_id\_uri, value and id fields.

An emsg box with flags &1 =1 is the updated instance of an emsg box with identical scheme\_id\_uri, value, and id fields that may have been previously processed by the DASH client. The DASH client may replace the previous event with the updated instance if the previous event has not been dispatched yet. The updated emsg may differ from the previous emsg except in the following fields: scheme\_id\_uri, value, and id.

*Change A.13.8:*

It is assumed that the application is subscribed to a specific event stream identified by a (scheme/value) pair with a specific dispatch\_mode, either on-start or on-receive, as described in subclause A.13.7.

The processing model varies depending on the value of dispatch\_mode.

1. Common process
   1. The DASH Client implements the following process:
      1. The DASH Client sets up a Pending Event Table (PET) for each subscribed scheme\_uri/(value) in the case of dispatch\_mode = on\_start. The PET maintains a single list of event ids that are waiting to be dispatched. The DASH Client also sets up a Dispatched Event Table (DET) for each subscribed scheme\_uri/(value). The DET maintains a single list of 'emsg' ids that have been dispatched.
      2. Parse the 'emsg'/timed metadata sample and retrieve scheme\_uri/(value).
      3. If Application is not subscribed to the scheme\_uri/(value) pair, end the processing of this 'emsg'.
      4. Derive the event instance/metadata sample’s *ST*
      5. Derive the ending time *ET*= *ST* + *DU*.
2. On-receive processing
   1. The DASH Client implements the following process when dispatch\_mode = on\_receive:
      1. If the current presentation time value is greater than *ET*, then end processing.
      2. In the case of event: Compare the event’s *id* with the entries of the DET of the same scheme\_uri/(*value)* pair:
         * If an entry with the identical *id* value exists, end processing;
      3. Dispatch the event/timed metadata, including *ST*, *id*, *DU*, *timescale*, and message\_data as described in subclause A.13.6, and add the event to the DET.
3. on-start processing
   1. DASH ClientThe DASH Client implements the following process when dispatch\_mode = on\_start:
      1. If the event is an update, remove any event, if any, with identical scheme\_uri/(*value)* and *id* from the PET.
      2. Derive the event instance/metadata sample’s *ST*.
      3. If the current media presentation time value is smaller than *ST*, then go to step vi.
      4. Derive the ending time *ET*= *ST* + *DU*.
      5. If the current presentation time value is greater than *ET*, then end the processing.
      6. In the case of event: Compare the event’s *id* with the entries of the PET of the same scheme\_uri/(*value)* pair:
         * If an entry with the identical *id* value exists, end the processing.
         * If not, add 'emsg'’s *id* to the corresponding P[ET](#active-event-table).
      7. Dispatch the event/metadata message\_data at time ST, or immediately if the current presentation time is larger than ST, as described in subclause A.13.6, remove the event, if any exists, from the PET and add it to the DET.

# Change 4: Adding DASH specific events dispatch modes

*Change A.13.10 to:*

**A.13.10 Prose description of APIs**

The event/timed metadata API is an interface defined between a DASH client and a device application in the exchange of subscription data and dispatch/transfer of matching DASH Event or timed metadata information between these entities. The Event/timed metadata API is shown in Figure A.1.

NOTE 1 In this Annex, the term "DASH Player" is used.

The description of the API below is strictly functional, i.e. implementation-agnostic. For example, the subscribeEvent() method as defined below may be mapped to the existing on(type,listener,scope) method as defined for the dash.js under MediaPlayerEvents.

As part of this API and before any operations, the DASH Player provides a list of *scheme\_id*/(*value*) listed in the MPD when it receives it. This list includes all MPD and inband events as well as *scheme\_id* of all timed metadata tracks. At this point, the Application is aware of the possible events and metadata deliverable by the DASH Player.

NOTE 2 The DASH Player may provide the Application the list of DASH event schemes as a part of listed event schemes in the MPD and consequently, the Application may subscribe to one or more of these event schemes.

Change A.13.12 to the following:

**A.13.12 Dispatch modes for DASH-specific events**

In 5.10.4, several DASH-specific event schemes are defined. Table A.4 describes their dispatch modes.

1. Table A.4 DASH-specific event schemes dispatch modes

| **scheme** | **Dispatch mode** |
| --- | --- |
| urn:mpeg:dash:event:2012 | on-receive |
| urn:mpeg:dash:event:callback:2015 | on-start |
| urn:mpeg:dash:event:ttfn:2016 | on-start |
| urn:mpeg:dash:event:period:2020 | On-receive |

1. Under preparation. [↑](#footnote-ref-1)