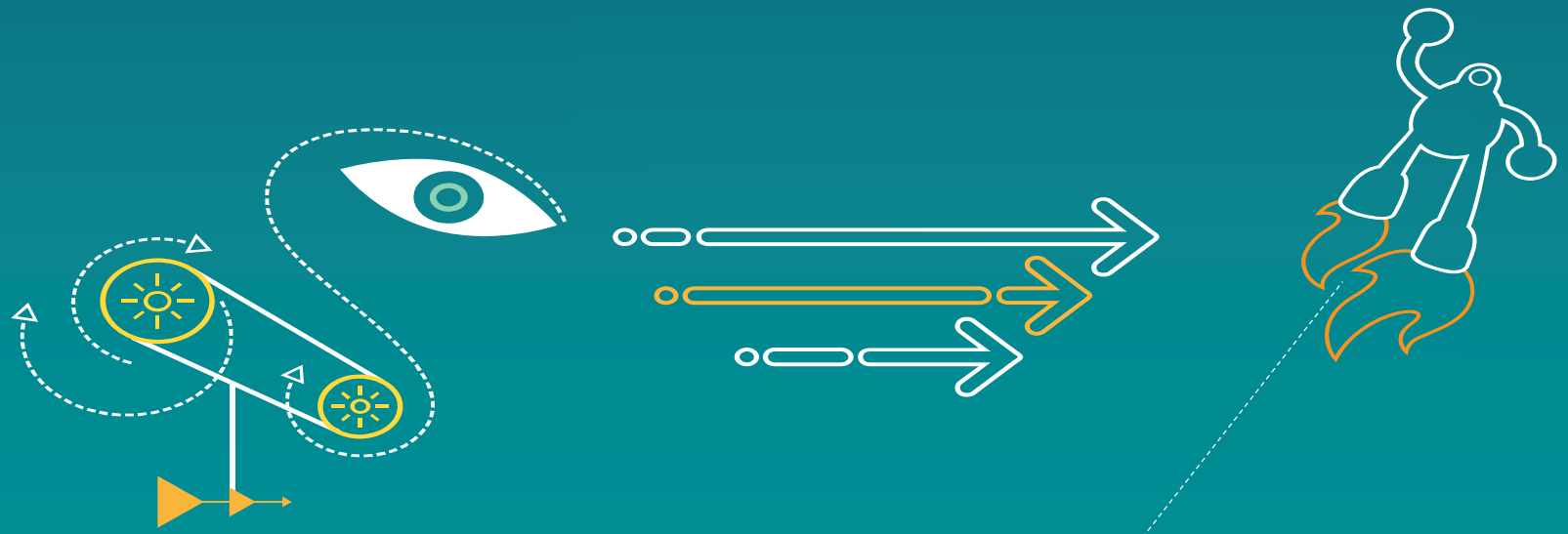
August-2015

W3C, ATSC and DASH Streaming Media

QUALCOMM®



Overview

- Moving to IP-based streaming will benefit broadcast services
  - Leverage existing IP-based ecosystem
    - Not just basic equipment (e.g. routers), but developers as well (particularly web)
  - Opens the door to hybrid content delivery
    - Use broadcast for wide area delivery of base forms of content (audio/video/captions)
    - Leverage broadband for specialized content delivery
      - Personalized content (e.g. additional language tracks, accessibility)
- DASH has provided a scalable solution for broadband media delivery
  - So what has to happen to make this work for broadcast-only and hybrid?
- W3C and ATSC are trying to address these topics, but DASH community needs to provide more guidance
  - Particularly in the areas of personalization (e.g. in context of ad insertion)

