AD INSERTION IN MPEG DASH

Alex Giladi

April 2015

invention | collaboration | contribution

Basics

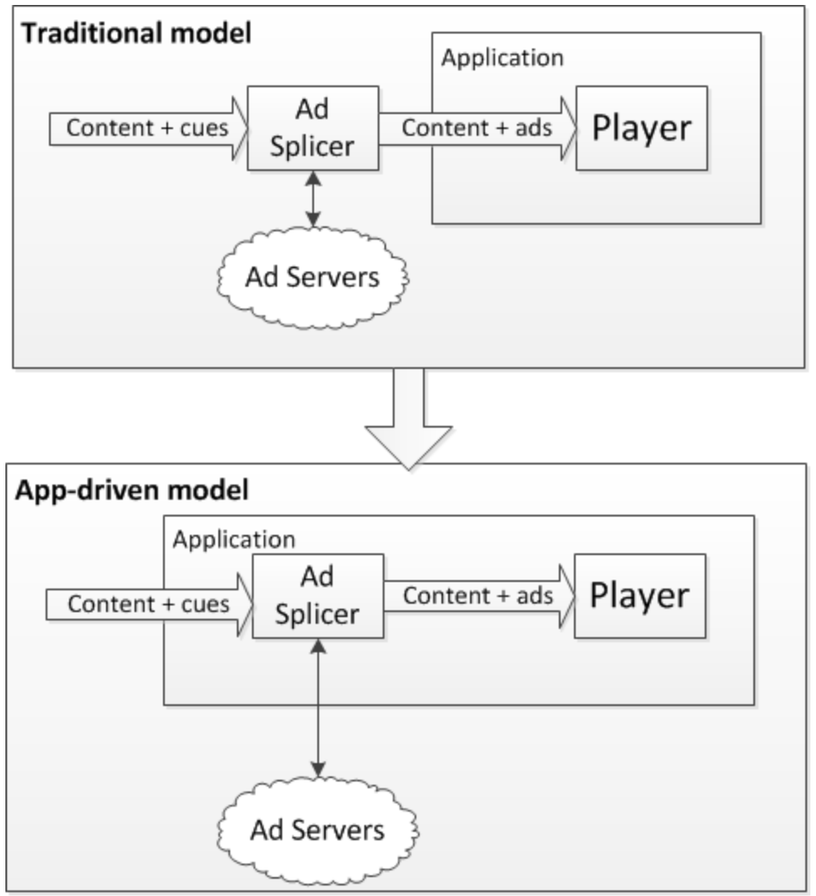
- Ads are not inserted at random
  - Possible placement opportunities (a.k.a. avails) are known
    - SCTE 104 / SCTE 35: marks (cues) carried inband
    - VMAP: out of band (separate document)
    - App can start/pause/resume a client
- Live case
  - Ad break location known a few seconds ahead of time
    - Inband SCTE 35 marks location and contains avail duration
  - Ad break duration is also known at that point
- VoD case
  - Ad break location known from the start
  - Ad break duration unknown

Two architectures

- App-driven architecture
  - App controls a DASH client
    - Interacts with ad servers
    - DASH client passes cue messages to app
    - App can start/pause/resume a client
  - DASH used as a transport mechanism
  - Similar to HDS practices
- Server-driven architecture
  - Native DASH implementation
    - Generic client is sufficient
  - Communication with ad servers hidden from the client
  - Similar to HLS practices

App-driven model: translating existing workflow into DASH

- "Ad Splicer" is a module in the client app
  - Cue messages passed to the client
  - Client-side module acts on cue message
    - Direct communication between client and ad servers
- DASH client is a module in the client app
  - Cues embedded in DASH content
  - DASH client conveys cues to a "splicer" module
- DASH events used to transport cue messages
  - For a DASH client user-defined events are "timed blobs"
    - Can be embedded within segments
    - Can be embedded in MPD (at Period level)
    - DASH client is not expected to parse user-defined event payloads
    - Application can register a callback for certain event type(s)
  - Cue events are essential
    - DASH client w/o support for a specific cue format cannot play the content



