

DASH-IF CTS 001 V1.0.0 (2023-06)



**DASH-IF Candidate Technical Specification:  
Content Steering for DASH**

***DASH Industry Forum***

3855 SW 153rd Dr.  
Beaverton, OR 97003 - USA

---

Email : [admin@dashif.org](mailto:admin@dashif.org)

***Important notice***

---

The present document can be downloaded from:  
<http://www.dashif.org/guidelines>

---

# Contents

Disclaimer.....	4
Intellectual Property Rights.....	4
Foreword.....	4
Modal verbs terminology .....	4
Executive summary .....	4
1 Scope.....	5
2 References.....	5
2.1 Normative references.....	5
2.2 Informative references.....	5
3 Definition of terms, symbols and abbreviations.....	5
3.1 Terms.....	5
3.2 Symbols.....	6
3.3 Abbreviations.....	6
4 Overview and Architecture .....	6
5 Content Steering Signalling in DASH .....	8
5.1 Introduction.....	8
5.2 Signalling Requirements and Recommendations.....	8
6 DASH Steering Manifest and Server Behaviour.....	9
6.1 Overview .....	9
6.2 JSON Syntax.....	9
6.3 Semantics.....	9
7 Normative DASH Client Steering behaviour.....	11
8 Extended HTTP GET request parametrization instructions.....	13
8.1 URL Query information for Content Steering .....	13
8.2 Updates to Annex I - Flexible Insertion of URL Parameters.....	14
8.2.1 Introduction.....	14
8.2.2 Modified content steering server URLs building process .....	14
<b>Annex A: Example implementations (informative).....</b>	<b>15</b>
<b>Annex (informative): Change History .....</b>	<b>19</b>

---

## Disclaimer

This document is a candidate Technical Specification. DASH-IF is publishing this initially, but the document has been submitted to ETSI and ETSI is asked to publish this document under the Publicly Available Specifications (PAS) agreement. For details on the PAS process, please refer to ETSI PAS Process Guide:

[https://www.etsi.org/images/files/ETSI\\_PAS\\_Process\\_Guide.pdf](https://www.etsi.org/images/files/ETSI_PAS_Process_Guide.pdf)

Note that the ETSI specification may be different, addressing editorial updates.

In addition, in case of any identified issues or bugs, please file issues here: <https://github.com/Dash-Industry-Forum/Content-Steering/issues>.

---

## Intellectual Property Rights

This is a document made available by DASH-IF. The technology embodied in this document may involve the use of intellectual property rights, including patents and patent applications owned or controlled by any of the authors or developers of this document. No patent license, either implied or express, is granted to you by this document. DASH-IF has made no search or investigation for such rights and DASH-IF disclaims any duty to do so. The rights and obligations which apply to DASH-IF documents, as such rights and obligations are set forth and defined in the DASH-IF Bylaws and IPR Policy including, but not limited to, patent and other intellectual property license rights and obligations. A copy of the DASH-IF Bylaws and IPR Policy can be obtained at <http://dashif.org/>.

The material contained herein is provided on an "AS IS" basis and to the maximum extent permitted by applicable law, this material is provided AS IS, and the authors and developers of this material and DASH-IF hereby disclaim all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of workmanlike effort, and of lack of negligence.

In addition, this document may include references to documents and/or technologies controlled by third parties. Those third-party documents and technologies may be subject to third party rules and licensing terms. No intellectual property license, either implied or express, to any third-party material is granted to you by this document or DASH-IF. DASH-IF makes no warranty whatsoever for such third-party material.

Note that technologies included in this document and for which no test and conformance material is provided, are only published as candidate technologies, and may be removed if no test material is provided before releasing a new version of this guidelines document. For the availability of test material, please check <https://www.dashif.org>.

---

## Foreword

This Technical Specification (TS) has been produced by the DASH-IF Technical Working Group.

---

## Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](#) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in deliverables except when used in direct citation.

---

## Executive summary

Content distributors often use multiple Content Delivery Networks (CDNs) to distribute their content to the end-users. They may upload a copy of their catalogue to each CDN, or more commonly have all CDNs pull the content from a common origin. Alternate URLs are generated, one for each CDN, that point at identical content. DASH players may access alternate URLs in the event of delivery problems. Content steering describes a deterministic capability for a content distributor to switch the content source that a player uses either at start-up or midstream, by means of a remote steering service.

---

# 1 Scope

The present document specifies Content Steering for DASH.

---

## 2 References

### 2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, DASH-IF cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document:

- [1] HTTP Live Streaming 2nd Edition, <https://datatracker.ietf.org/doc/html/draft-pantos-hls-rfc8216bis-10>
- [2] ISO/IEC 23009-1:2022/CDAM2:2023: “Information technology -- Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats” (available as MDS22328\_WG03\_N00804)
- [3] RFC3986 Uniform Resource Identifier (URI): Generic Syntax <https://datatracker.ietf.org/doc/html/rfc3986>
- [4] RFC9110 HTTP Semantics <https://www.rfc-editor.org/rfc/rfc9110.html>
- [5] RFC6585 Additional HTTP Status Codes <https://www.rfc-editor.org/rfc/rfc6585>

### 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are not necessary for the application of the present document, but they assist the user with regard to a particular subject area.

- [i.1] DASH-IF Interoperability Points, Part 1: Overview, architecture and interfaces
- [i.2] ISO/IEC 14496-12:2021 “Information technology -- Coding of audio-visual objects -- Part 12: ISO base media file format”
- [i.3] ISO/IEC 23000-19:2020 “Information technology -- Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media”
- [i.4] CTA-5004: Web Application Video Ecosystem-Common Media Client Data. [Online] Available: <https://cdn.cta.tech/cta/media/media/resources/standards/pdfs/cta-5004-final.pdf>

---

# 3 Definition of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the following terms apply:

**Content Steering Server:** A network element that provides steering information to one or several or many DASH Players for DASH operation across multiple CDNs.

