**ISO/IEC JTC 1/SC 29/WG 03 N0786**

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

**Document type:** Output Document

**Title:** Technologies under Consideration for OMAF

**Status:** Approved

**Date of document:** 2023-01-27

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

**No. of pages:** 12 (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 N0786**

**January 2023, Online**

|  |  |
| --- | --- |
| **Title** | **Technologies under Consideration for OMAF** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **22310** |

# Introduction

This document includes technologies under consideration for ISO/ IEC 23090-2 (OMAF)

This includes the following:

1. Server-side dynamic adaptation architecture (m61074)
2. Adaptation parameters for server-side adaptation (m61226)
3. Adaptation parameters for foveated rendering (m61227)
4. Server-side dynamic adaptation diagram for study (m62082)

# Server-side Dynamic Adaptation Architecture

The contribution m61074 proposes a generalized OMAF architecture by enabling SSDA that satisfies not only spatial adaptation with viewport dependent media processing with bitstream rewriting, but also other adaptations of the SSDA requirements in MPEG-I pahse2 requirements.

## Introduction

Figure 1 is OMAF architecture for SSDA which modified block diagram for OMAF content flow process to include dynamic adaptation service. This architecture was proposed and accepted into TuC[1].

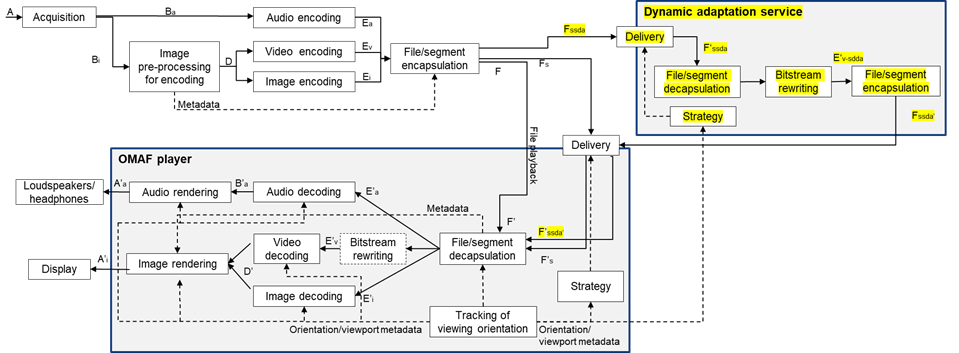


Figure 1 OMAF architecture in TuC

There were below comments in the previous meeting and described in the meeting minutes[2].

* *A Git comment about this document suggests that architecture should be further updated to reflect generality of SSDA and consider viewport-dependent media processing as an example. It was asked to clarify the proposed text and it was suggested that some editorial improvement should be made to the text.*
* *It was commented that some additional modifications would be needed in OMAF to support SSDA.*
* *It was asked why delivery, file/segment decapsulation and bitstream rewriting is needed in dynamic adaptation service.*
* *It was asked and suggested that bitstream rewriting can be optional on the server side.*

The architecture in Figure 1 was specific architecture for SSDA because it was only for the dynamic adaptive server that handles the functions of quality selection of tile tracks and merging bitstreams from tile tracks for content with a tile-based viewport-dependent profile.

These comments are due to the specific SSDA architecture.

## Proposal

There are requirements for SSDA in MPEG I pahse2 requirement[3].

1. *(Parameters) The specification shall support definition, capture and communication of dynamic content adaptation parameters from an immersive content consumption client and its underlying delivery network to a delivery server in a timely manner, if server-side dynamic adaptation is needed for:* 
   * *bitrate adaptation (e.g., DASH representation switching),*
   * *temporal adaptation (e.g., trick plays),*
   * *spatial adaptation (e.g., viewport/viewpoint dependent media processing, especially for V3CD immersive media content), and*
   * *content adaptation (e.g., NBMP pre-rendering and storyline selection).*
2. *(Architecture) The specification shall support the content delivery server to conduct dynamic adaptation based on adaptation parameters collected from the client and underlying delivery network, when server-side dynamic adaptation is needed.*

The architecture in Figure 1 is a spatial adaptation with viewport dependent media processing with bitstream rewriting.

Therefore, for general SSDA, it should enable other adaptations of the SSDA requirements in MPEG-I pahse2 requirements. It does not require bitstream rewriting for this adaptation processing.

This contribution proposes a generalized OMAF architecture, which enable only handing the function of quality selection for such as bitrate and spatial adaptation requirements, by changing the blocks related to bitstream rewriting to optional in Figure 1.

