

# DASH Content Steering at Scale

**Will Law**  
Chief Architect  
**Akamai**



**Yuriy Reznik**  
VP Research  
**Brightcove**



**Media Web Symposium (MWS'23)**  
June 2023  
Berlin, Germany

# Content steering is a bit of a misnomer



We're not actually steering the content.

In fact, there is only one version of the content.

We are steering between **CDNs** that host and serve the content.

So content steering **is the art and science of switching CDNs**

# Why do content distributors want to switch CDNs?

- **Performance**

- CDN performance and capacity varies dramatically with AS and time.
- Don't start new users and don't keep existing users on a poorly performing CDN

- **Capacity**

- Switch users away midstream if a CDN is developing capacity problems due to competing traffic.

- **Popularity**

- COGS reductions by keeping popular content on lowest cost CDNs

- **Contractual commits**

- CDN A gets 35% of traffic
- CDN B gets 65% of traffic

- **Cost**

- Price can vary by region and even time of day.

# Real Life Example



End user in Santiago Chile via Telefonica

- 51ms latency to Shield BR
- 157ms latency to Shield US-East

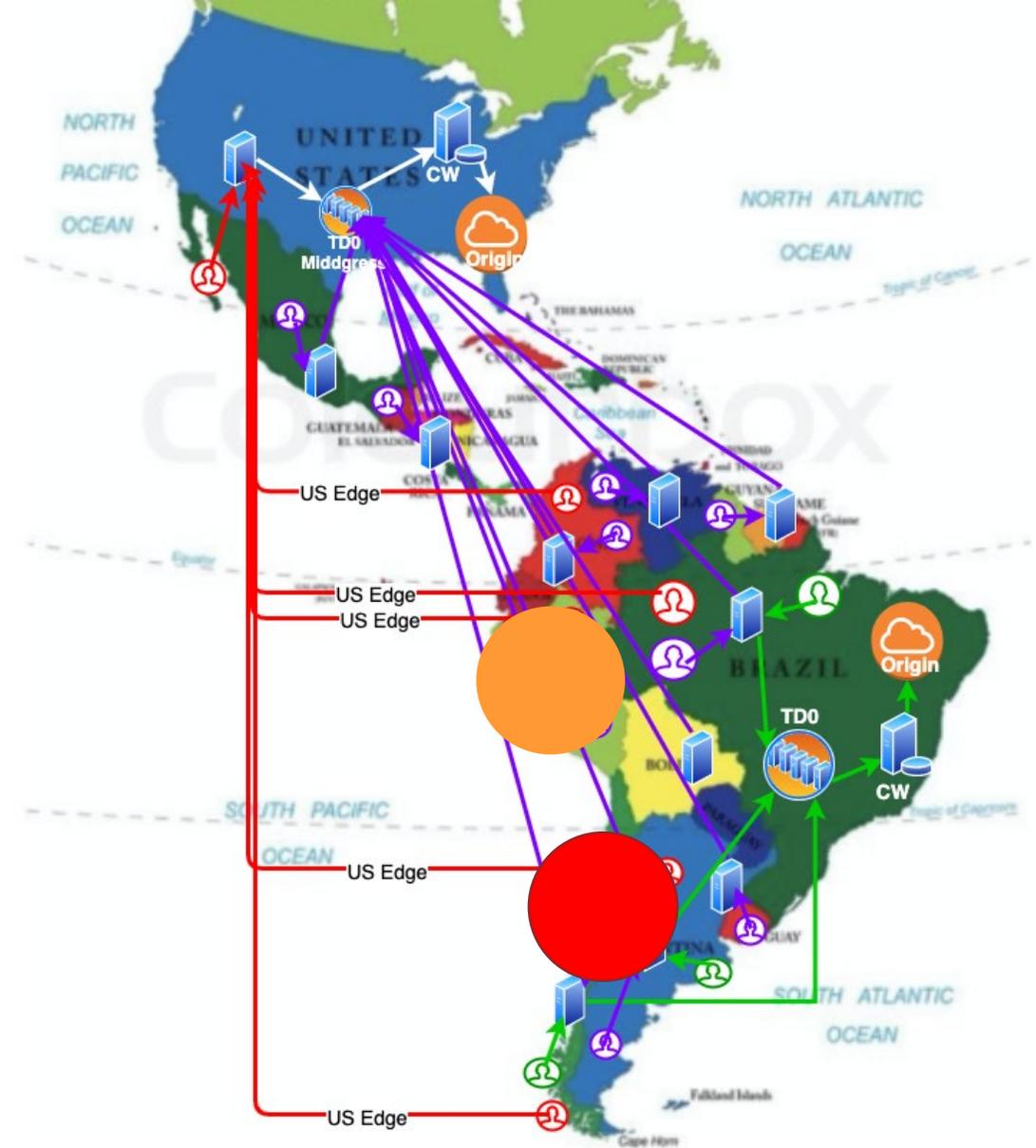


End user in Lima Peru via Claro

- 127ms latency to Shield BR
- 71ms latency to Shield US-East

\*latency represented by ping times

\*Image credit Dylan Armajani



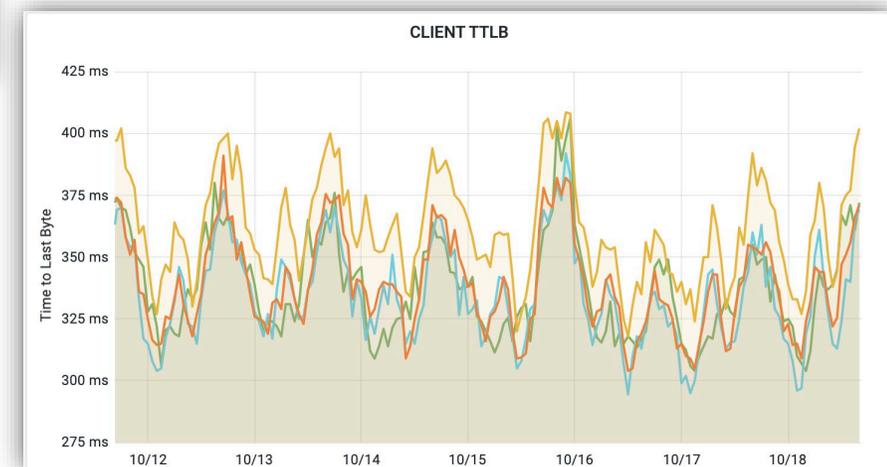
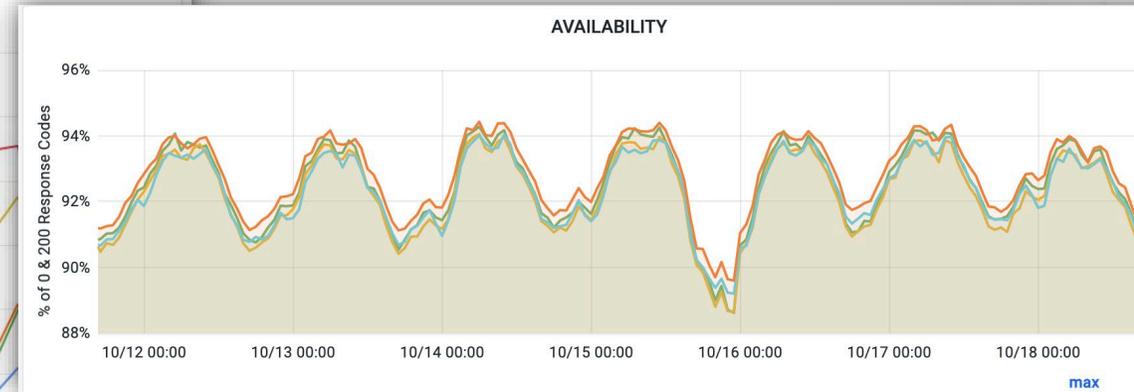
-  Users served in LATAM from Brazil TD, CW and Origin (only some users in Argentina, Chile and Brazil)
-  Users mapped to Edge servers in country but served from TD0, CW and Origin in US
-  Users mapped outside to US Edge servers and served from TD0, CW and Origin in US