**DASH Content Steering Manifest:** A document that includes steering instructions to a DASH player provided by a Content Steering Server

## 3.2 Symbols

For the purposes of the present document, the following symbols apply:

## 3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

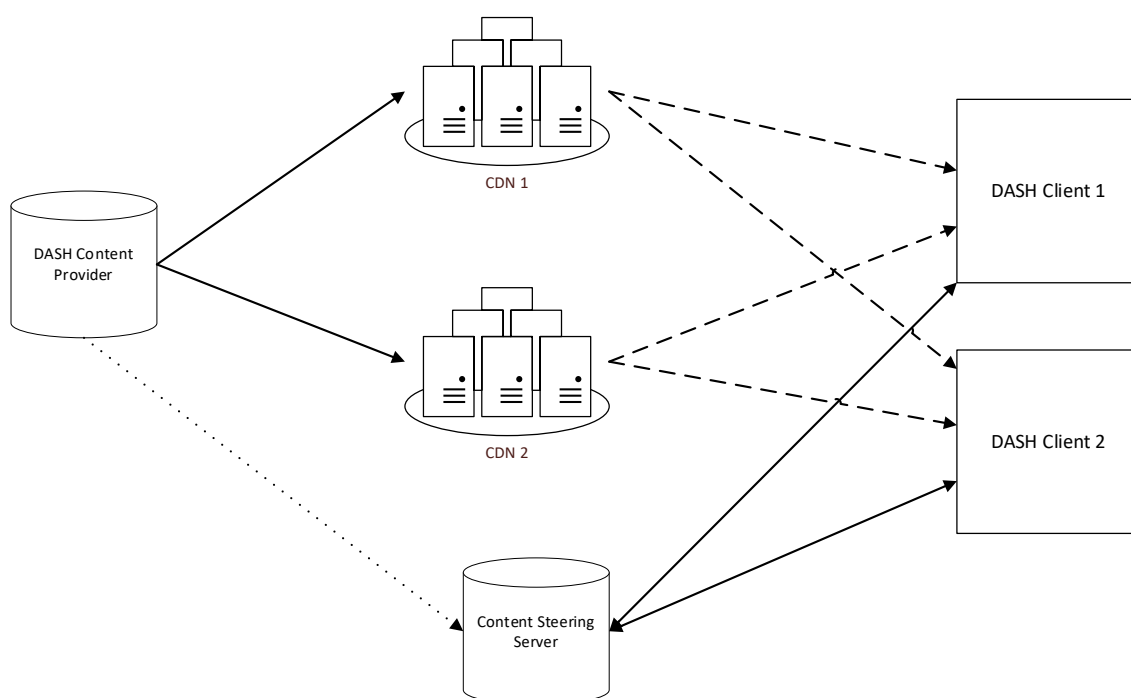
CDN	Content Delivery Network
CMAF	Common Media Application Format
CMCD	Common Media Client Data
CTA	Consumer Technology Association
DASH	Dynamic Adaptive Streaming over HTTP
DCSM	DASH Content Steering Manifest
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ISO	International Standards Organization
MPD	Media Presentation Description
TTL	Time-To-Live
URL	Uniform Resource Locator

# 4 Overview and Architecture

Content distributors often use multiple Content Delivery Networks (CDNs) to distribute their content to the end-users as shown in Figure 4-1. They may upload a copy of their catalogue to each CDN, or more commonly have all CDNs pull the content from a common origin. In the DASH Media Presentation Description (MPD) as defined in ISO/IEC 23009-1 [2], multiple URLs are provided, one for each CDN, that point at identical content. Typically, a DASH player will access content from one single location, using the default location defined by the MPD.

If the DASH player then observes delivery problems, it may choose to access content via the alternate URLs. This operation is completely client-driven, is not standardized between players and may not be the desired behaviour of the content distributor.

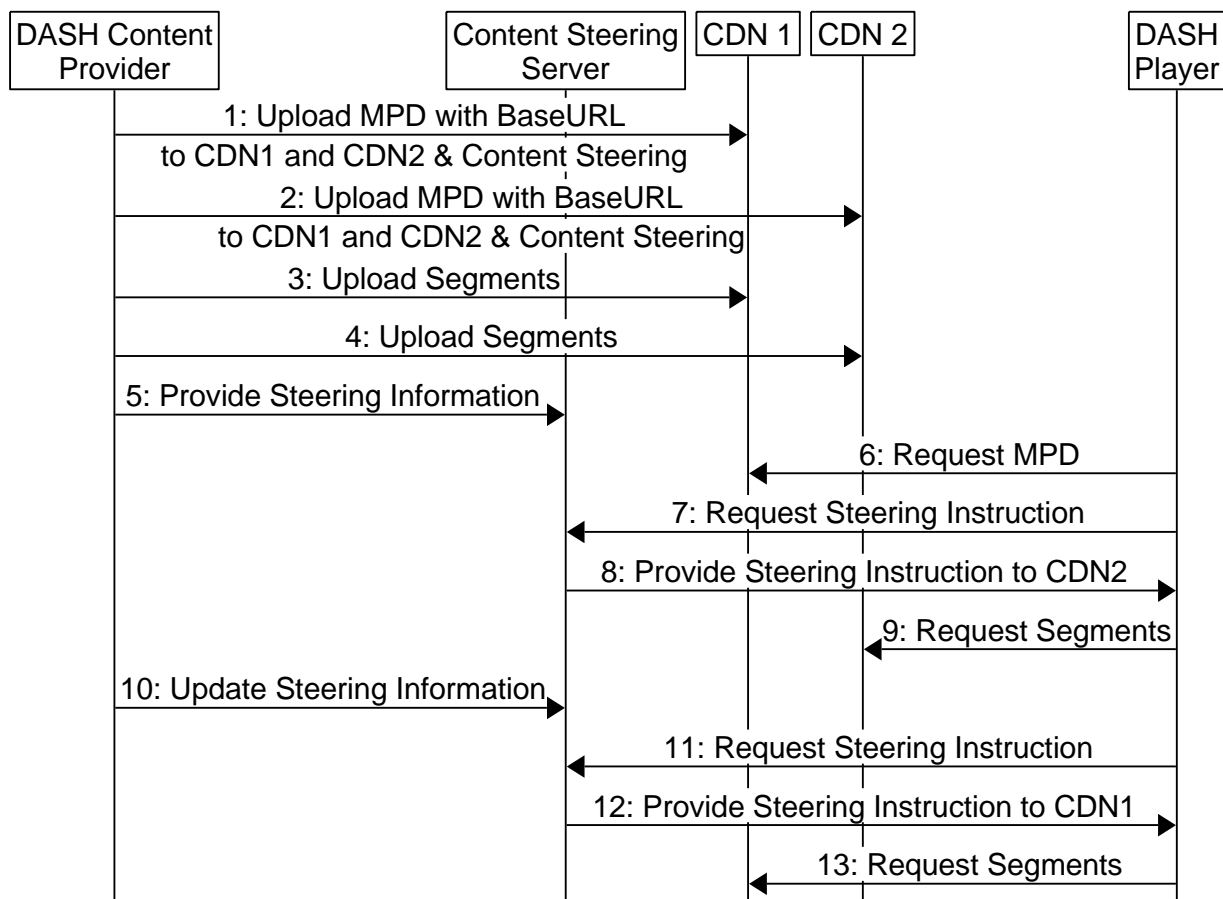
Content steering provides a deterministic capability for a content distributor to switch the content source that a player uses either at start-up or midstream, by means of a remote steering service.



