**** **ISO/IEC JTC 1/SC 29/WG 03 N0339**

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

**Document type:** Output Document

**Title:** WD of ISO/IEC 23009-8 AMD 1 URL customization and other extensions

**Status:** Approved

**Date of document:** 2021-07-23

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

**Expected action:** None

**Action due date:** None

**No. of pages:** 18 (with cover page)

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

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

**INTERNATIONAL ORGANISATION FOR STANDARDISATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC1/SC29/WG 03 MPEG SYSTEMS**

**ISO/IEC JTC 1/SC 29/WG 03 N** **0339**

**July 2021, Virtual**

|  |  |
| --- | --- |
| **Title** | **ISO/IEC 23009-8 CDAM1 URL customization and other extensions** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **20593** |

ISO/IEC JTC 1/SC 29

Date: 2021-07-23

ISO/IEC  23009-8 CDAM1

ISO/IEC JTC 1/SC 29/WG 11

Secretariat:

Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 8: Session-based DASH operations- Amendment 1: URL customization and other extensions

Technologies de l'information — Diffusion en flux adaptatif dynamique (DASH) — Partie 8: Opérations DASH basées sur une session

Copyright notice

This ISO document is a Draft International Standard and is copyright-protected by ISO. Except as permitted under the applicable laws of the user's country, neither this ISO draft nor any extract from it may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, photocopying, recording or otherwise, without prior written permission being secured.

Requests for permission to reproduce should be addressed to either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office

Case postale 56 • CH-1211 Geneva 20

Tel. + 41 22 749 01 11

Fax + 41 22 749 09 47

E-mail copyright@iso.org

Web [www.iso.org](http://www.iso.org)

Reproduction may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see [www.iso.org/directives](https://www.iso.org/directives-and-policies.html)).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see [www.iso.org/patents](https://www.iso.org/patents)).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: [www.iso.org/iso/foreword.html](https://www.iso.org/iso/foreword.html).

This document was prepared by Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

A list of all parts in the ISO/IEC 23009 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at [www.iso.org/members.html](http://www.iso.org/members.html).

Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 8: Session-based DASH operations- Amendment 1: URL customization and other extensions

## *Change 1:* *Replace 2nd paragraph in 4.1 with:*

Session parameter strings are strings pertaining to a specific time range on the timeline. These strings are parsed by the Session-based description client and are translated into parameters to be added to HTTP request(s) issued by the DASH client.

## *Change 2:* *Replace 4.2 with:*

Figure 1 shows the general architecture for session-based DASH operations.



Figure 1 High-level architecture of session-based DASH operations.

In this figure, the media content is described by MPD, and the corresponding media segments are delivered in the media delivery path. MPD also has a reference to an external document, Session-Based Description (SBD) document, which defines the variables and their values for the current session. SBD is parsed and interpreted by an SBD client.

This SBD client is conceptually a simple timed key-value store. It accepts requests for a named variable value given a time range and returns a string value.

The DASH client uses the variables and the associated values provided in the SBD to derive changes to the ~~segment~~ URL as well as an added query when the segment is requested using HTTP GET.

Note that there may be multiple session controllers controlling multiple aspects of the session, each of which providing one SBD document (e.g., one controlling forensic watermarking and another controlling access tokens).

As shown in the above figure, Session Client may apply its processing to the segment URL generated by the DASH Access Client, after retrieving enough information from the MPD in the DASH Access client. The SBD operations does not intercept the DASH Access Client operation other than changing the segment URL before DASH Access Client making the segment request.

Note: This standard does not preclude the use of newer versions of HTTP, such as HTTP/2, as this standard only relies on the existence of the GET method.

