YANA23
A curious mixture of stuff we should work on and questions we should answer.

Will Law
Chief Architect
Akamai

Prague, November 2023

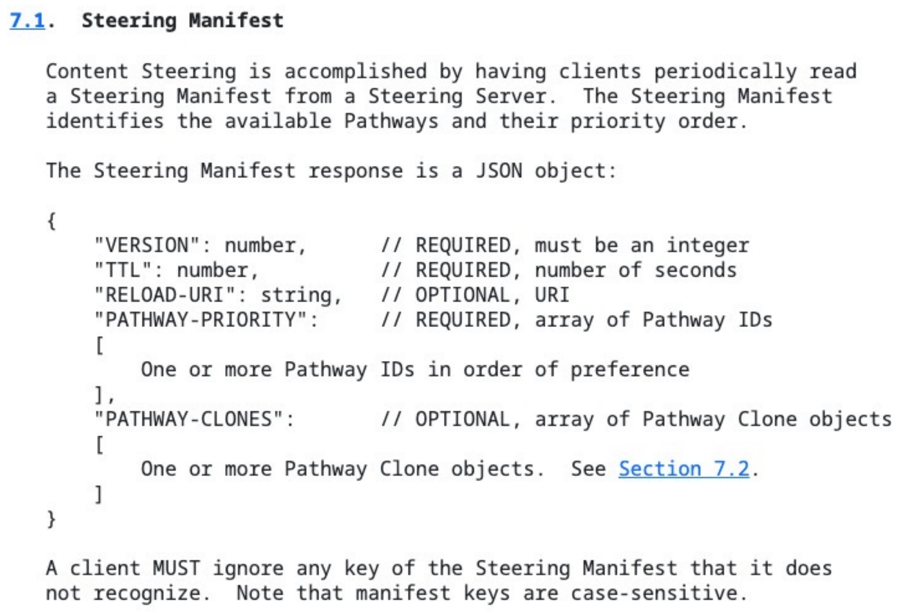
I’d like to cover

- The motivation for a new Content Steering Specification
- Should CMCD be expanded to be a generic QoE reporting mechanism?
- Should we collectively push MMS?
- Why I’m interested in MoQ.
- It’s 2024, looking back what did we get right?
- It’s 2024, looking back what did we get wrong?
- It’s 2028, looking back what did we get right? (the wish list).

Akamai Experience the Edge

The motivation for a separate Content Steering Spec

Both HLS and DASH define content steering, both in terms of how steering servers should operate, what their response looks like and how the clients should react.



7.1. **Steering Manifest**

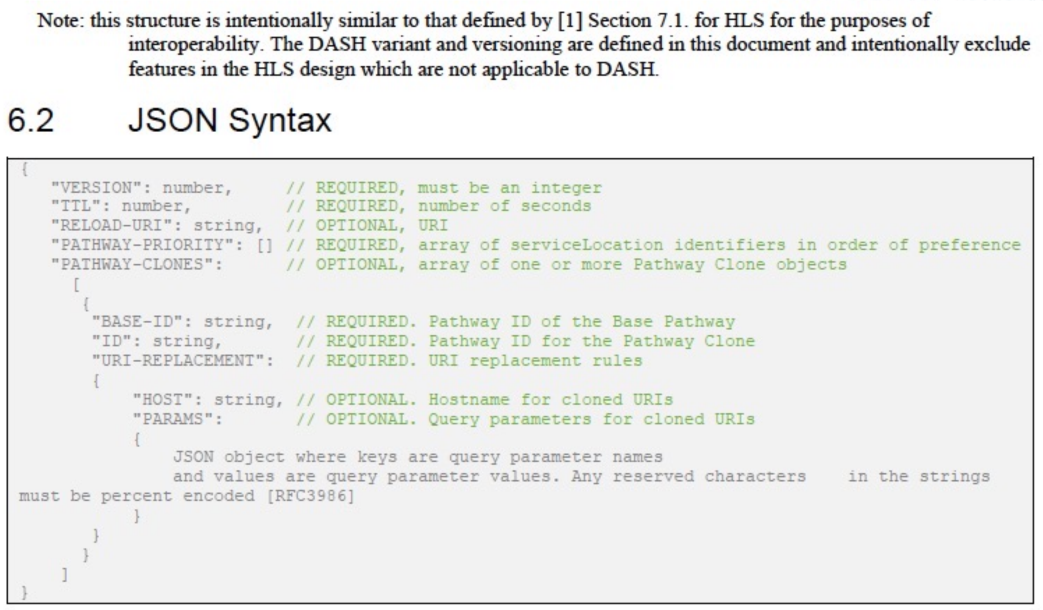
```
Content Steering is accomplished by having clients periodically read
a Steering Manifest from a Steering Server.  The Steering Manifest
identifies the available Pathways and their priority order.

The Steering Manifest response is a JSON object:

{
    "VERSION": number,      // REQUIRED, must be an integer
    "TTL": number,          // REQUIRED, number of seconds
    "RELOAD-URI": string,   // OPTIONAL, URI
    "PATHWAY-PRIORITY":     // REQUIRED, array of Pathway IDs
    [
        One or more Pathway IDs in order of preference
    ],
    "PATHWAY-CLONES":       // OPTIONAL, array of Pathway Clone objects
    [
        One or more Pathway Clone objects.  See Section 7.2.
    ]
}

A client MUST ignore any key of the Steering Manifest that it does
not recognize.  Note that manifest keys are case-sensitive.
```