Carrying SCTE 35 cue messages

MPD Event

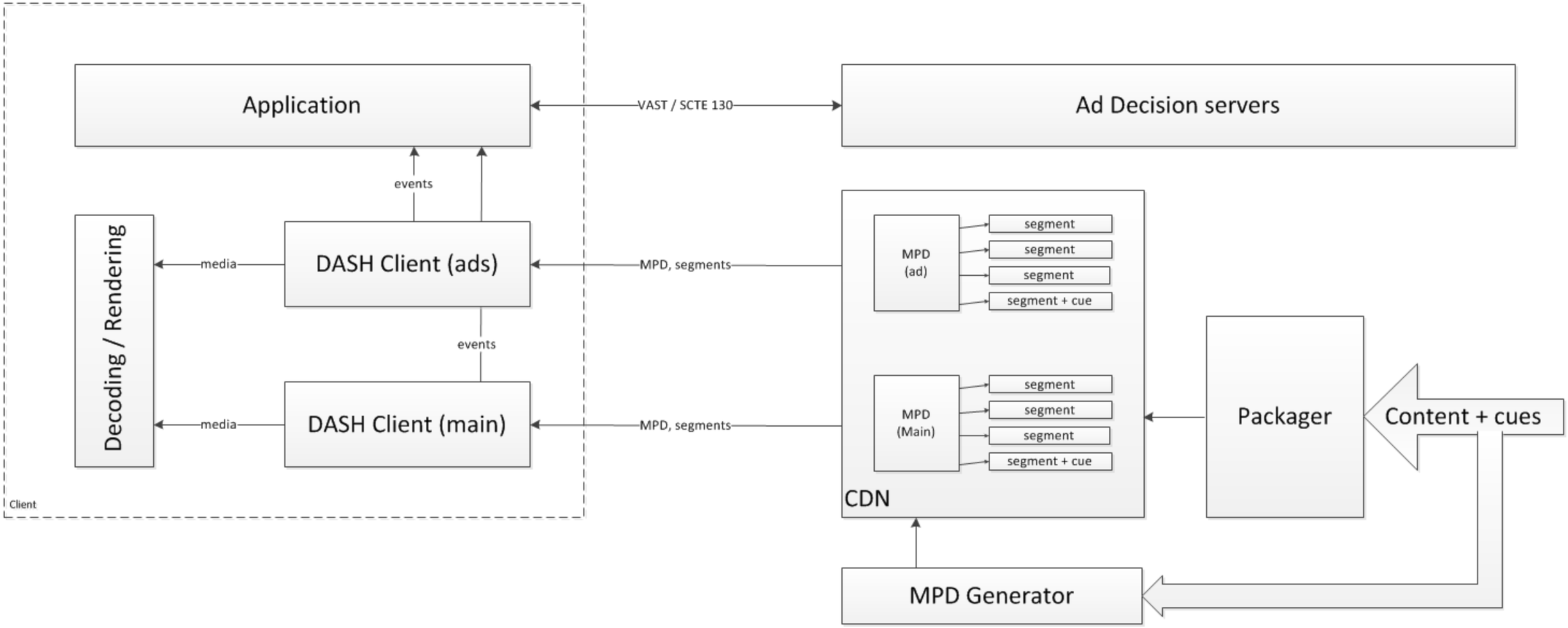
```
<Period>
…
  <EventStream schemeIdUri="urn:scte:scte35:2013:xml">
    <Event timescale="90000"
          presentationTime="54054000"
          duration="5400000" id="1">
          <scte35:SpliceInfoSection scte35:ptsAdjustment="0"
              scte35:tier="22">
              <scte35:SpliceInsert
                   scte35:spliceEventId="111"
                   scte35:spliceEventCancelIndicator="false"
                   scte35:outOfNetworkIndicator="true"
                   scte35:uniqueProgramId="65535"
                   scte35:availNum="1"
                   scte35:availsExpected="2"
                   scte35:spliceImmediateFlag="false">
                   <scte35:Program>
                       <scte35:SpliceTime
                                scte35:ptsTime="122342"/>
                   </scte35:Program>
                   <scte35:BreakDuration
                           scte35:autoReturn="false"
                           scte35:duration="5400000"/>
              </scte35:SpliceInsert>
              <scte35:AvailDescriptor
                  scte35:providerAvailId="332"/>
          </scte35:SpliceInfoSection>
    </Event>
  </EventStream>
…
</Period>
```