# CDN performance does vary by time

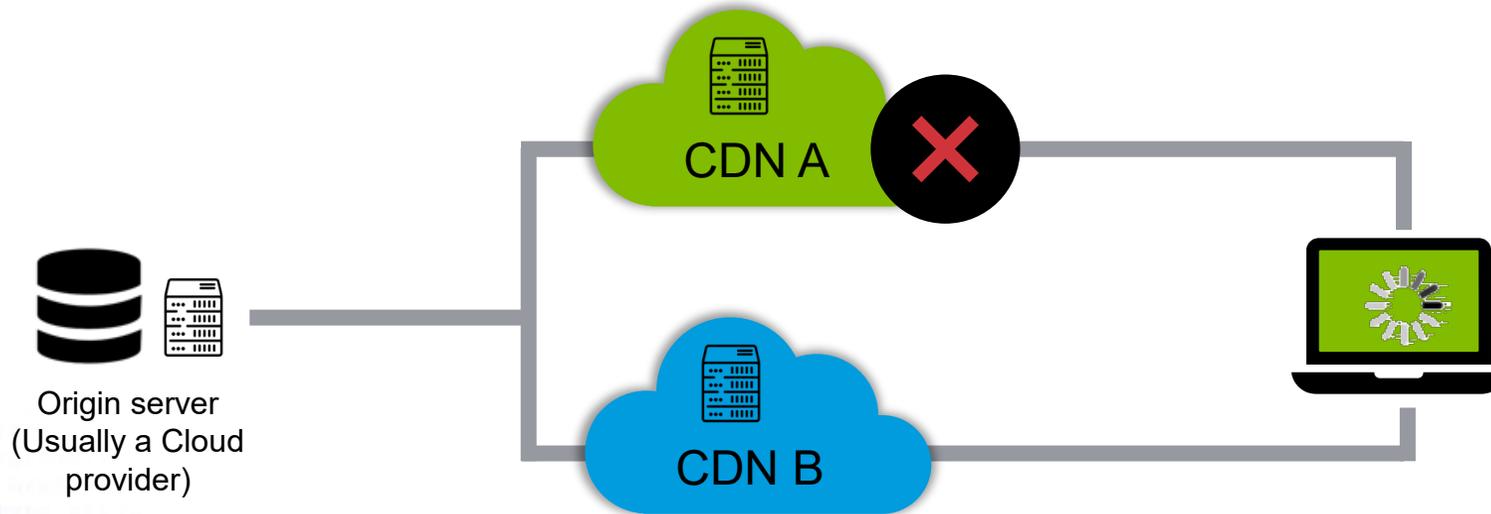


[Image data courtesy of Paramount]

For best performance during this 10-minute window, switch to blue CDN and away from orange



# But DASH already has BaseURL elements ..



<https://content.com/manifest.mpd>

```
<MPD ... >  
<Period>  
  <BaseURL>  
    https://cdnA.com  
  </BaseURL>  
  <BaseURL>  
    https://cdnB.com  
  </BaseURL>  
  ...  
</Period>  
</MPD>
```

# DASH Player failover logic is indeterminate

When a player should switch to an alternate BaseURL is left **to the discretion of the player.**

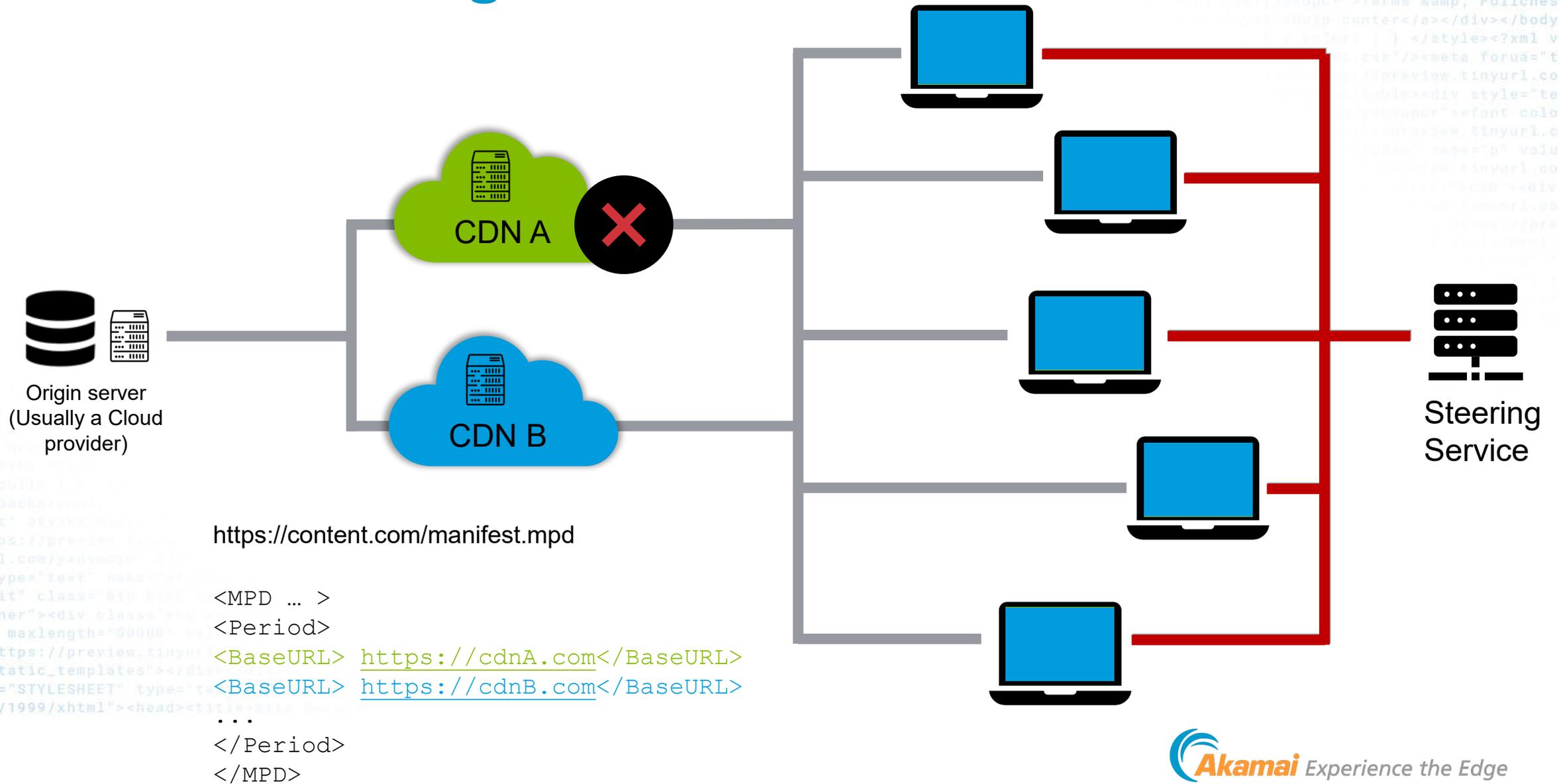
All sorts of behaviors exist in the wild:

- After receiving a 404, retry 3 times at the same bitrate then switch down to a lower bitrate
- After receiving a 404, switch bitrates immediately
- After receiving a 404, switch to an alternate BaseURL immediately at the same bitrate,
- After receiving a 404, re-request content indefinitely as long as buffer is above threshold.