## *Change 3:* *Replace Table 4 with*

Table 4 MPD EssentialProperty Descriptor attributes for session-based DASH

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element or Attribute Name** | | | **Use** | **Description** |
|  | **EssentialProperty** | |  | instantiation of EssentialProperty (defined in ISO/IEC 23009-1) for session-based DASH operations |
|  |  | @schemeIdUri | M (string) | shall be set to “urn:mpeg:dash:sbd:2020”. |
|  |  | @value | M (string) | URL of the SBD document for this session |
|  |  | @urlClass | OD  (default: "segment") | specifies which HTTP GET requests shall be the subject to SBD processing. Value is a white spaced concatenated list of the following keys:  1) "segment" (all segment requests)",  2) "xlink" (all XLink resolution requests),  3) "mpd" (all MPD requests),  4) "callback" (all requests triggered by DASH callback events),  5) "chaining" (requests for chained-to MPDs,  6) "fallback" (requests for the alternative MPDs.  Each key defines an URL class. Each URL class defines a collocation of URLs that its members are used to request a different class of resources, such as the class of segments, the class of Xlink resolutions, and etc.  Default value is "segment", i.e. only segment URLs are subject to SBD processing. |
|  |  | @template | O | template for applying to the key-value pair found in SBD document. For each **Key**@name in the template, when the value of a **Key**@name is found in the SBD document, its corresponding key-value pair of SBD document shall replace the string ‘$’**Key**@name‘$’, where ‘$’ is unescaped ‘$, in the @template. The result, after applying the replacements for all ‘$’**Key**@name‘$’ in the @template, is added to the end of given URL  The @template value shall have no whitespace characters.  If absent, for each **Key**@name, the **Key**@name and its corresponding value in the SBD document separated by ‘=’ shall be added to the end of the given URL query, where each consecutive key-value pairs are separated by ‘&’. The order of key-value pairs is defined by the order of **Key** elements in this descriptor. |
|  |  | @hostTemplate | O | host template for applying to the value found in SBD document. For each **Host**@name in the template, when the value of a **Host**@name is found in the SBD document, its corresponding value of SBD document shall replace the string ‘$’**Host**@name‘$’, where ‘$’ is unescaped ‘$’, in the @hostTemplate. The result, after applying the replacements for all ‘$’**Host**@name‘$’ in the @hostTemplate, shall replace the given URL’s host (as defined in RFC 3986).  The @hostTemplate value shall have no whitespace characters. |
|  |  | @portTemplate | O | port template for applying to the value found in SBD document. For each **Port**@name in the template, when the value of a **Port**@name is found in the SBD document, its corresponding value of SBD document shall replace the string ‘$’**Port**@name‘$’, where ‘$’ is unescaped ‘$’, in the @portTemplate. The result, after applying the replacements for all ‘$’**Port**@name‘$’ in the @portTemplate, shall replace the given URL’s port (as defined in RFC 3986).The @portTemplate value shall have no whitespace characters.  If absent, for each **port**@name that occurs at least once in the given URL’s port, as well as found in the SBD document, the corresponding value in the SBD document shall replace the first occurrence of the @name the given URL’s port. |
|  |  | @pathTemplate | O | path template for applying to the value found in SBD document. For each **Path**@name in the template, when the value of a **Path**@name is found in the SBD document, its corresponding value of SBD document shall replace the string ‘$’**Path**@name‘$’, where ‘$’ is unescaped ‘$’, in the @pathTemplate. The result, after applying the replacements for all ‘$’**Path**@name‘$’ in the @pathTemplate, shall replace the given URL’s path (as defined in RFC 3986).  The @pathTemplate value shall have no whitespace characters.  If absent, for each **path**@name that occurs at least once in the given URL’s path, as well as found in the SBD document, the corresponding value in the SBD document shall replace the first occurrence of the @name of the given URL’s path. |
|  |  | @urlMatch | OD  Default:  ’false’ | If ’true’ and any of the URL host, path or port is not replaced, then other parts are also not replaced, and the given URL shall remain unchanged. |
|  |  | @hostMatch | OD  Default:  ’false’ | If ’true’ and any key of @hostTemplate not found in the SBD document, then the given URL host shall remain unchanged. |
|  |  | @portMatch | OD  Default:  ’false’ | If ’true’ and any key of @portTemplate not found in the SBD document, then the given URL port shall remain unchanged. |
|  |  | @pathMatch | OD  Default:  ’false’ | If ’true’ and any key of @pathTemplate not found in the SBD document, then the given URL path shall remain unchanged. |
|  |  | **Key** | 1..N | a key name to be found in SBD document and its default value.  If absent, all keys and corresponding values in the corresponding keyList of the SBD document shall be added to the (sub)segment request URL query. |
|  |  | @name | M | Name of the key of the SBD document to be added to the queries for this session after processing described by @template. This value shall be a string without whitespaces, start with a letter, and contain only unreserved characters per RFC 3986.  If @name value does not appear in the keyList of SBD document, the @name value-@defaultValue value pair shall be used in the template or in the absence of the template. |
|  |  | @defaultValue | OD  (‘null’) | default value in key-value pair if @name value not found in SBD document or if there is no value defined for a requested time range or segment number in the SBD document. |
|  |  | **Host** | 0..N | a set of URL host keys to be found in SBD document and its default value. |
|  |  | @name | O | URI host key for SBD processing. This value shall be a string without whitespaces, start with a letter, and contain only unreserved characters per RFC 3986. |
|  |  | @default | O | The default host if the host key is not found in SBD |
|  |  | **Port** | 0..N | a set of URL port keys to be found in SBD document and its default value. |
|  |  | @name | O | URI port key SBD processing. This value shall be a string without whitespaces, start with a letter, and contain only unreserved characters per RFC 3986. |
|  |  | @default | O | The default port if the port key is not found in SBD |
|  |  | **Path** | 0..N | a set of URL path keys to be found in SBD document and its default value. |
|  |  | @name | O | URI path key SBD processing. This value shall be a string without whitespaces, start with a letter, and contain only unreserved characters per RFC 3986. |
|  |  | @default | O | The default path if the path key is not found in SBD |
| Legend:  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 @. | | | | |