Inband Event (`emsg`)

scheme_id_uri=”urn:scte:scte35:2013:bin”
value=1001
timescale=90000
presentation_time_delta=540000
duration=5400000
id=0
0xFC 0x30 0x08 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x10 0x00 0x06 0x7F 0x23 0x45 0x67 0x89 0x00 0x10 0x02 0x00 0x43 0x55 0x45 0x49 0x40 0x00 0x00 0x00 0x7F 0x9C 0x00 0x00 0x00 0x00

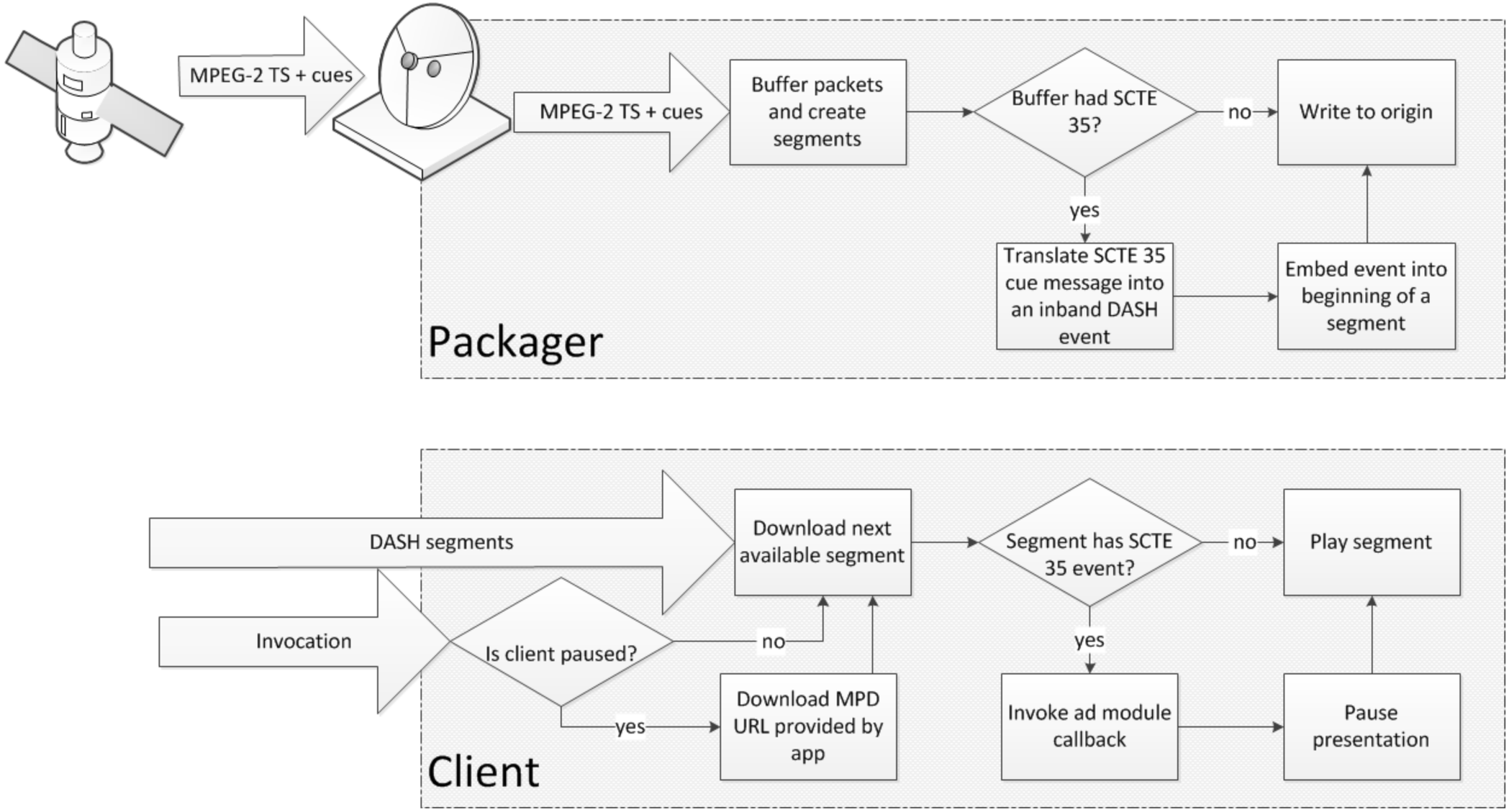
Binary SCTE 35 cue message