**Behavior in response to slowness and lower buffer levels is completely undefined.**

**Large player populations are not deterministically controllable in the face of CDN slowness.**

# Content Steering



# Who did the work to standardize DASH Content Steering?



- Founded in 2012 to promote and catalyze the adoption of MPEG-DASH and transition it from a standard to a deployed ecosystem.
- With more than 60 members, DASH-IF represents a large footprint of the ecosystem:
  - Service providers and broadcasters
  - CDN and client implementors
  - Technology providers
- DASH-IF serves as the point of contact/coordinator for other standards organizations using DASH-based distribution.

## Charter Members



## Contributor Members



## Associate Members



# How do we do it?

```
<MPD ... >
```

```
<Period>
```

```
<BaseURL serviceLocation="alpha">https://cdnA.com</BaseURL>
```

```
<BaseURL serviceLocation="beta">https://cdnB.com</BaseURL>
```

```
...
```

```
</Period>
```

```
<ContentSteering defaultServiceLocation="beta"  
queryBeforeStart="false">https://steeringservice.com/app/inst  
ance1234</ContentSteering>
```

```
</MPD>
```

*Note that the ContentSteering element is the last element in the manifest*

# JSON Steering server response for DASH

```
{  
  "VERSION": number, REQUIRED – must be an integer  
  "TTL": number, REQUIRED – number of seconds  
  "RELOAD-URI": "https://server.com/steer", OPTIONAL – URI  
  "PATHWAY-PRIORITY": [ "CDN", REQUIRED, array of serviceLocation identifiers  
  "PATHWAY-CLONES": [ { "BA", OPTIONAL, instructions to clone new serviceLocations . . } ]  
}
```

# Player workflow by example – step 1

```
<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.example.com/</BaseURL>
```

```
<BaseURL serviceLocation="beta">https://cdn2.example.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</ContentSteering>
```

```
</MPD>
```

# Player workflow by example – step 2

Player makes a request to

```
https://steeringservice.com/app/instance1234
```

Note that the `_DASH_` parameters are not attached to this request, since the player has not yet started playback. The server responds with:

```
{
  "VERSION": 1,
  "TTL": 300,
  "RELOAD-URI": "https://steeringservice.com/app/instance1234?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.example.com/` as it begins to request content. It also sets a timer for 300s.

# Player workflow by example – step 3

After 300s, the player makes a request to

`https://steering.service.com/app/instance1234?session=abc&_DASH_pathway=alpha&_DASH_throughput=5140000`

Note the two special reserved query args

`_DASH_pathway`: specifies the current serviceLocation

`_DASH_throughput`: specifies the current throughput in bps.

The use of `_DASH_` differentiates the response from HLS players, which use `_HLS_`.

```
{
  "VERSION": 1,
  "TTL": 250,
  "RELOAD-URI": "https://steering.service.com/app/instance1234?session=abc"
  "PATHWAY-PRIORITY": ["beta", "alpha"]
}
```

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

# Can we steer ads separately from content? – Yes!

```
{  
  "VERSION": 1,  
  "TTL": 300,  
  "RELOAD-URI":  
  "https://steeringservice.com/app/1234"  
  "PATHWAY-PRIORITY":  
  ["segments2", "ad3", "ad2", "segments1", "ad1", "ad4"]  
}
```

As it enters each period, the player simply re-evaluates the BaseURL priorities.

<MPD ... >

<BaseURL serviceLocation="segments1">http://cdn1.com/</BaseURL>

<BaseURL serviceLocation="segments2">http://cdn2.com/</BaseURL>

<Period ... (primary content)/>

<Period ... (ad)>

<BaseURL serviceLocation="ad1">http://adcdn1.com/</BaseURL>

<BaseURL serviceLocation="ad2">http://adcdn2.com/</BaseURL>

...

</Period>

<Period ... (primary content)/>

<Period ... (ad)>

<BaseURL serviceLocation="ad1">http://cdn1.com/</BaseURL>

<BaseURL serviceLocation="ad2">http://cdn2.com/</BaseURL>

...

</Period>

<Period ... (primary content)/>

<Period ... (ad)>

<BaseURL serviceLocation="ad3">http://adcdn1.com/</BaseURL>

<BaseURL serviceLocation="ad4">http://adcdn4.com/</BaseURL>

...

</Period>

<ContentSteering>https://steeringservice.com/app/1234</ContentSteering>

</MPD

# Can we steer the manifest refreshes themselves?

Yes – by adding a serviceLocation identifier to the existing <Location> element.