## Proposed Text

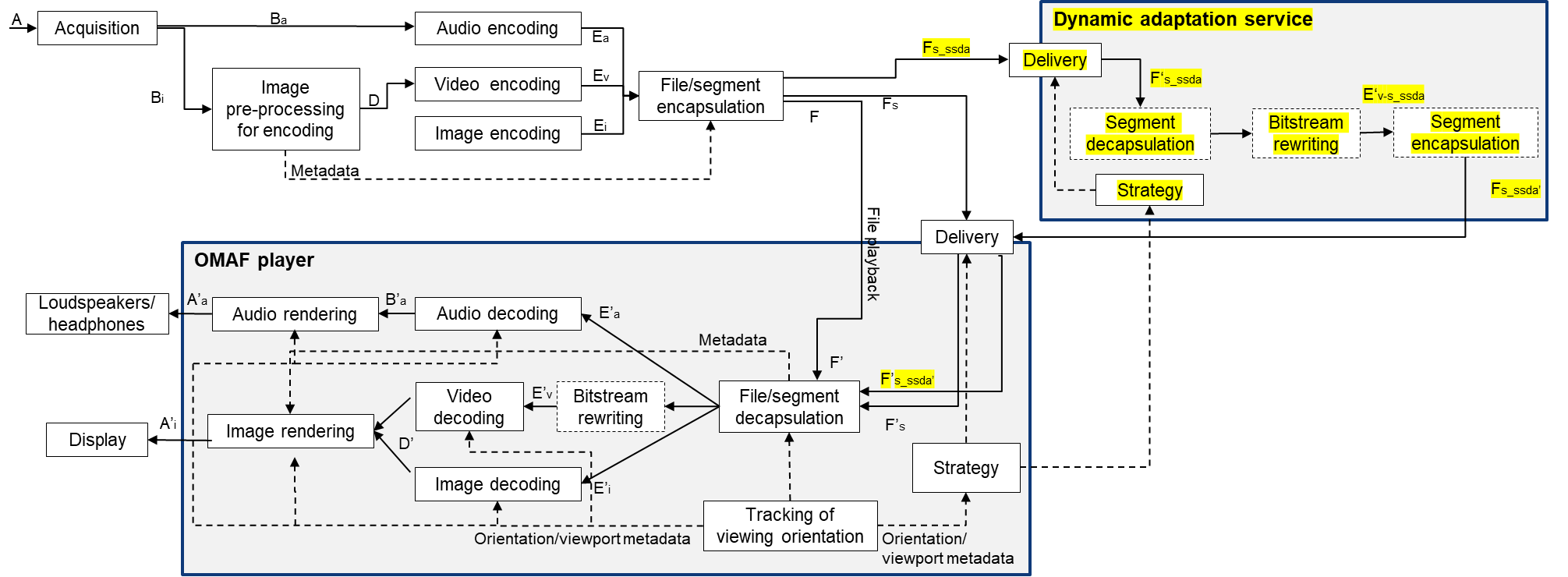
This contribution proposes an updated OMAF architecture which is described in 4.2 overall architecture of OMAF specification to deal with above SSDA.

The proposed text is marked in yellow with removed text in ~~red strike through~~ on top of the current OMAF spec

**4.2 Overall architecture**

Figure X shows a typical content flow process for an omnidirectional media application.

NOTE In this content flow process, Dynamic adaptation service is a different server process than content generation process.



**Figure X — Content flow process for omnidirectional media with projected video**

The following interfaces are specified in this document:

* E'a, E'v, E'i: audio bitstream, video bitstream, coded image(s), respectively; see Clause10.
* E'v-s\_ssda : video bitstream, see Clause 10.
* F/F': media file; see Clause7. Moreover, media profiles specified in Clause 10 include the specification of the track formats for F/F', which may contain constraints on the elementary streams contained within the samples of the tracks.
* Clause 8 specifies the delivery related interfaces for DASH delivery.
* Clause 9 specifies the delivery related interfaces for MMT delivery.

The other interfaces in Figure X are not specified in this document.

NOTE While the syntax and semantics of the bitstreams Ea, Ev, and Ei are the same as those for E'a, E'v, E'i, respectively, the input interface to the file/segment encapsulation module is not specified.

NOTE While the syntax and semantics of the bitstreams Ev, is the same as for E'v-s\_ssda the input interface to the segment encapsulation module is not specified.

A real-world audio-visual scene (A) is captured by audio sensors as well as a set of cameras or a camera device with multiple lenses and sensors. The acquisition results in a set of digital image/video (Bi) and audio (Ba) signals. The cameras/lenses typically cover all directions around the centre point of the camera set or camera device, thus the name of 360-degree video.