Putting it all together: App-driven architecture



INTERDIGITAL

Linear workflow, app-driven case

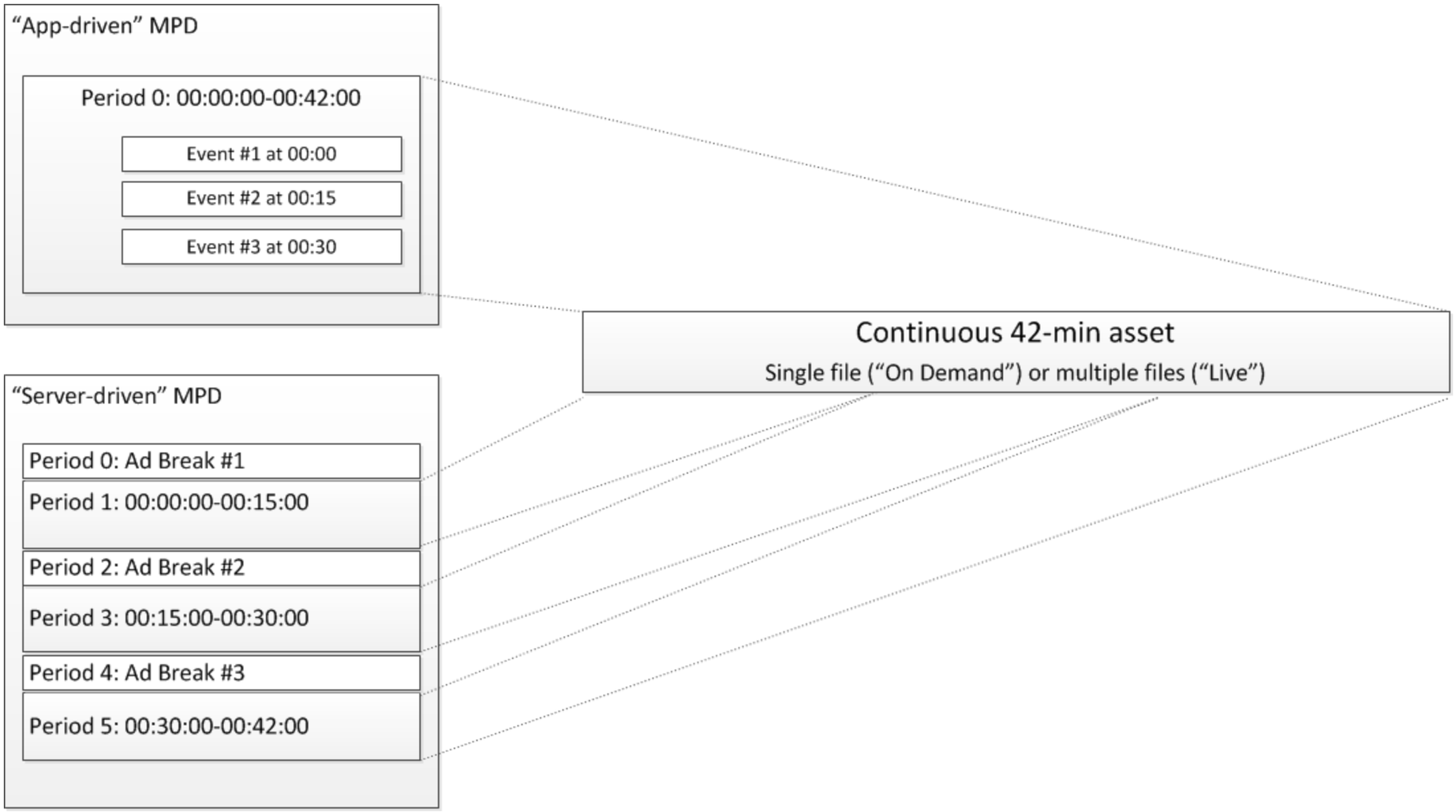


INTERDIGITAL

Server-driven model: single-client native ad insertion

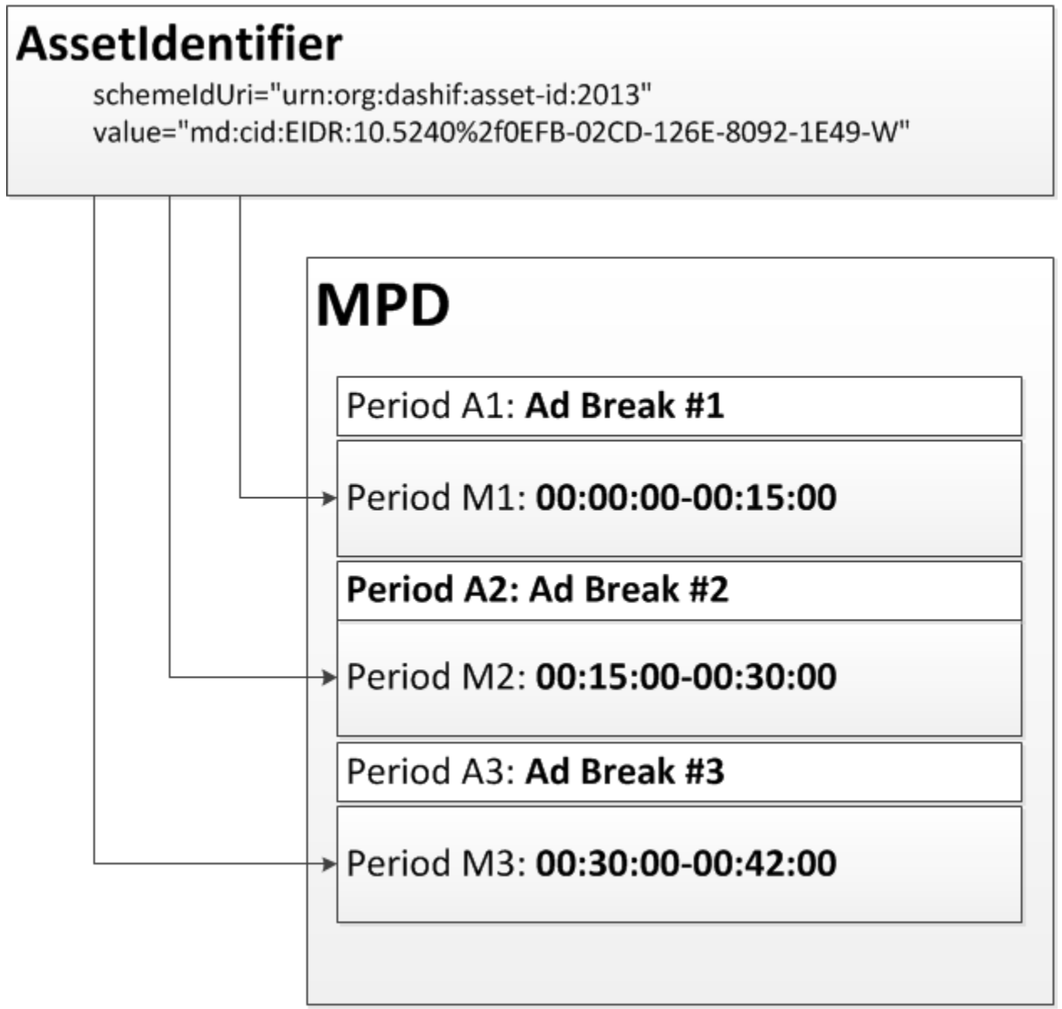
- **App-driven**
  - Non-DASH logic and modules needed
  - Multiple DASH clients are needed
- **Server-driven**
  - Use built-in DASH tools
    - Ad is a period
    - Remote period is a cue
    - MPD updates and XLink for just-in-time functionality
  - Client plays out the MPD it is given
    - Possibly requests ads from an ad proxy using XLink.
  - Ad decision logic solely on server side
    - Ad proxy communicates with ad decision servers on behalf of the client
  - Additional non-DASH logic is possible
    - This logic may not affect client functionality – e.g. tracking