From https://datatracker.ietf.org/doc/html/draft-pantos-hls-rfc8216bis-13#page-71



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]
           }
       }
      }
     ]
}
```

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

Akamai Experience the Edge

Problems with this

Content distributors today want to steer populations of players across both HLS and DASH.

The HLS spec defines a mixture of server response and client behavior. There is a mirror specification in the DASH world.

Someone building a steering service must reverse engineer the necessary behavior and properties of the service itself.

Several ambiguities remain unanswered

- are additional params allowed?
- behavior with redirects?
- how to report multiple pathways and throughputs since the last request?

Akamai Experience the Edge

We discussed this at DASH IF and at the HLS Interest meeting in Cupertino two week ago.

y topics
adata, content steering, CMCD
Playlist signaling of embedded event streams
DASH-HLS interoperability: Event binding
Proposal: Separate specification for content steering services
DASH-HLS interoperability: Content steering
Adding CMCD support to Apple players

Akamai Experience the Edge

Proposal for a separate Content Steering specification.

Create a separate content steering specification which normatively defines

- the JSON response
- How steering should be performed for QoE or content selection reasons
- How generic clients are expected to behave
- Allowed response types, error conditions, enriched player data etc.

The HLS & DASH specs would then reference this specification and define the carriage of the steering server URL within their particular format.

Akamai Experience the Edge

My questions for you

1. Where to create this spec?
   - At IETF, as an RFC
     - The best workgroup fit would be Media OPerationS (mops)
       - Is this within their charter scope?
       - Would they need a BoF to begin work like this?
     - HLS spec is already an IETF draft
   - At CTA WAVE project
     - Focused on standards to improve OTT interop (CMCD, CMSD and HDMI before that.
     - WAVE is chaired by Apple
   - Somewhere else?

2. Who is willing to be the champion and driver of this?

CMCD (Common Media Client Data)

CMCD has seen the fastest adoption of any CTA WAVE spec.

Now supported on all major player with AVPlayer in the works.

V1 focused on conciseness – it has just 18 fields that it can report.

Goals were two-fold

- Allow merged player health and CDN logs to be collected together
- Allow CDNs to optimize their delivery based on media object type.

Encoded bitrate - **br**	Object type - **ot**
Buffer length - **bl**	Playback rate - **pr**
Buffer starvation - **bs**	Requested max tput - **rtp**
ContentID - **cid**	Streaming format - **sf**
Object duration - **d**	Session ID - **sid**
Deadline - **dl**	Stream type - **st**
Measured throughput - **mtp**	Startup - **su**
Next object request - **nor**	Top bitrate - **tb**
Next range request - **nrr**	Version - **v**

Akamai *Experience the Edge*

FOMS and Demuxed discussions raised a potential new objective for CMCDv2

Issue #113 – "*Definition around use of CMCD for beacons enabling out-of-band QOE Reporting*"

Basically, extend CMCD to become the primary collection mechanism for player QoE data by

- Increasing the number of fields available
- Creating a mode whereby the player would beacon data back to a traditional collection point at periodic intervals.

Discuss: should CMCD be expanded to enable out-of-band QOE Reporting?

1. Reasons to expand CMCD
   1. A common player data standard, implemented by all players, removes the need for custom client implementations by every QoE vendor.
   2. It's easier to have a single beacon for multiple CDNs than it is to collect and process logs from multiple CDNs.
   3. Solves issue when CDN is not up to collect data.
   4. Interval based reporting.

2. Reasons not to expand CMCD
   1. You can never add enough fields to make every analytics provider happy. They will end up writing their own plug-in anyway.
   2. Beaconing data takes the CDN performance benefit out of the picture, unless you dual post, which is inefficient.
   3. Privacy issue if we start collecting more data?
   4. Cost of sending more data.

Akamai