```
<MPD ... >
```

```
<BaseURL serviceLocation="segments1">http://cdn3.com/</BaseURL>
```

```
<BaseURL serviceLocation="segments2">http://cdn4.com/</BaseURL>
```

```
<Location serviceLocation="mpd1">https://ssai1.com/manifest.mpd</Location>
```

```
<Location serviceLocation="mpd2">https://ssai2.com/manifest.mpd</Location>
```

```
<ContentSteering defaultServiceLocation="alpha">https://steeringservice.com/app</ContentSteering>
```

```
<Period .../>
```

```
</MPD
```

**Steering server  
would return**

```
{  
  "VERSION": 1,  
  "TTL": 300,  
  "RELOAD-URI": "https://steeringservice.com/app/instance12345?session=abc"  
  "SERVICE-LOCATION-PRIORITY": ["mpd2", "segments2"]  
}
```

# Can we clone pathways?

Yes – by adding a "PATHWAY-CLONES" array to the steering server response

```
<MPD ... >
```

```
<BaseURL serviceLocation="alpha">http://cdn1.com/</BaseURL>
```

```
<BaseURL serviceLocation="beta">http://cdn2.com/</BaseURL>
```

```
<ContentSteering defaultServiceLocation="alpha">https://steeringservice.com/app</ContentSteering>
```