**Figure 4-1 Basic Architecture with Content Steering**

Steering is accomplished by having the DASH client periodically access a content steering server to retrieve a steering manifest, which instructs the player as to the availability and priority of content sources.

The typical procedures followed when content steering is in use are shown in Figure 4-2 for the case when the content is provided on two CDNs. The DASH content provider generates an MPD that includes Base URLs to CDN1 and CDN2, as well as an address where the clients can access the content steering server. The provider also uploads the MPD and the Content segments to both CDNs. At the start of playback, the DASH client requests the MPD from one CDN, in this case from CDN2. It finds the content steering server URL, and it may find information that instructs it to contact the content steering server prior to the first segment request versus the default behaviour of making the request once its starting buffer is full. The player then makes a request to the content steering server. The content server responds with a content steering manifest and the DASH client uses the information within to prioritize the segment source, in this case from CDN2. After some time, the content provider may collect operational information from the participating clients, for example by using Common Media Client Data (CMCD) as defined in CTA-5004 [i.4]. Based on this information, the content provider may update the content steering server, and based on this updated information, the content steering manifest may change. When the client requests an update to the content steering manifest, new information may be provided that instructs the DASH client to request the Segments from CDN1 instead of CDN2. The DASH client then switches smoothly, at a segment boundary, to download the Segments from CDN1 instead of CDN2. The steering server response can also be used to steer the DASH client between alternate sources for DASH manifest refreshes, via service descriptors contained within the MPD.



<https://gitlab.com/msc-generator/v8.0>

**Figure 4-2 Typical procedures in Content Steering operation**

This specification provides the following detailed information to support interoperable operation of content steering in DASH Media Presentations:

- Clause 5 provides the details on how to signal different alternate CDNs in the DASH MPD using existing **BaseURL** elements as well as an extension to the DASH MPD in order to provide the location of the Content Steering server as well as some additional instructions to the client based on ISO/IEC 23009-1:2022/Amd.2 [2] using the **ContentSteering** element as well as the Service Location functionality. .

- Clause 6 provides the details on the DASH content steering manifest (DCSM) and the behaviour of the content steering server, in particular the semantics and JSON syntax of the steering server response.
- Clause 7 defines the DASH player behaviour in detail. A DASH player supporting Content Steering is required to implement the detailed procedures documented in this clause.
- Clause 8 provides guidelines on the usage of DASH Annex I for providing the client identity as part of the DASH player to content steering server communication.
- Annex A provides operational examples of a DASH MPD as well as of the response of the content steering server.

## 5 Content Steering Signalling in DASH

### 5.1 Introduction

Content distributors often use multiple Content Delivery Networks (CDNs) to distribute their content to the end-users. They may upload a copy of their catalogue to each CDN, or more commonly have all CDNs pull the content from a common origin. Alternate URLs are generated, one for each CDN, that point at identical content.

The DASH MPD supports the **BaseURL** element to allow the listing of these alternate URLs, as well as the **Location** and **PatchLocation** element for pointing at alternate manifest sources. DASH players may access alternate URLs in the event of delivery problems. Content steering describes a deterministic capability for a content distributor to switch the content source that a player uses either at start-up or midstream, by means of a remote steering service.

To enable Content Steering the following functionalities are provided in MPEG DASH.

- Content Steering information can be provided to the DASH client including the following information according to Annex K.3 of ISO/IEC 23009-1:2022/Amd.2:
  - **ContentSteeringServer**: A URL that can be used to access the Content Steering server. The URL references to a DASH Content Steering Manifest as defined in clause 6 of this specification.
  - **defaultServiceLocation**: This parameter specifies a space-separated list of Service Locations as defined in clause 5.6.6 of ISO/IEC 23009-1:2022/Amd.2 that the client should use to access the selected resources, in case multiple access exist. This for example applies, when no content steering server is available, or before a valid response from a content steering server is available.
  - **queryBeforeStart**: This flag, if set to true, indicates that the player is expected to resolve the response from the Steering Server prior to starting playback. The default value is false.
  - **clientRequirement**: The flag, if true, indicates that the client, if it is in the context of the Service Description shall follow the content steering rules as defined in clause 7 of this specification. If false, indicates to client that it is its own decision to make use of content steering operation or not.
- This content steering information can be provided as part of the MPD, either in the MPD directly or as part of a Service Description. The latter allows to target only a subset of specific clients that follow the service description. The MPEG-DASH specification does not prevent that the above content steering information is provided independent of the MPD, for example directly from the application using a configuration API for the DASH access engine as shown in clause K.2 of ISO/IEC 23009-1:2022.
- Service Location as introduced in clause 5.6.6 of ISO/IEC 23009-1:2022/Amd.2 allows to bind two or more resources a common network location, for example a common Content Delivery Network. This information may be attached to a **BaseURL** element, a **Location** element or a **PatchLocation** element.
- Flexible URL of URL Parameters as defined in Annex I of ISO/IEC 23009-1:2022/Amd.2, for which the request to a Content Steering server may add specific query parameters.