## *Change 4:* *Replace 5.2.1 with*

|  |
| --- |
| <!—Descriptor schema -->  <?xml version="1.0" encoding="UTF-8"?>  <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"    xmlns:sbd="urn:mpeg:dash:sbd:2020" targetNamespace="urn:mpeg:dash:sbd:2020" elementFormDefault="qualified" attributeFormDefault="unqualified">      <!-- Key -->      <xs:complexType name="Key">          <xs:attribute name="name" type="xs:Name"/>          <xs:attribute name="defaultValue" type="StringNoWhitespaceType"/>          <xs:anyAttribute namespace="##other" processContents="lax"/>      </xs:complexType>      <!-- Host -->      <xs:complexType name="Host">          <xs:attribute name="name" type="xs:Name"/>          <xs:attribute name="default" type="StringNoWhitespaceType"/>          <xs:anyAttribute namespace="##other" processContents="lax"/>      </xs:complexType>      <!-- Path -->      <xs:complexType name="Path">          <xs:attribute name="name" type="xs:Name"/>          <xs:attribute name="default" type="StringNoWhitespaceType"/>          <xs:anyAttribute namespace="##other" processContents="lax"/>      </xs:complexType>      <!-- Port -->      <xs:complexType name="Port">          <xs:attribute name="name" type="xs:Name"/>          <xs:attribute name="default" type="StringNoWhitespaceType"/>          <xs:anyAttribute namespace="##other" processContents="lax"/>      </xs:complexType>      <xs:attribute name="urlClass" type="xs:string"/>  <xs:attribute name="template" type="StringNoWhitespaceType"/>  <xs:attribute name="hostTemplate" type="StringNoWhitespaceType"/>  <xs:attribute name="pathTemplate" type="StringNoWhitespaceType"/>  <xs:attribute name="portTemplate" type="StringNoWhitespaceType"/>  <xs:attribute name="urlMatch" type=" xs:boolean"/>  <xs:attribute name="hostMatch" type=" xs:boolean"/>  <xs:attribute name="pathMatch" type=" xs:boolean"/>  <xs:attribute name="postMatch" type=" xs:boolean"/>  </xs:schema> |