Audio may be captured using many different microphone configurations and stored as several different content formats, including channel-based signals, static or dynamic (i.e. moving through the 3D scene) object signals, and scene-based signals (e.g. Higher Order Ambisonics). The channel-based signals typically conform to one of the loudspeaker layouts defined in ISO/IEC 23091-3. In an omnidirectional media application, the loudspeaker layout signals of the rendered immersive audio program are binauralized for presentation via headphones.

For audio, no stitching process is needed, since the captured signals are inherently immersive and omnidirectional.

This document specifies the following types of omnidirectional video and images, which differ in the architecture in the image pre-preprocessing for encoding and in the image rendering processing blocks:

* Projected omnidirectional video/images:
* Image pre-processing for encoding: The images (Bi) of the same time instance are stitched, possibly rotated, and projected onto a 2D picture coordinates using a mathematically specified projection format. Optionally, the resulting projected pictures may be further mapped region-wise onto a packed picture. Either projected pictures or packed pictures are subject to video or image encoding.
* Image rendering: Either regions of the decoded packed pictures (if region-wise packing has been applied) or the entire projected picture (otherwise) is mapped onto a rendering mesh suitable for the projection format in use.
* Fisheye omnidirectional video/images:
* Image pre-processing for encoding: Circular images (Bi) captured by fisheye lenses are arranged onto a 2D picture, which is then input to video or image encoding.
* Image rendering: The decoded circular images are stitched using the signalled fisheye-specific parameters.
* Mesh omnidirectional video:
* Image pre-processing for encoding: A 3D mesh consisting of mesh elements is generated, where mesh elements can be either parallelograms or regions of a sphere surface. The images (Bi) of the same time instance are stitched, possibly rotated, and projected onto the 3D mesh. Mesh elements are mapped onto rectangular regions of one or more 2D pictures, which are input to video encoding.
* Image rendering: Rectangular regions of the decoded 2D picture(s) are mapped to the 3D mesh, which is used directly as the rendering mesh.

Further details of the architecture for projected, fisheye, and mesh omnidirectional video/images are provided in subclauses4.3, 4.4, and4.5, respectively.

The pre-processed pictures (D) are encoded as coded images (Ei) or a coded video bitstream (Ev). The captured audio (Ba) is encoded as an audio bitstream (Ea). The coded images, video or audio are then composed into a media file for file playback (F) or a sequence of an initialization segment and media segments for streaming (Fs, Fs\_ssda), according to a particular media container file format. In this document, the media container file format is the ISO Base Media File Format specified in ISO/IEC 14496-12. The file encapsulator also includes metadata into the file or the segments.

The segments Fs\_ssda are delivered using a delivery mechanism to Dynamic adaptation service.

In the Dynamic adaptation service, the dynamic adaptation parameters, such as bitrate, are send from the strategy module in OMAF player to the strategy module in dynamic adaptation service, which determines the tracks to be received based on the dynamic adaptation parameters. The received segments (F's\_ssda) which determines tracks by delivery mechanism are identical to the segment (Fs\_ssda’) except when merging bitstreams. Viewport-dependent video may be carried in multiple tracks, which may be processed and extract the coded video streams in segment decapsulator. The coded video streams are merged in the bitstream rewriting into a single video bitstream (E'v-s\_ssda). This single video bitstream (E'v-s\_ssda) is composed into segments for streaming (Fs\_ssda’) in segment encapsulator, according to a particular media container file format. The segments Fs\_ssda’ are delivered to a OMAF player.

NOTE When OMAF player use the dynamic adaptation service, the OMAF player can be simple process because instead dynamic adaptation service handling the adaptation and bitstream merging function, the OMAF player only handling to send the dynamic adaptation parameters.

The segments Fs are delivered using a delivery mechanism to a OMAF player.

The file that the file encapsulator outputs (F) is identical to the file that the file decapsulator inputs (F'). A file decapsulator processes the file (F') or the received segments (F's and F’s\_ssda’) and extracts the coded bitstreams (E'a, E'v, or E'i) and parses the metadata. Viewport-dependent video may be carried in multiple tracks, which may be merged in the bitstream rewriting into a single video bitstream E'v prior to decoding. The audio, video or images are then decoded into decoded signals (B'a for audio, and D' for images/video). In the image rendering block, the decoded pictures (D') are projected onto the screen of a head-mounted display or any other display device based on the metadata parsed from the file. Likewise, decoded audio (B'a) is rendered, e.g. through headphones, according to the current viewing orientation. The current viewing orientation is determined by the viewing orientation tracking functionality. When a head-mounted display is in use, the viewing orientation tracking can involve head tracking and possibly also eye tracking. When sphere-relative overlays are in use, the viewing orientation tracking functionality can include or be complemented by viewing position tracking and rendering of overlays with background visual media can take both the viewing position and the viewing orientation into account. Besides being used by the renderer to render the appropriate part of decoded video and audio signals, the current viewing orientation may also be used by the video and audio decoders for decoding optimization. In viewport-dependent delivery, the current viewing orientation is also passed to the strategy module in OMAF player, which determines the video tracks to be received based on the viewing orientation.