### 5.2 Signalling Requirements and Recommendations

Based on the functionalities in ISO/IEC 23009-1:2022/Amd.2, the following requirements hold for signaling Content Steering in a DASH MPD in the context of this specification.

1. A **ContentSteering** element shall be present either on MPD level if all clients are targeted, or, if a specific set of clients are targeted, as part of a **ServiceDescription** element. The **ContentSteering**



element shall appear at most once in a parent element. The value of the element shall reference to a DASH Content Steering Manifest (DCSM) as defined in clause 6.

If the resolved resource is not a Content Steering Manifest (DCSM), the resource is ignored.

2. The optional attributes `@defaultServiceLocation`, `@queryBeforeStart` and `@clientRequirement` may be used according to their semantics.
3. The `@serviceLocation` attribute may be present in **BaseURL**, **Location** and **PatchLocation** elements. If present and used in a content steering manifest, the string value of the `@serviceLocation` shall only contain characters from the set `[a..z]`, `[A..Z]`, `[0..9]`, `'.'`, `'-'`, and `'_'`.

NOTE: Overlapping `@serviceLocation` names for steerable and non-steerable BaseURLs is discouraged. i.e. if a server-side ad insertion provides **BaseURL** elements with an overlapping `@serviceLocation` value it could cause unintended behavior.

## 6 DASH Steering Manifest and Server Behaviour

### 6.1 Overview

Based on the description in clause 4, the DASH Steering server provides a Steering Manifest on a request from a DASH client. The DASH Content Steering Manifest (DCSM) is a json document and shall be formatted according to the JSON schema provided in clause 6.2.

The semantics of the key-value pairs of the DCSM are defined in clause 6.3.

A client shall ignore any key of the DCSM that it does not recognize. DCSM keys are case-sensitive.

Note: this structure is intentionally similar to that defined by [1] Section 7.1. for HLS for the purposes of interoperability. The DASH variant and versioning are defined in this document and intentionally exclude features in the HLS design which are not applicable to DASH.

### 6.2 JSON Syntax

```
{
  "VERSION": number,           // REQUIRED, must be an integer
  "TTL": number,              // REQUIRED, number of seconds
  "RELOAD-URI": string,       // OPTIONAL, URI
  "PATHWAY-PRIORITY": []      // REQUIRED, array of serviceLocation identifiers in order of preference
  "PATHWAY-CLONES":          // OPTIONAL, array of one or more Pathway Clone objects
  [
    {
      "BASE-ID": string,       // REQUIRED. Pathway ID of the Base Pathway
      "ID": string,           // REQUIRED. Pathway ID for the Pathway Clone
      "URI-REPLACEMENT":      // REQUIRED. URI replacement rules
      {
        "HOST": string,       // OPTIONAL. Hostname for cloned URIs
        "PARAMS":            // OPTIONAL. Query parameters for cloned URIs
        {
          JSON object where keys are query parameter names
          and values are query parameter values. Any reserved characters in the strings
          must be percent encoded [RFC3986]
        }
      }
    }
  ]
}
```

### 6.3 Semantics

The semantics of the DCSM are defined in Table 6.3-1.

Table 6.3-1 — Semantics of the DASH Content Steering Manifest

Key	Use	Description
DCSM		A URL that can be used to access the Content Steering server. The URL points to a DASH Content Steering Manifest (DCSM) as defined in clause 6.
VERSION	M	<p>The version of DCSM.</p> <p>This specification defines DASH Steering Manifest version 1.</p> <p>A client shall refuse to use a DCSM with an unrecognized value for this key.</p>
TTL	M	<p>Specifies how many seconds the client shall wait before reloading the DCSM.</p> <p>The recommended value is 300 seconds .</p> <p>The Steering Server may vary the TTL by client and the TTL may vary with each reload of the steering server manifest</p>
RELOAD-URI	O	<p>If present, specifies the URI the client shall use the next time it obtains the DCSM. The RELOAD-URI may be relative to the current DCSM Manifest URI.</p> <p>If not present, the current DCSM Manifest URI shall be used.</p>
PATHWAY-PRIORITY	OD Default []	<p>PATHWAY-PRIORITY is an array of <b>BaseURL@serviceLocation</b> and <b>Location@serviceLocation</b> IDs.</p> <p>Elements in the PATHWAY-PRIORITY array are ordered by preference of the <code>serviceLocation</code> to be selected, with the first being most preferred.</p> <p>If present, the value of this key shall contain at least one element for <code>serviceLocation</code>.</p> <p>A <code>@serviceLocation</code> ID in the PATHWAY-PRIORITY array shall not appear more than once.</p> <p>Clients shall ignore unrecognized <code>@serviceLocation</code> IDs in the PATHWAY-PRIORITY array.</p>
PATHWAY-CLONES	OD Default []	<p>PATHWAY_CLONES is an array of pathway clones. A pathway clone defines a virtual <b>BaseURL@serviceLocation</b> or <b>Location@serviceLocation</b>. A Pathway Clone is produced by taking an existing <code>serviceLocation</code> and applying well-defined replacements to the URIs of every <code>serviceLocation</code> member.</p> <p>If present, the value of this key shall contain at least one element defining a pathway clone.</p>
<p><b>Key</b></p> <p>M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory</p>		

## 7 Normative DASH Client Steering behaviour

A DASH client supporting DASH Content Steering as defined in this specification shall adhere to the following procedures:

1. If the **ContentSteering** element is present in the MPD and the @clientRequirement is not set to FALSE, then the client shall parse the element and extract the server URI, as well as the @defaultServiceLocation, and @queryBeforeStart attributes, if present.
2. If any extended HTTP GET request parametrization instructions as defined in clause 8 are present in the MPD which target the **ContentSteering** element, then they shall be executed at this stage to modify the server URI.
3. The client shall use the server URI as the STEERING-SERVER-URL.
4. If @queryBeforeStart is absent, or present and set to false, then the client shall follow its default start-up sequence. Once playback has started and the client has reached its target buffer, it shall proceed to the next step.
5. The client shall make a GET request to the STEERING-SERVER-URL.
6. The GET request should be accompanied by the two optional query parameters shown below.
  - a. The `_DASH_pathway` parameter shall contain a value of the currently selected **BaseURL**@serviceLocation, contained in double-quotes. If playback has not yet started due to this being the first request with @queryBeforeStart set to true, then the `_DASH_pathway` parameter shall be omitted. If multiple @serviceLocation attributes are being currently matched, for example in the Location element and in a **BaseURL** element, or if multiple @serviceLocation attributes have been matched since the last steering server request, then all matched @serviceLocation attributes shall be listed, separated by a comma delimiter and contained within a single set of double-quotes.
  - b. The `_DASH_throughput` parameter represents a current prediction of media download throughput observed by the client, in units of integer bits per second, from the applied @serviceLocation. The exact method of bit rate estimation may vary by client. If the client has multiple throughput estimates available, for example from demuxed audio and video downloads, then it should report the higher of the available estimates. If playback has not yet started due to this being the first request with @queryBeforeStart set to true, or if the throughput value is unknown for other reasons, then the `_DASH_throughput` parameter should be omitted. If the `_DASH_pathway` parameter value references multiple serviceLocations, then the `_DASH_throughput` parameter value shall consist of a comma separated list of throughput values, sequenced such that the n'th item in the `_DASH_throughput` list applies to the n'th item in the `_DASH_pathway` list.

NOTE: the client may send additional useful information via query arguments to the steering server, such as the transfer latency between client and server. It should not use the `_DASH_` prefix to transfer this information and such communication would constitute a private relationship between the distributor and the steering server service. Distribution vendors will collect constant real-time QoE information from their clients which they will use to make steering server decisions and it is not anticipated that the core steering decisions will be based solely on information provided by clients as they make steering server requests.

7. Upon receipt of the steering server response, the client should parse it and retrieve the VERSION, TTL, PATHWAY-PRIORITY array, the optional RELOAD-URI and the optional PATHWAY-CLONES array. The client shall ignore any steering manifest keys it does not recognize. Manifest keys are case-sensitive.
8. The client sets a timer to re-request the STEERING-SERVER-URL after TTL seconds.
9. If RELOAD-URI is present, then the client shall update the STEERING-SERVER-URL to match that specified by RELOAD-URI. The RELOAD-URI may be relative to the current server URI.
10. If the VERSION is a value other than 1, then the client shall abort any further steering behavior.
11. The string entries in the PATHWAY-PRIORITY array represent a prioritized list of serviceLocations from which playback should take place, with the highest priority option listed first. This highest priority item is termed the preferred service location. The PATHWAY-PRIORITY array is applicable across all periods present

in the manifest as long as they include one or more **BaseURL** elements with a `@serviceLocation` attribute value included in the `PATHWAY-PRIORITY` array, or inherit from top level **BaseURL** elements with a `@serviceLocation` attribute value included in the `PATHWAY-PRIORITY` array. The `PATHWAY-PRIORITY` array may also reference a `serviceLocation` which is defined by the `ID` attribute of a `PATHWAY-CLONE`.

12. If the preferred service location is one defined by a `PATHWAY-CLONE`, then the client must synthesize the new `serviceLocation` by cloning an existing `serviceLocation`. The `serviceLocation` it shall clone is defined by the `PATHWAY-CLONES@BASE-ID` attribute. The client would process the **BaseURL** or **Location** element URI as normal, but then make two changes to the generated URI. Firstly, it would substitute the `HOST` component of the URI with the new host, defined by the `URI-REPLACEMENT@HOST` attribute. Secondly, if the optional `URI-REPLACEMENT@PARAMS` attribute is present, then for each item in the `PARAMS` object, it would add query arguments to the URL matching the key and value defined by the `PARAMS` item. These query arguments are appended to any existing arguments which may already be present in the URI due to a template definition, inheritance or Annex I Extended HTTP GET request parametrization. If a parameter of the same name is already present in the URI, then it shall be replaced by the one from the `PARAMS` object. A DASH client must ignore all other attributes in the `PATHWAY-CLONE` object, including any `URI-REPLACEMENT.PER-VARIANT-URIS` and `URI-REPLACEMENT.PER-RENDITION-URIS` objects. A pathway clone may use another pathway clone as its base if it appears earlier in the `PATHWAY-CLONES` array. A client that does not have a `serviceLocation` specified by the `BASE-ID` string of a Pathway Clone object shall ignore the Pathway Clone.
13. If the client is playing content defined by a **BaseURL** element with a `@serviceLocation` not equal to the preferred service location, then the client shall switch at the next segment load to retrieving future content from the **BaseURL** referenced by the preferred service location. Note that existing requests against a prior `@serviceLocation` should be allowed to complete and forward buffers should not be trimmed. If the preferred service location is not described in the DASH MPD, then the client should attempt to switch to the next highest priority service location. If no `@serviceLocation` attributes match the manifest, then the client may ignore the current `PATHWAY-PRIORITY` array and make a default decision about which **BaseURL** to use. It should still reload the `RELOAD-URI` after the specified `TTL` interval in case new service locations are added.
14. If the manifest contains one or more **Location** elements with `@serviceLocation` attributes, then prior to a manifest update, the client shall evaluate the `@serviceLocation` attributes against the `PATHWAY-PRIORITY` array and select the **Location** element with the highest matching `@serviceLocation`. If no `@serviceLocation` attributes match then the client may make a default decision about which **Location** element to select.
15. A HTTP 410 [4] response from the steering server indicates that access to the steering server is no longer available and that this condition is likely to be permanent. As a result, if the client has a previous valid `PATHWAY-PRIORITY` array from the steering server, it should continue to enforce that prioritization for the remainder of playback and should cancel any all future reloads. If the 410 response is received on the first steering server request, then the client should abandon all steering behaviors, cancel all future requests and proceed with playback as if the **ContentSteering** element were not present.
16. A HTTP 429 response [5] from the steering server indicates that the server has received too many requests. The client should react by substituting the parsed `TTL` value with the 429 `Retry-After` value, if present.
17. If the client encounters playback problems which would normally cause it to try an alternate **BaseURL**, it may continue to make that local switching decision, while following these constraints:
  - a. The client may only try **BaseURL**@`serviceLocation` attributes which were present in the last steering server response.
  - b. The client shall try these `serviceLocations` in the order in which they were prioritized in the last steering server response.
  - c. As it switches away from the highest priority `@serviceLocation` for local performance reasons, it shall exclude that `@serviceLocation` for a time-limited period equal to the last steering server `TTL` that it received. Effectively this means that if the next steering server response again assigns the excluded `@serviceLocation` as the highest priority, the client shall ignore that instruction and instead process the `PATHWAY-PRIORITY` array as if the excluded `@serviceLocation` were not present.