W3C Technology Development has Complimented DASH

- Key enablers introduced in HTML5 timeframe
  - HTMLMediaElement, including <video> tag
    - Allows for playback of video (streaming, file-based, etc.) without requirement for plug-ins
  - Media Source Extensions (MSE)
    - Allows for adaptation of <video> source buffer
      - Necessary for DASH-style rate adaptation
  - Encrypted Media Extensions (EME)
    - Incorporates common encryption (CENC) into <video>
- Dovetailed nicely with DASH.js development
- Some areas which could be explored further
  - Rate adaptation metrics in browser (e.g. alternatives to Navigation Timing)
  - Recording API's
  - Compatibility with redistribution mechanisms such as WebRTC

Broadcast Compatibility

- Sizeable interest exists in applying HTML5 technologies to rendering of streaming media over broadcast
- Technologies such as HbbTV 2.0 (and OIPF DAE) have started to move in this direction, but not to the extent that DASH.js is fully applicable as a reference design in these environments
- ATSC 3.0 has adopted a full IP transport and allows for compatibility with DASH.js-derived reference clients
  - Runtime environment can be based on off-the-shelf browser that can retrieve broadcast content from web proxy integrated in broadcast receiver
- There are some technical challenges in certain areas that require additional technical solutions on top of any javascript DASH player
  - DASH eventing
  - Channel/program change
  - Ad insertion