## *Change 5:* *Replace Table 7 with*

Table 7 orderline element semantics

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element or Attribute Name** | | | **Use** | **Description** |
|  |  | v | CM (array of string) | list of values in order matching their names in keylist , separated by white space. The default value is specified in **Key**@defaultValue attribute.  Each value shall contain only unrestricted characters per RFC 3986 Section 2.2. Namely, the permitted characters are uppercase and lowercase letters, decimal digits, hyphen, period, underscore, and tilde. |
|  |  | n | O (integer) | starting number of the (sub)segment to which the value applies, i.e., for “n”:”5”, the first 4 (sub)segments are skipped and the fifth (sub)segment is the starting (sub)segment.  If absent, the starting (sub)segment is the first (sub)segment in corresponding MPD or Period element, depending on which element the SBD descriptor is included.  Only one of s or n attributes may be present, but not both. |
|  |  | r | O (integer) | number of additional (sub)segments in this **O** element using the key-value pairs, e.g. “r”:”1”, two (sub)segments are using the key-value pairs of this element.  If not present, only one (sub)segment in this element is using the key-value pairs defined in this element. |
|  |  | collective | OD (boolean)  Default: ‘false’ | if “true”, with the processing of each URL, regardless of its class, the orderline is advanced.  If “false”, the order line is applied to each URL class separately, e.g. SBD processing of a URL from one URL class does not have any impact on the processing of the next URL if the second URL is from a different URL class. |
| Legend:  M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory.  JSON: objects are bold; name-value pairs are non-bold. | | | | |

## *Change 6:* *Replace Table 8 with*

Table 8 KeyValue JSON schema

|  |
| --- |
| {    "$id": "https://github.com/MPEGGroup/DASHPart8/keyvalue-schema.json",    "$schema": "http://json-schema.org/draft-07/schema#",    "title": "The schema for the array of keyvalue objects, as defined in ISO/IEC 23009-8 Session-based DASH operations",    "type": "array",    "items": {      "type": "object",      "required": [        "keyList"      ],      "oneOf": [        {          "required": [            "timeline"          ]        },        {          "required": [            "orderline"          ]        }      ],      "properties": {        "keyList": {          "type": "array",          "item": "string"        },        "comment": {          "type": "string"        },        "timescale": {          "type": "integer",          "minimum": 0,          "default": 1        },        "type": {          "type": "string",          "enum": [            "static",            "dynamic"          ],          "default": "static"        },        "startTime": {          "type": "integer",          "minimum": 0        },        "duration": {          "type": "integer",          "minimum": 0        },        "loop": {          "type": "boolean",          "default": "false"        },        "timeline": {          "type": "array",          "item": {            "type": "object",            "required": [              "v"            ],            "properties": {              "v": {                "type": "array",                "item": "string"              },              "s": {                "type": "integer",                "minimum": 0              },              "n": {                "type": "integer",                "minimum": 0              },              "r": {                "type": "integer",                "minimum": -1,                "default": 0              },              "d": {                "type": "integer",                "minimum": 0              }            }          }        },        "orderline": {          "type": "array",          "item": {            "type": "object",            "required": [              "v"            ],            "properties": {              "v": {                "type": "array",                "item": "string"              },              "n": {                "type": "integer",                "minimum": 0              },              "r": {                "type": "integer",                "minimum": -1,                "default": 0              },              "collective": {                "type": "boolean",                "default": "false"              }            }          }        }      },      "if": {        "properties": {          "type": {            "const": "dynamic"          }        }      },      "then": {        "properties": {          "ttl": {            "type": "integer",            "minimum": 0          }        }      }    }  } |

## *Change 7:* *Replace 7.1 with*

The client shall recognize and process the SBD descriptor’s @schemeIdUri according to clause 6.

The client shall download the SBD document from the location indicated by the SBD descriptor’s @value.

The client shall parse the SBD document JSON object and build the timeline/orderline table according to clause 7.

The client shall find the matching row in the timeline/orderline table of a requesting (sub)segment according to clause 7.

The client shall find the corresponding value for each key listed in the SBD descriptor for the matching row, replaces the host, path and port of the segment URL if requested, and build the query with the order of keys defined in the SBD descriptor according to clauses 6 and 7.

The client shall calculate the URL host, path and port, and the query for the matching row, build the URL and query string, according to clause 6.