18. If the client encounters playback problems which would normally cause it to try an alternate **Location**, it may continue to make that local switching decision, while following these constraints:
- The client may only try **Location**@serviceLocation attributes which were present in the last steering server response. If no @serviceLocation attributes match, then the client may make a default decision as to which **Location** element to use.
  - The client shall try these **Location**@serviceLocation attributes in the order in which they were prioritized in the last steering server response.
  - As it switches away from the highest priority **Location**@serviceLocation for local performance reasons, it shall exclude that @serviceLocation for a time-limited period equal to the last steering server TTL that it received. Effectively this means that if the next steering server response again assigns the excluded @serviceLocation as the highest priority, the client shall ignore that instruction and instead process the PATHWAY-PRIORITY array as if the excluded @serviceLocation were not present.

## 8 Extended HTTP GET request parametrization instructions

### 8.1 URL Query information for Content Steering

Query arguments attached to the request if the DCSM can be used to initialize content steering parameters.

In order to do so, it is recommended that the URL substitution mechanism and syntax as defined in MPEG-DASH ISO/IEC 23009-1 [2], clause I.3, “Extended HTTP GET request parametrization” is used. In order to fully support functionality, the substitution mechanism needs to be extended. Details of this is provided in clause 8.2.

The following provides guidelines on how to make use of this functionality.

Consider a MPD URL of the following form

```
https://cdn.distributor.com/content/common-cachable-
manifest.mpd?steeringToken=12345
```

with relevant contents shown as:

```
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:mpeg:dash:schema:mpd:2011"
xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd" type="dynamic"
minimumUpdatePeriod="PT30S" timeShiftBufferDepth="PT30M" availabilityStartTime="2022-02-
25T12:30:00" minBufferTime="PT4S" profiles="urn:mpeg:dash:profile:isoff-live:2011">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:urlparam:2014"
  xmlns:up="urn:mpeg:dash:schema:urlparam:2014">
    <up:ExtUrlQueryInfo queryTemplate="token=$query:steeringToken$" useMPDUrlQuery="true"
includeInRequests="steering"/> </up:ExtUrlQueryInfo>
    <BaseURL serviceLocation="alpha">https://cdn1.example.com/</BaseURL>
    <BaseURL serviceLocation="beta">https://cdn2.example.com/</BaseURL>
    <Period id="1">
      ...
    </Period>
  <ContentSteering defaultServiceLocation="alpha"
queryBeforeStart="false">https://steeringservice.com/app/instance1234</ContentSteering>
</MPD>
```

In this example the request to the steering server would be processed as

```
https://steeringservice.com/app/instance1234?steeringToken=12345&_DASH_pathwa
y=alpha&_DASH_throughput=5140000
```

Note that also a @includeInRequests="" may be added to URLs to support any kind of URL being extended with query parameters.

## 8.2 Updates to Annex I - Flexible Insertion of URL Parameters

### 8.2.1 Introduction

In order to fully support the functionality of content steering, extensions to ISO/IEC 23009-1:2022/Amd.2 [2] are available in clause I.3, “Extended HTTP GET request parametrization” . These include

- Updates to Table I.3 in order to add steering (“steering”) and any (“\*”) requests.
- Updates to clause I.2.4.5 adding an example

### 8.2.2 Modified content steering server URLs building process

In order to create modified content steering server URLs for example as

```
https://steeringservice.com/app/instance1234?token=1234&sessionID=h48djn&_DASH_pathway=beta&_DASH_throughput=5140000
```

the `_DASH_` parameters are added automatically by the player as part of the processing rules for Content Steering according to clause 7 and are independent of the URL building mechanisms described in Annex I of ISO/IEC 23009-1 [2].

## Annex A: Example implementations (informative)

### A.1 Basic workflow example

This case illustrates service location changes along with performance override.