The process described above is applicable to both live and on-demand use cases.

# Adaptation Parameters to Support Server-Side Dynamic Adaptation

The contribution m61226 proposes lists of adaptation parameters from the previous contribution [3] for

* Track Selection and Switching, and
* Viewport and Viewpoint Selection.

As indicated in the CTA CMCD specification [5], these parameters defined in the CMCD tabular format [5] can be used as part of http requests for OMAF related media segments in MPEG DASH [4], in the forms of

* URL query parameters, and/or
* http header parameters.

It is expected that, when received at an OMAF content delivery server, OMAF content (e.g., in the form of streaming segments) will be dynamically selected and/or adapted, according to the received adaptation parameters, and delivered back to a requesting client.

## Track Selection and Switching

The list of parameters provided in the table below is a selected list of differentiating attributes specified in clause 8.10.3, “track selection box”, in the latest ISOBMFF specification [6] for the purpose of track selection or switching:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Key Name | Header Name | Type & Unit | Value Definition |
| ~~Codec~~ | ~~cdec~~ | ~~SSDA-Object~~ | ~~String, 4CC~~ | ~~Sample Entry (in SampleDescriptionBox of media track)~~ |
| Screen width | scsw | SSDA-Request | Unsigned Integer, pixels | Width field of VisualSampleEntry. |
| Screen height | scsh | SSDA-Request | Unsigned Integer, pixels | Height field of VisualSampleEntry. |
| ~~Max packet size~~ | ~~mpsz~~ | ~~SSDA-Request~~ | ~~Unsigned Integer, bits~~ | ~~Maxpacketsize field in RtpHintSampleEntry~~ |
| ~~Media type~~ | ~~mtyp~~ | ~~SSDA-Object~~ | ~~Unsigned Integer, 4CC~~ | ~~Handler\_type in HandlerBox (of media track)~~ |
| ~~Media language~~ | ~~mela~~ | ~~SSDA-Object~~ | ~~String, 3-character code~~ | ~~Language field in MediaHeaderBox~~ |
| Bitrate | bitr | SSDA-Request | Integer, kbps | Total size of the samples in the track divided by the duration in the TrackHeaderBox |
| Frame rate | frar | SSDA-Request | Integer, fps | Number of samples in the track divided by duration in the TrackHeaderBox |
| Number of views | nvws | SSDA-Object | Integer | Number of views in the track |

With these parameters in DASH http requests for segments related to a track selection and switching, it is expected to return an http response containing:

* a segment from a track matching the requirements from the parameters, selected or produced with server-side dynamic adaptation, or
* a MIME multipart message of segments from tracks matching the requirements from the parameters, selected or produced with server-side dynamic adaptation.

## Viewport and Viewpoint Selection

The list below is a collection of viewpoint and viewport related adaptation parameters from OMAF [3] (Clause 7.5.6 Sphere region structure). Collectively, the three centre parameters define an adaptation request for a viewpoint, and they define an adaptation request for a viewport (as a sphere region defined by two big) together with the two range parameters. The shape type parameter defines (currently as in the OMAF 3rd Edition [2]) whether the sphere region is formed by four great circles or by two azimuth circles and two elevation circles.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Key Name | Header Name | Type & Unit | Value Definition |
| Centre Azimuth | 'azim' | SSDA-Request | Integer, degrees | Azimuth component (centre\_azimuth) of a sphere region |
| Centre Elevation | 'elev' | SSDA-Request | Integer, degrees | Elevation component (centre\_elevation) of a sphere region |
| Centre Tilt | ‘tilt’ | SSDA-Request | Integer, degrees | Tilt component (centre\_tilt) of a sphere region |
| Azimuth range | 'azrg' | SSDA-Request | Integer, degrees | Azimuth range of a sphere region |
| Elevation range | 'elrg' | SSDA-Request | Integer, degrees | Elevation range of a sphere region |
| Shape Type | ‘styp’ | SSDA-Request | Integer | Shape type (0 or 1) of a sphere region |

With these parameters in DASH http requests for segments related to a viewpoint and/or viewport, it is expected to return an http response containing:

* a viewpoint/viewport segment, selected or produced with server-side dynamic adaptation, or
* a package of tile/sub-picture segments in a MIME multipart message for constructing a viewpoint/viewport segment, just like in the viewport-dependent media processing, but whose content has been adapted for the viewpoint/viewport separately carried in the tile/sub-picture segments.