## *Change 8:* *Replace 7.2.2 with*

*Inputs: @keys, (sub)segment EPT/(sub)segment time/number, segment/subsegment URL*

*Output: string q*

1. The SBD client calculates the earliest presentation time (EPT)/the number of a requested (sub)segment.
2. The SBD client finds the row i in the timeline/orderline table such that (sub) segment EPT is ti<= EPT<ti+1 or *segment number* = ni.
3. For each name in element **Key**j in the Essential Property descriptor
   1. Find a matching item key in keylist array
   2. Find the corresponding item value in the v array of the entry row i
   3. If template exists,
      1. replace the template’s substring between ‘$’ with valueand add the results to the end of query string
      2. Otherwise, add key=value to the end of query *q*
   4. If any of hostTemplate, portTemplate, and pathTemplate exists, for each
      1. replace the template’s substring of the corresponding part of segment/subsegment URL between corresponding key between two ‘$’ with it found value in the SBD table, or
      2. if the match is not found in the table, replace the corresponding part of the segment/subsegment URL with it default value.
4. Return the updated URL with the query *q* to the DASH client

NOTE: when the full URL is assembled, query *q* will be prefixed with the character `?` if it starts the query part of the URL, or `&` otherwise.

## *Change 9:* *Replace 7.2.3 with*

The DASH client uses the URL received from the SBD to request the (sub)segment..

NOTE: without SBD, all requests for different subsegments within a representation will use the same URL, but different byte ranges. Application of SBD will result in different *q* values for different segments (depending on their earliest presentation time), hence the single URL assumption no longer holds.

## *Change 10:* *Replace 8.1 with*

|  |  |
| --- | --- |
| <AdaptationSet  mimeType="video/mp4" segmentAlignment="true" startWithSAP="1"  maxWidth="1280" maxHeight="720" maxFrameRate="24000/1001" par="16:9">  <EssentialProperty schemeIdUri=" urn:mpeg:dash:sbd:2020"  value=" <http://example.com/variant_sequence.json?client_id=ctulhu>"  xmlns:sbd="urn:mpeg:dash:sbd:2020"  sbd:template=”&parameter=$p1$”  sbd:hostTemplate=”$subdomain.$example.com” >  <sbd:Key name=”p1” defaultValue=”nil”/>  <sbd:Host name=”subdomain.” default=””/>  </EssentialProperty>  <SegmentTemplate duration="2" startNumber="1" media=" example.com/m1/video/$RepresentationID$\_$Number%05d$.mp4">  <Representation id="720p" codecs="avc3.4d401f" width="1280" height="720" frameRate="24000/1001" sar="1:1" bandwidth="3000000"/>  <Representation id="360p" codecs="avc3.4d401f" width="640" height="360" frameRate="24000/1001" sar="1:1" bandwidth="1500000"/>  </AdaptationSet> |  |

## *Change 11:* *Replace 8.2.1 with*

|  |
| --- |
| {      "KeyValue": [          {              "keylist": ["p1", "subdomain."],              "comment": "A/B sequence document for user Ctulhu",              "timeline": [                  { "s": "0", "d": "3", "v": ["861d34d7-56eb-4893-a7b7-60edabebe3e6", "s1."]},                  {"d": "2", "v": ["d8a56fd3-6c21-44be-94f8-a519cd6b4169", "s2."]},                  {"d": "1", "v": ["75b49311-008c-4272-9aff-b855ee94707a", "s3."]}              ]          }      ]  } |

## *Change 12:* *Replace 8.2.2 with*

|  |
| --- |
| {        "KeyValue": [          {              "keylist": ["p1"],              "comment": "A/B sequence document for user Ctulhu",              "orderline": [                  {"v": ["861d34d7-56eb-4893-a7b7-60edabebe3e6", "s1."]},                  {"v": ["d8a56fd3-6c21-44be-94f8-a519cd6b4169", "s2."]},                  {"v": ["75b49311-008c-4272-9aff-b855ee94707a", "s3."]}              ]          }      ]  } |

## *Change 13:* *Replace 8.3 with*

The first three segments described by the SBD and MPD above will have the following URLs:

http://s1.example.com/m1/video/720p\_00000.mp4?parameter1=d4baa823-8ff2-445b-847b-d6ead52cf6ce

http://s2.example.com/m1/video/720p\_00001.mp4?parameter1=861d34d7-56eb-4893-a7b7-60edabebe3e6

http://s3.example.com/m1/video/720p\_00004.mp4?parameter1=d8a56fd3-6c21-44be-94f8-a519cd6b4169