A DASH MPD is presented to a player.

```
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:mpeg:dash:schema:mpd:2011"
xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd" type="dynamic"
minimumUpdatePeriod="PT30S" timeShiftBufferDepth="PT30M" availabilityStartTime="2022-02-
25T12:30:00" minBufferTime="PT4S" profiles="urn:mpeg:dash:profile:isoff-live:2011">
  <BaseURL serviceLocation="alpha">https://cdn1.com/</BaseURL>
  <BaseURL serviceLocation="beta">https://cdn2.com/</BaseURL>
  <Period id="1">
    <AdaptationSet mimeType="video/mp4" codecs="avc1.4D401F" frameRate="30000/1001"
segmentAlignment="true" startWithSAP="1">
      <BaseURL>video/</BaseURL>
    ...
  </AdaptationSet>
</Period>
<ContentSteering defaultServiceLocation="beta"
queryBeforeStart="true">https://steering.service.com/app/instance1234?token=234523452</ContentSt
eering>
</MPD>
```

The player would parse the **ContentSteering** element upon receiving the MPD. Since the @queryBeforeStart attribute is present and set to true, instead of starting playback using the @defaultServiceLocation of "beta", it would make a request to the steering server at https://steering.service.com/app/instance12345. This request would be

<https://steering.service.com/app/instance1234?token=234523452>

Note the `_DASH_` params are not attached to this request since the player has not yet started playback. The server may then return the JSON response below:

```
{
  "VERSION": 1,
  "TTL": 300,
  "RELOAD-URI": "https://steering.service.com/app/instance12345?session=abc"
  "PATHWAY-PRIORITY": ["alpha","beta"]
}
```

The player would recognize that the highest priority serviceLocation specified is "alpha", so it would use the **BaseURL** construct of https://cdn1.com/ as it begins to request content. The player would then set a timer so that in 300s, when the throughput it was estimating is 5.14Mbps, it would again query the steering server, with the URL

[https://steering.service.com/app/instance12345?session=abc&\\_DASH\\_pathway=alpha&\\_DASH\\_throughput=5140000](https://steering.service.com/app/instance12345?session=abc&_DASH_pathway=alpha&_DASH_throughput=5140000)

At that time the steering server may return

```
{
  "VERSION": 1,
  "TTL": 250,
  "RELOAD-URI": "https://steering.service.com/app/instance12345?session=abc"
  "PATHWAY-PRIORITY": ["beta","alpha"]
}
```

The player would then switch to loading the next media objects using the **BaseURL** of https://cdn2.com/. 250s later it would again request the steering service and the cycle would continue until end-of-stream was reached.

Let's now assume that the player runs in to a delivery problem 100s after the last steering server response. This problem may be triggered by 404 responses, or throughput degradation. The player decides that "beta" is not a good source and makes a local decision to switch to the next highest priority @serviceLocation "alpha" and to blacklist "beta". This blacklist last for a time-period equal to the last TTL received, which is 250s. 150s after taking this

action, the player calls the steering server and reports the @serviceLocation it is currently playing using the `_DASH_pathway` parameter:

[https://steeringservice.com/app/instance12345?session=abc&\\_DASH\\_pathway=alpha&\\_DASH\\_throughput=4880000](https://steeringservice.com/app/instance12345?session=abc&_DASH_pathway=alpha&_DASH_throughput=4880000)

Since the steering server is stateful, it knows that it last assigned "beta" but the player is now reporting "alpha" implying a client-initiated change. The steering server can take this signal in to account when making its steering decisions. It may however still reply with

```
{
  "VERSION": 1,
  "TTL": 250,
  "RELOAD-URI": "https://steeringservice.com/app/instance12345?session=abc"
  "PATHWAY-PRIORITY": ["beta","alpha"]
}
```

Since the client has excluded "beta" for performance reasons for 250s since the switch was made, it processes the `PATHWAY-PRIORITY` array as if "beta" were not present and continues to play "alpha". At the next steering server response, the exclusion would have expired and the client should apply the conventional processing rules to the response.

## A.2 Advanced steering example

This case illustrates the independent steering of manifests, media segments and advertising content.

A DASH MPD is presented to a player.

```
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:mpeg:dash:schema:mpd:2011"
xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd" type="dynamic"
minimumUpdatePeriod="PT30S" timeShiftBufferDepth="PT30M" availabilityStartTime="2022-02-
25T12:30:00" minBufferTime="PT4S" profiles="urn:mpeg:dash:profile:isoff-live:2011">
  <Location serviceLocation="1234">https://manifest-cdn1.com/</Location>
  <Location serviceLocation="5678">https://manifest-cdn2.com/</Location>
  <BaseURL serviceLocation="alpha">https://segments-cdn-A.com/</BaseURL>
  <BaseURL serviceLocation="beta">https://segments-cdn-B.com/</BaseURL>
  <Period id="Primary-Content-1">
    ...
  </Period>
  <Period id="Ad-break-1">
    <BaseURL serviceLocation="ad1">https://ad-server-1.com/</BaseURL>
    <BaseURL serviceLocation="ad2">https://ad-server-2.com/</BaseURL>
    ...
  </Period>
  <Period id="Primary-Content-2">
    <BaseURL serviceLocation="gamma">https://segments-cdn-C.com/</BaseURL>
    <BaseURL serviceLocation="delta">https://segments-cdn-D.com/</BaseURL>
    ...
  </Period>
  <Period id="Ad-break-2">
    <BaseURL serviceLocation="ad3">https://ad-server-3.com/</BaseURL>
    <BaseURL serviceLocation="ad4">https://ad-server-4.com/</BaseURL>
    ...
  </Period>
  <Period id="Primary-Content-3">
    ...
  </Period>
  <ContentSteering defaultServiceLocation="1234,alpha,ad1">
    https://steeringservice.com/app?token=567
  </ContentSteering>
</MPD>
```

Since `ContentSteering@queryBeforeStart` is not `TRUE`, the player would start by matching the content of the `ContentSteering@defaultServiceLocation` elements against the available `Location` and `BaseURL` options. As a result, it would start with

1. Load media segments within period "primary-content-1" from <https://segments-cdn-A.com>.
2. Refresh the manifest from <https://manifest-cdn1.com/>

As soon as playback has started and the player's target buffer has been reached, it would make the following request to the content steering server:



`https://steering-service.com/app?token=567&_DASH_pathway="1234,alpha"&_DASH_throughput=32000000,19000000`