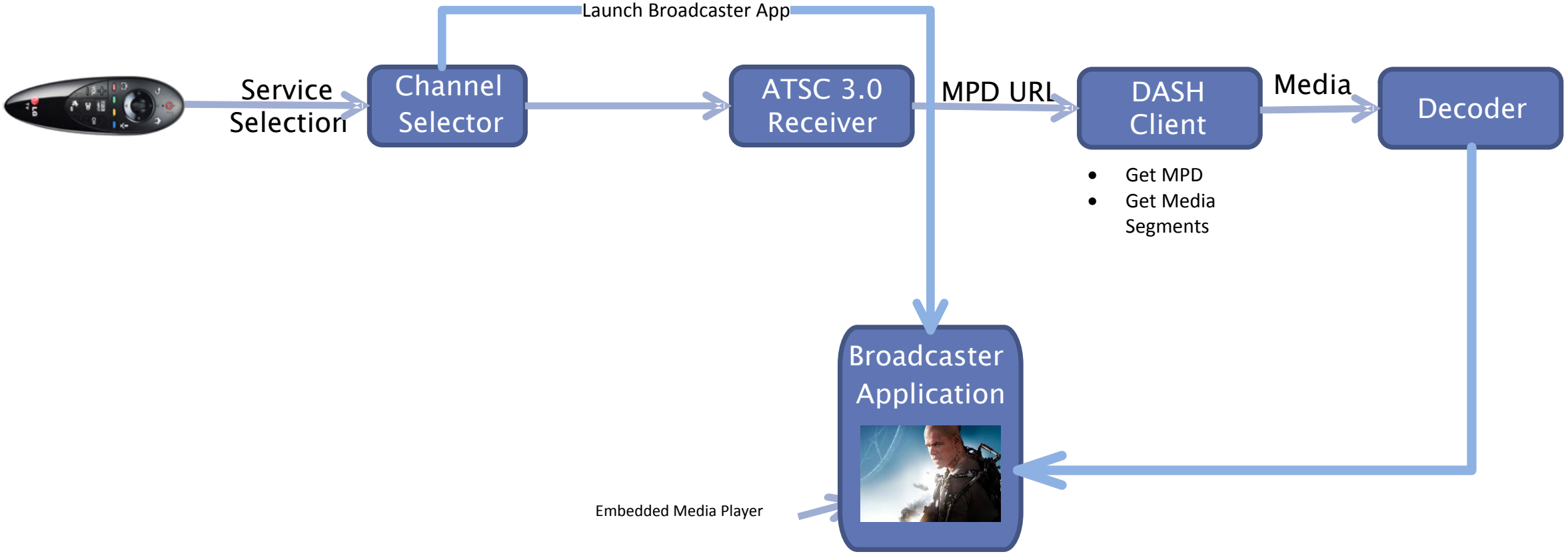
DASH Eventing

- DASH events are a means for indicating interactivity timed with the media
- DASH events may be conveyed via the MPD or through the ‘emsg’ box in the ISO BMFF media track
- Neither mechanism is well-suited for HTML5
  - MPD eventing requires the application (HTML/JS) to execute the events as per media playback time
    - Application may not have time-accurate insight into media playback status
    - Places dependency on MPD generator (usually resident with broadcaster)
  - emsg eventing support is non-existent in current browsers
    - Even if browsers propagated emsg data into JS, the event handler in the application can become a bottleneck
- There is also the possibility to embed DASH events into text tracks (e.g. TTML, WebVTT)
  - <video> implementation would have to propagate event to application layer for handling – a potential source of delay

Channel/Program Change

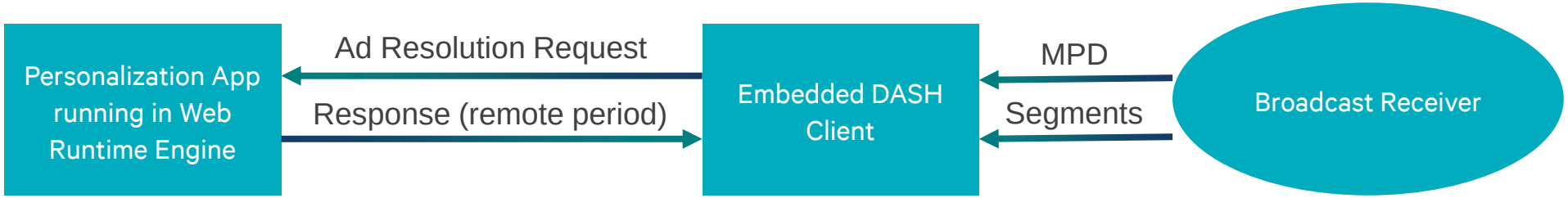
- Solutions such as HbbTV allow for association of a broadcaster-application to a streaming service
  - Upshot is that a specific application may be launched upon a channel change or a program change that would handle rendering of streaming media
- If the application renders streaming media using a library like DASH.js, will there be intolerably long interruptions in media playback?
  - Playback interruption affected by radio functions (frequency tuning, service acquisition), but could also be affected as a result of initialization of new <video> instance
- Therefore ATSC 3.0 is considering the use of 'always on' DASH clients that render media directly from the broadcast receiver
  - Use of DASH.js is possible, but would be abstracted from the service-bound application

Channel/Program Change (cont.)



Ad Insertion

- Broadcaster desire to drive client-side ad insertion
  - Example criteria for ad selection: geolocation, accessibility settings on broadcast receiver
- DASH-IF guidelines for ad insertion can work in this context
  - Use of Xlink resolution in MPD
    - Xlink in the MPD context is a hyperlink to a remote period description
- Use of embedded DASH clients (as depicted on previous slide) can present issues
  - Embedded DASH client has visibility into MPD and can alert broadcaster application about the need to resolve an Xlink
  - Will the embedded DASH client alert the app in a timely manner? Will the broadcaster app respond in time?



Summary

- Javascript DASH clients are well-suited for broadband content
  - DASH.js has gone a long way to establishing this
- W3C enablers for streaming media and reference Javascript Dash clients (DASH.js) can be used in broadcast context, but there are issues to consider
  - Is potential media disruption due to channel change significant?
    - Potential solutions (e.g. embedded DASH clients) present challenges as well (particularly ad selection)
- W3C has ongoing TV control API effort that could provide a browser-friendly way of handling broadcast streaming
  - Generalized to cover many different types of broadcast content delivery (MPEG2-TS, ATSC, etc.)
- DASH community should not just consider live streaming requirements going forward, but also the impact to web runtime engines
  - Personalization for broadcast content may pose challenges, but hybrid services brings new opportunities