# Adaptation Parameters for Foveated Rendering

The contribution m61074 proposes adaptation parameters for foveated rendering.

In the server-side dynamic adaptation in OMAF TuC [1], the client can make some static selection (such as those related to video codec profile, screen size and encryption algorithm), and only leave dynamic adaptation (such as those related to video bitrate, network bandwidth) to the server, by collecting and passing dynamic adaptation parameters needed for Adaptation Logic to the server as part of (http) segment requests. The communication of these adaptation parameters can be implemented in anyone of the following mechanisms:

1. URL query parameters
2. HTTP header parameters

To enable foveated rendering at the server-side and to reduce delivery loads for foveated viewports, a list of parameters is defined in the following table, which can be used as suggested in the CTA WAVE Common Media Client Data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Description | Key Name | Header Name | Type & Unit | Value Definition |
| Centre Azimuth | 'azim' | SSDA-Request | Integer, degrees | Azimuth component (centre\_azimuth) of a sphere region |
| Centre Elevation | 'elev' | SSDA-Request | Integer, degrees | Elevation component (centre\_elevation) of a sphere region |
| Centre Tilt | ‘tilt’ | SSDA-Request | Integer, degrees | Tilt component (centre\_tilt) of a sphere region |
| Azimuth range | 'azrg' | SSDA-Request | Integer, degrees | Azimuth range of a sphere region |
| Elevation range | 'elrg' | SSDA-Request | Integer, degrees | Elevation range of a sphere region |
| Shape Type | ‘styp’ | SSDA-Request | Integer | Shape type (0 or 1) of a sphere region |
| Foveal Levels | ‘flvl | SSDA-Request | Integer | Number of foveated area levels. Usually, it is equal to   * 0: foveal only, implying no foveated rendering, * 1: foveal and peripheral, * 2: foveal, blended and peripheral, * n>2: foveal, n-2 blended and peripheral.   For all foveated areas signalled within a foveated viewport, they share the viewpoint (specified by the three centre parameters), and each of them has its additional range parameter(s) and one quality ranking parameter. Particularly,   * for each level of foveated area of spherical region shape, a list of Azimuth range, Elevation range, and quality ranking shall be provided. * for each level of foveated area of circular region shape, a list of Azimuth range and Quality Ranking shall be provided. |
| Foveal Shape | 'fovs' | SSDA-Request | Integer | Foveal shape: 0 = rectangular, 1 = circle, others = reserved. |
| Quality Ranking | ‘qrnk’ | SSDA-Request | Integer | Quality ranking of a foveated area or region.  Note that, if more appropriate, this quality ranking parameter can be replaced or supplemented with another parameter for a bitrate in units of kbps. |

With these parameters in DASH http requests for segments related to a foveated viewport, it is expected to return an http response containing:

1. a foveated viewport segment; this is a server-side adaptation or rendering, or
2. a package of tile/sub-picture segments in a MIME multipart message for constructing a foveated viewport segment, just like in the viewport-dependent media processing, but whose content has been adapted with foveated rendered content separately carried in the tile/sub-picture segments.

# Server-side Dynamic Adaptation Diagram for Study

In the contribution m62082 Server-side dynamic adaptation architecture diagram is proposed to be updated by taking some components from the diagram shown below (in particular the “adaptation logic” piece on the server side.)

Server-side dynamic adaptation (SSDA), on the other hand, as complementary to client-side dynamic adaptation (CSDA) suggests that some dynamic adaptation can be performed at the server side, instead of at the client side, as shown in the following diagram:



It should be noted that in an SSDA scheme, the Streaming Client can still make some static selection (such as those related to video codec profile, screen size and encryption algorithm), and only leave dynamic adaptation to the server, by collecting and passing dynamic adaptation parameters needed for Adaptation Logic to the server as part of (http) segment requests. Moreover, in a hybrid mode, SSDA and CSDA schemes can be jointly implemented to share dynamic adaptation tasks split between the Streaming Client and Server. In any case, the communication of adaptation parameters for the Server can be implemented in anyone of the following mechanisms:

1. URL query parameters
2. HTTP header parameters

----