Assume the steering server then returns the following JSON response for all future refreshes:

```
{
  "VERSION": 1,
  "TTL": 300,
  "RELOAD-URI": "https://steering-service.com/app/instance1234"
  "PATHWAY-PRIORITY":
  ["beta", "alpha", "ad1", "delta", "5678", "gamma", "1234", "ad4", "ad3"
]
}
```

Following the rules of selecting the highest matching **Location** and **BaseURL** serviceLocation from the PATHWAY-PRIORITY array, along with **BaseURL** inheritance, the client would:

3. Switch to loading media segments within period “primary-content-1” from <https://segments-cdn-B.com>.
4. Switch to loading the manifest from <https://manifest-cdn2.com/>
5. Load media segments within period “ad-break-1” from <https://ad-server-1.com/>
6. Load media segments within period “primary-content-2” from <https://segments-cdn-D.com>
7. Load media segments within period “ad-break-2” from <https://ad-server-4.com>
8. Load media segments within period “primary-content-3” from <https://segments-cdn-B.com>

Assuming that the second steering server request (after 300s) occurs during period “Primary-Content-2”, then client would construct the request in the following manner:

`https://steering-service.com/app/instance1234?_DASH_pathway="5678,beta,ad1,delta"&_DASH_throughput=450000,56000000,21000000,32000000`

## A.3 Pathway cloning example

This case illustrates the application of Pathway Cloning to synthesize a new serviceLocation. Interaction with Annex I - Flexible Insertion of URL Parameters is also shown to provide clarity on how the query arguments should be appended.

A DASH MPD is presented to a player at the URL

`http://www.example.com/dash/cloning.mpd?token=1234`

with the following content:

```
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="urn:mpeg:dash:schema:mpd:2011"
xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd" type="dynamic"
minimumUpdatePeriod="PT30S" timeShiftBufferDepth="PT30M" availabilityStartTime="2022-02-
25T12:30:00" minBufferTime="PT4S" profiles="urn:mpeg:dash:profile:isoff-live:2011">
  <BaseURL serviceLocation="alpha">https://cdn1.com/</BaseURL>
  <BaseURL serviceLocation="beta">https://cdn2.com/</BaseURL>
  <EssentialProperty schemeIdUri="urn:mpeg:dash:urlparam:2014"
xmlns:up="urn:mpeg:dash:schema:urlparam:2014">
    <up:UrlQueryInfo includeInRequests="mpd segment steering" queryTemplate="$querypart$"
useMPDUrlQuery="true"/>
  </EssentialProperty>

  <Period id="1">
    <AdaptationSet mimeType="video/mp4" contentType="video" subsegmentAlignment="true"
subsegmentStartsWithSAP="1" par="16:9">
      <SegmentTemplate duration="120" timescale="30"
media="$RepresentationID/$RepresentationID$ $Number$.m4v?geo=US" startNumber="1"
initialization="$RepresentationID/$RepresentationID$.m4v?geo=US"/>
      <Representation id="1024x576_2500k" codecs="avc1.64001f" bandwidth="3134488" width="1024"
height="576" frameRate="30" sar="1:1" scanType="progressive"/>
      ...
    </AdaptationSet>
  </Period>
```

```
<ContentSteering defaultServiceLocation="beta"
  queryBeforeStart="true">https://steeringservice.com/app/instance1234?sessionID=64829</ContentSteering>
</MPD>
```

The player would parse the **ContentSteering** element upon receiving the MPD. Since the `@queryBeforeStart` attribute is present and set to `true`, instead of starting playback using the `@defaultServiceLocation` of "beta", it would make a request to the steering server. This request, after applying the rules of Annex I, would be

<https://steeringservice.com/app/instance1234?sessionID=64829&token=1234>

Note the `_DASH_` params are not attached to this request since the player has not yet started playback. The server may then return the JSON response below:

```
{
  "VERSION": 1,
  "TTL": 300,
  "RELOAD-URI": "https://steeringservice.com/app/instance12345?session=abc"
  "PATHWAY-PRIORITY": ["charlie", "alpha", "beta"]
  "PATHWAY-CLONES": [
    {
      "BASE-ID": "alpha",
      "ID": "charlie",
      "URI-REPLACEMENT": {
        "HOST": "segments-cdn-charlie.com",
        "PARAMS": {
          "token-for-charlie": "dkfs1239414"
        }
      }
    }
  ]
}
```

The player would recognize that the highest priority `serviceLocation` specified is "charlie", which is a synthesized `serviceLocation` built via cloning. The player would parse the `PATHWAY-CLONES` array to locate the definition of "charlie". It would construct "Charlie" by building the URI for "alpha" as it would normally, but then substituting the `HOST` component of the URL with "segments-cdn-charlie.com" and also appending the query argument "token-for-charlie". The first request for a media segment would be:

[https://segments-cdn-charlie.com/1024x576\\_2500k/1024x576\\_2500k\\_1.m4v?geo=US&token=1234&token-for-charlie=dkfs1239414](https://segments-cdn-charlie.com/1024x576_2500k/1024x576_2500k_1.m4v?geo=US&token=1234&token-for-charlie=dkfs1239414)

The player would then set a timer so that in 300s, when the throughput it was estimating is 5.14Mbps, it would again query the steering server, with the URL

[https://steeringservice.com/app/instance12345?session=abc&token=1234&\\_DASH\\_pathway=charlie&\\_DASH\\_throughput=5140000](https://steeringservice.com/app/instance12345?session=abc&token=1234&_DASH_pathway=charlie&_DASH_throughput=5140000)

The cycle would then continue with the player parsing the steering server response and making the appropriate pathway selection.

---

## Annex (informative): Change History

Date	Version	Information about changes
2022-07-10	0.9.0	Version published for community review.
2022-12-07	0.9.5	Updates following community review.
2022-12-21	0.9.6	Version submitted for second community review
2023-03-31	0.9.7	Version addressing the comments from the second community review
2023-05-08	0.9.8	Version prepared for IPR Review and ETSI submission
2023-06-12	1.0.0	Version published on DASH-IF web site