Expressing cues: periods vs user-defined events



Multi-period MPD: identifying assets

- Assets split into multiple periods
  - Periods from the same asset may be separated by inserted content
  - Keeping track of asset time is often needed
    - Progress bar, bookmarks, trick modes, etc.
- Period continuity needs signaling
  - Identical asset identifiers across periods
  - M1, M2, and M3 have same id, thus are continuous
  - Lack of identifiers – no expectation of continuity
- AssetIdentifier is extensible
  - DASH-IF interop point uses MovieLabs URNs
  - SCTE created a multi-identifier scheme
    - Based on SCTE 35 UPID mechanism
  - It is possible to explicitly indicate “main”

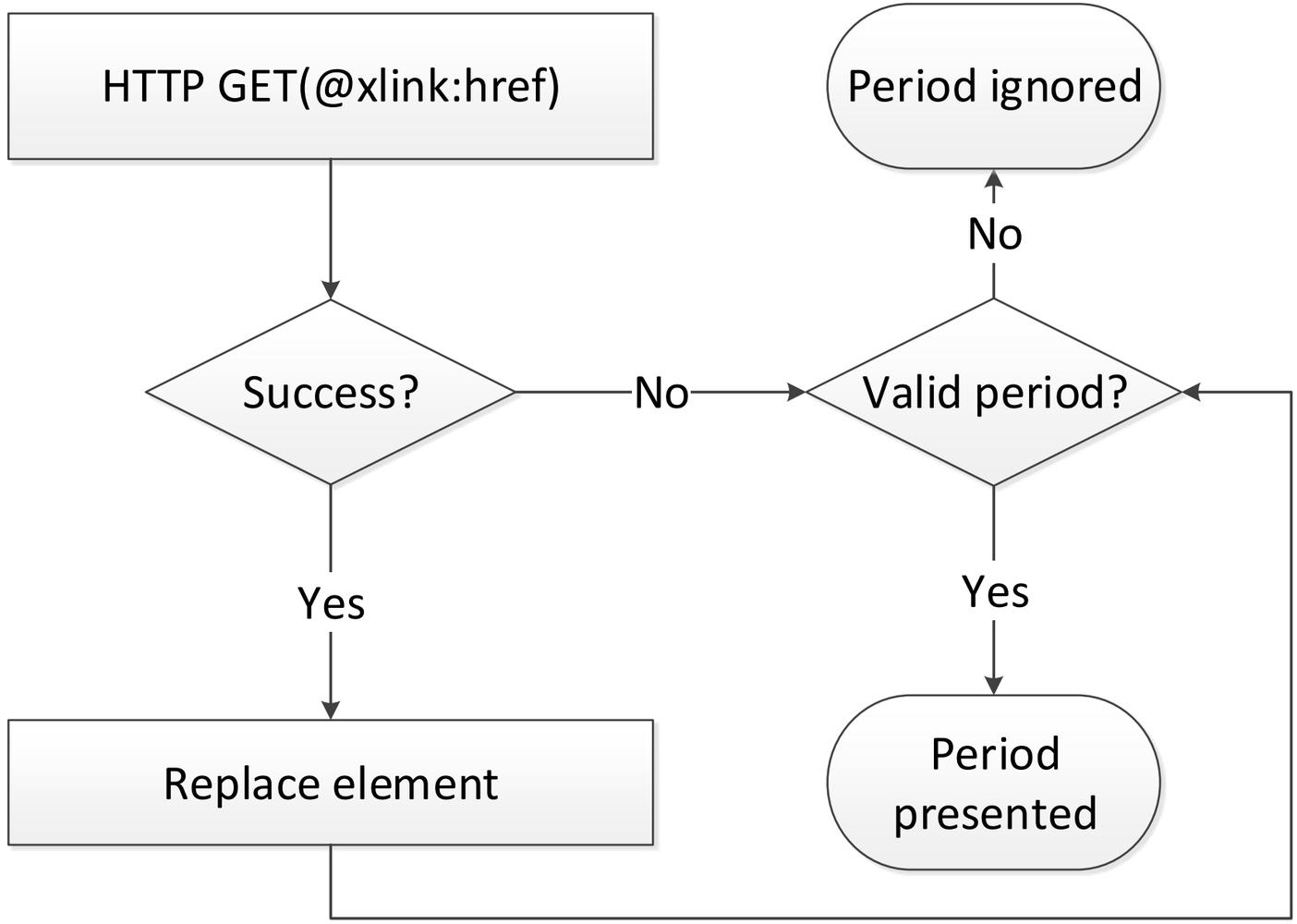


Enter dynamicity

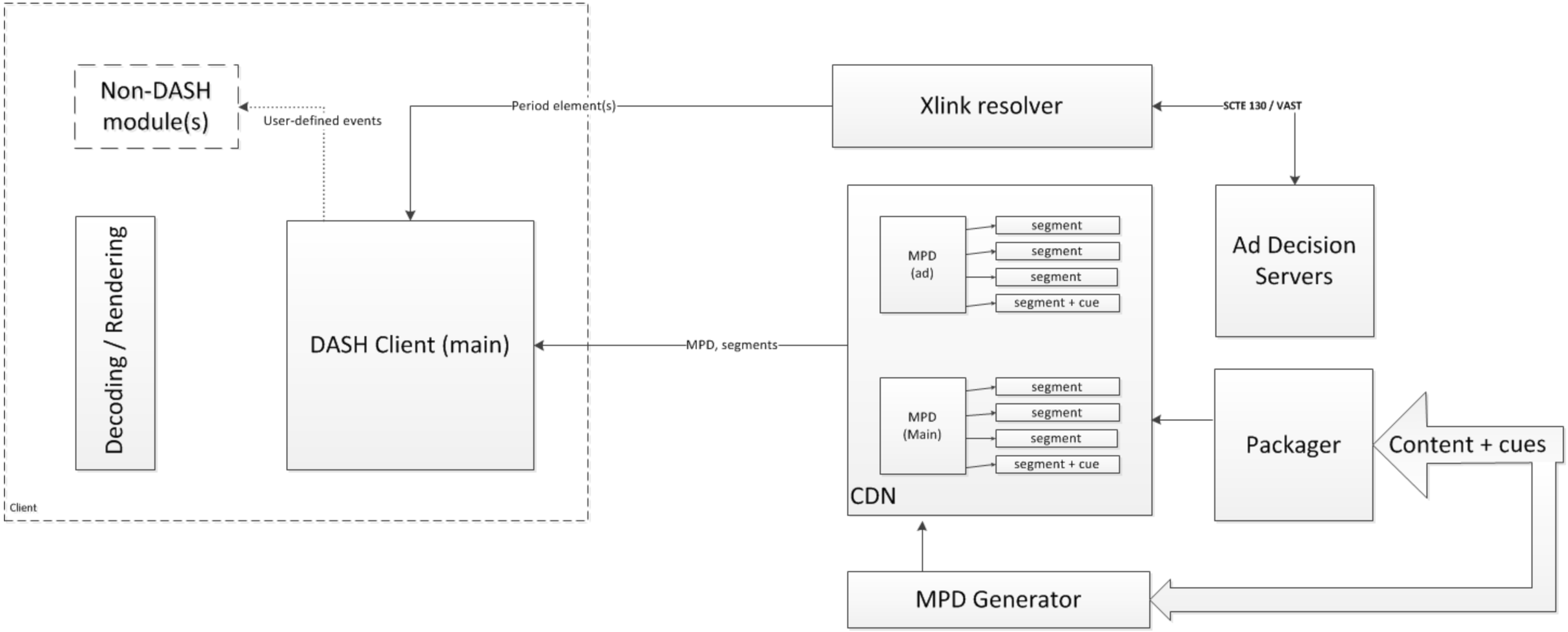
- Periods and asset identifiers provide static ad insertion functionality
  - All ad decisions are made at MPD generation time
    - Trade-off between scalability and targeting
- Insert new ad periods close to their playback time
  - MPD can be updated in real time
    - Regular updates
      - *inefficient for ad insertion – cue message is a rare near-realtime event*
    - DASH MPD Validity Expiration events
  - New ad periods appear in the returned MPDs
- Remote Period elements
  - Remote Period is a period containing **`Period`**`@xlink:href` attribute
  - One or more Period elements can be retrieved from XLink resolver using XLink URL
  - Remote element used as a cue
    - Cue message can be passed in real time to the XLink resolver (making it an ad proxy)
    - XLink resolver conceptually equivalent to the “ad splicer” module in app-driven model