```
<Period .../>
```

```
</MPD
```

```
{ "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": "cdn3.com", "PARAMS": { "token": "dkfs1239414" } } } ]
}
```

Steering server returns

# Useful things to know about DASH Content Steering

1. It is designed to be **compatible with HLS Content Steering** and leverages the same steering server response.
2. If the player encounters a delivery error, it's first job is to **try and maintain the best QoE** for the end-user.
3. DASH reports the active serviceLocation and throughput using the reserved **\_DASH\_pathway** and **DASH\_throughput** parameters. Unlike HLS, these allow the **reporting of multiple values**.
4. DASH has the ability to force a player to query the steering server **before starting playback**. This allows steering to be used to update old manifests in the field that point to content sources that no longer exist.

# Interesting idea – default manifests

What if we made a base template that did not define a delivery host and then we use the steering server to allocate the CDN at request time?

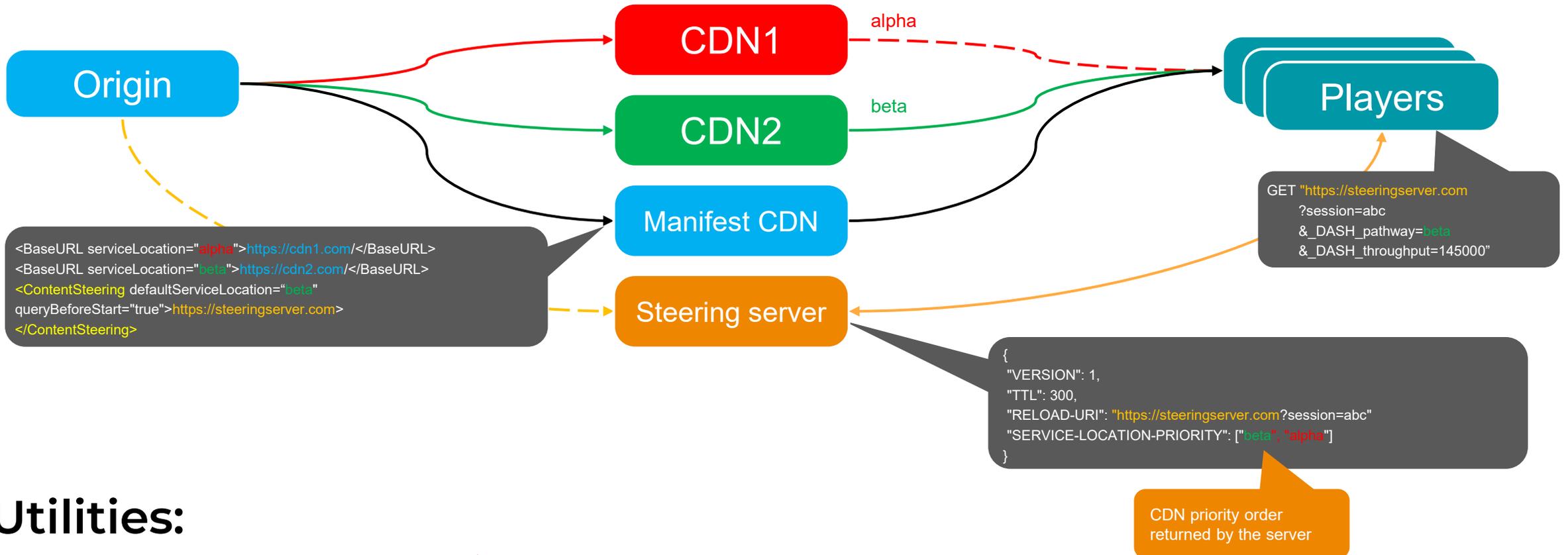
```
{
  "VERSION": 1,
  "TTL": 300,