XLink

- Just in time resolution
  - Single period can be resolved into several periods
  - XLink URL can pass cue parameters to resolver
- Default “slate” content
  - Period can have valid “slate” content, replaced in real time
  - Can be used for ad replacement



Putting it all together: Server-driven architecture



INTERDIGITAL

Tracking ad impressions

- Naïve implementation
  - Collect CDN logs as a proof of downloads
  - Assumes something was seen because it was downloaded
- IAB VAST
  - IAB VAST standard has extensive support for reporting impressions
    - Time-based (start, 25%, 50%, 75%, 100%)
    - User action (click, pause, mute, etc.)
  - Embedded in an MPD event
- DASH Callback event
  - DASH client issues an HTTP GET to a URL provided in the event
    - HTTP GET issued at event presentation time,
    - HTTP Response body is discarded w/o parsing;
  - Simple native ad reporting compatible with time-based VAST tracking
    - No support for user action
  - Added in AMD3

Improving targeting with custom parameters

- Generic URL parameters
  - Name-value pairs that can be added as query or fragment parameters to segment URIs
  - Parameters can be instantiated at different times
    - MPD generation / fetch
    - XLink resolution
    - Client-side computation
  - Powerful mechanism for reporting user identity and state using segment requests
  - Added in AMD2
- Generic HTTP header parameters
  - Mechanism analogous to URL parameters, applied to HTTP headers
  - Parameters can be added to regular (non-template) HTTP GET requests
    - Callback events
    - XLink resolution
  - Powerful mechanism to provide state for ad decision and ad tracking
  - Technology under consideration

Restricting client behavior

- Annotate limits on trick modes
  - Use of SCTE 130-10 – Source restriction data model
  - Indicates range of playback speeds within a time range
    - Similar to RTSP
    - Example: “playback speed cannot exceed x1.0 between 0:00:00.00 and 0:42:42.42”
  - Used as period-level annotation
    - Added in SCTE DVS 1202

Summary

- DASH has enough native tools to implement interoperable ad insertion
  - "Batteries included"
  - Both HLS-style and HDS/Smooth-style app-driven functionality can be supported
  - Easy to integrate with existing back-end systems
- Specification status
  - DASH-IF
    - Extensions (events, XLink) added to DASH-IF IOP 3.0
    - Recommendation document finalized
  - SCTE
    - SCTE DASH specifications finished, to be balloted in May 2015
    - Carriage of SCTE 35 normatively defined in DVS 1202 and DVS 1208
      - *will become ANSI/SCTE standards in 2015*
- New work in MPEG
  - New events and parameterization

THANK YOU!