  "RELOAD-URI": "https://cdn.dashjs.org/test/steering-server-response-6.json",
  "PATHWAY-PRIORITY": ["primary"],
  "PATHWAY-CLONES": [
    {
      "BASE-ID": "default",
      "ID": "primary",
      "URI-REPLACEMENT": {
        "HOST": "https://dash.akamaized.net"
      }
    }
  ]
}
```

# Implementing DASH Content Steering at Scale



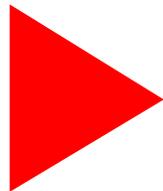
# EXAMPLE SYSTEM

A system with 2 CDNs:



## Utilities:

- ▶ Failover protection
- ▶ Load-balancing
- ▶ QOS/QOE optimizations
- ▶ COGS optimizations



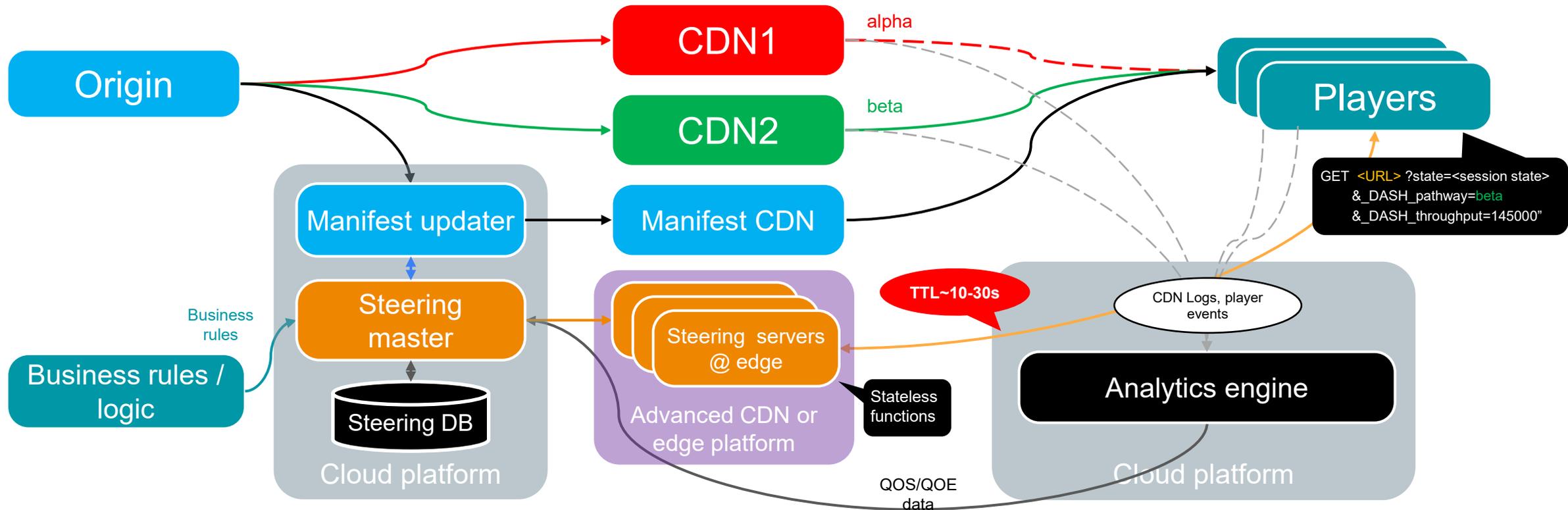
Requires information about

- ▶ QOS/QOE metrics as delivered by each CDN
- ▶ Business rules / pricing information



# CONTENT STEERING @ EDGE

## Edge-based implementation

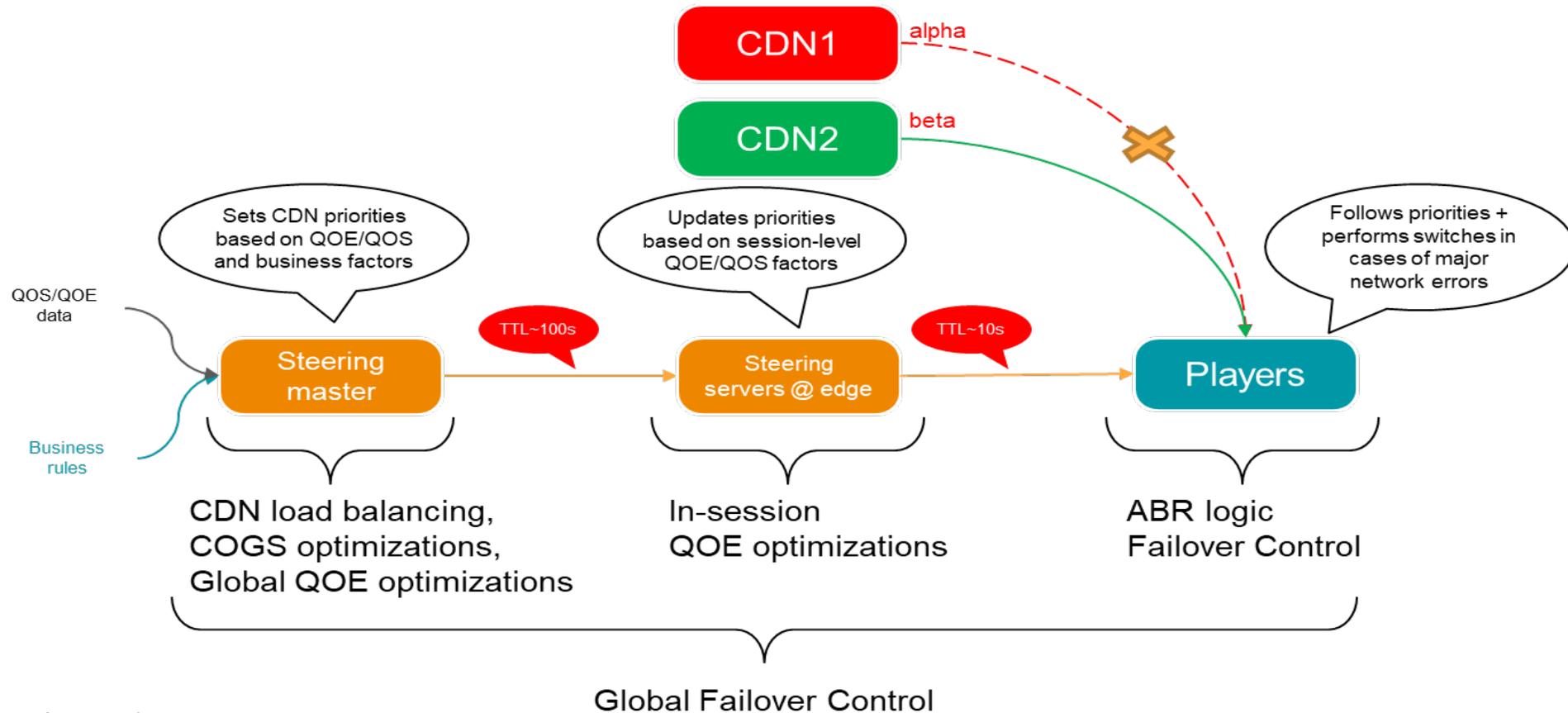


## Advantages:

- ▶ TTL can be much smaller – comparable to player buffer delay
- ▶ Enables QOE optimizations, faster switching / failover, more precise load-balancing.
- ▶ Scales well. Multiple CDNs or platforms can be used for redundancy.

# DISTRIBUTION OF FUNCTIONS

Between clients, servers, edge



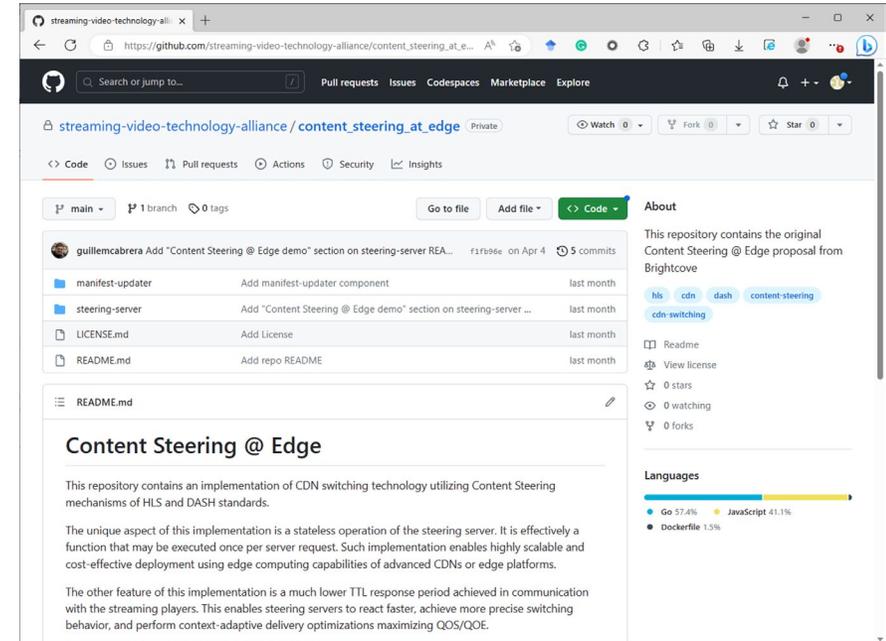
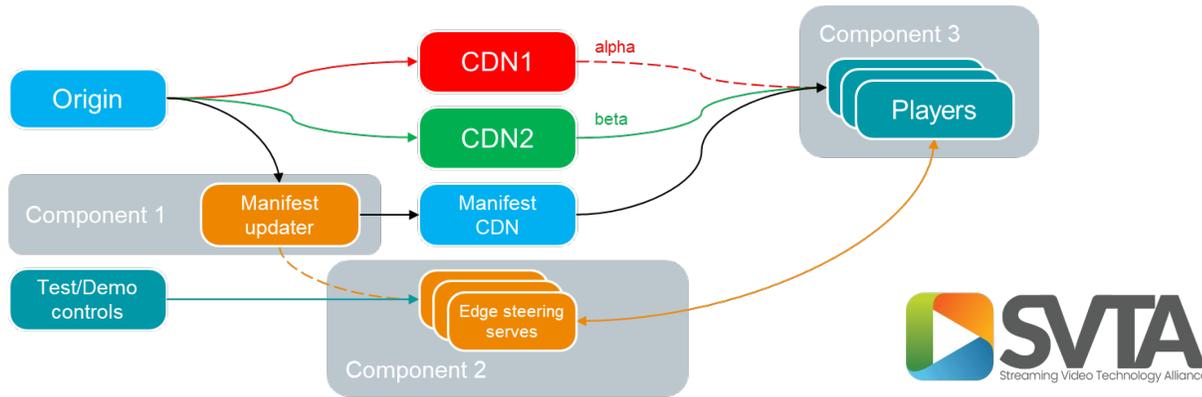
## Main principles

- ▶ Move instantaneous / local decisions to the client and the edge; use local, player provided information
- ▶ Make global decisions at the master server and using all information available about the system

# OPEN SOURCE PROJECT

## SVTA Labs project

### ► POC system:

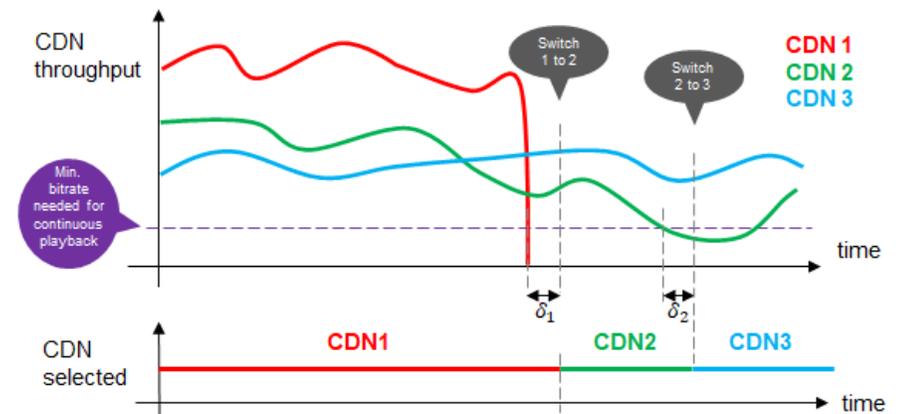


### ► Open source components:

1. Manifest updaters (Go, standalone server)
2. Steering servers (JS, Lambda @ Edge)
3. Players (DASH.js, HLS.js, video.js)

### ► Supported functions:

- Failover
- In-session QOS/QOE optimizations
- Centrally forced state updates



# TECHNOLOGY DEMONSTRATION

## Content steering @ edge

- ▶ SVTA/Brightcove steering servers
- ▶ DASH.js player
  
- ▶ Functions:
  - > QOE optimizations
  - > CDN load-balancing
  - > Failover protection
  - > Manual controls
  
- ▶ Advanced applications
  - > Delivery COGS optimizations
  - > Multi-region delivery optimizations

The screenshot displays the Brightcove Content Steering interface in a browser window. The main content area shows a video player with a white rabbit-like character in a field. To the right, there is a 'CDN Selection' diagram showing a computer icon connected to three server icons labeled 'CDN A', 'CDN B', and 'CDN C'. Below the video player, the 'Steering Data' section shows request details including a timestamp, steering URL, pathway (cdn-c), and throughput (1272000 bps). The 'Response' section shows version 1, TTL 10, and pathway priority (cdn-c,cdn-a,cdn-b). On the right side, there are 'Fragment Requests' and 'Steering manual controls' sections. The 'Fragment Requests' table lists audio and video requests. The 'Steering manual controls' section includes a 'CDN Order' dropdown and 'CDN Throughputs' input fields for each CDN, with an 'Update' button.

# TAKEAWAYS



# CONTENT STEERING: TAKEAWAYS

## The Standard

- Done and ready for deployment

## Many Benefits

- multi-CDN solutions for the masses – reliable, simple, no mess
- load balancing, COGS optimizations – can be done by a simple server
- QOS/QoE optimizations, fast failover, etc. – doable using Edge platforms

## Open source players, servers, POCs:

- DASH.js, HLS.js, video.js (upcoming)
- ExoPlayer (in development) <https://tinyurl.com/yck2rw2r>
- Shaka packager (POC) <https://tinyurl.com/2e53pm2a>
- SVTA content steering at edge <https://tinyurl.com/3scm5pdn>