



mhv/2025

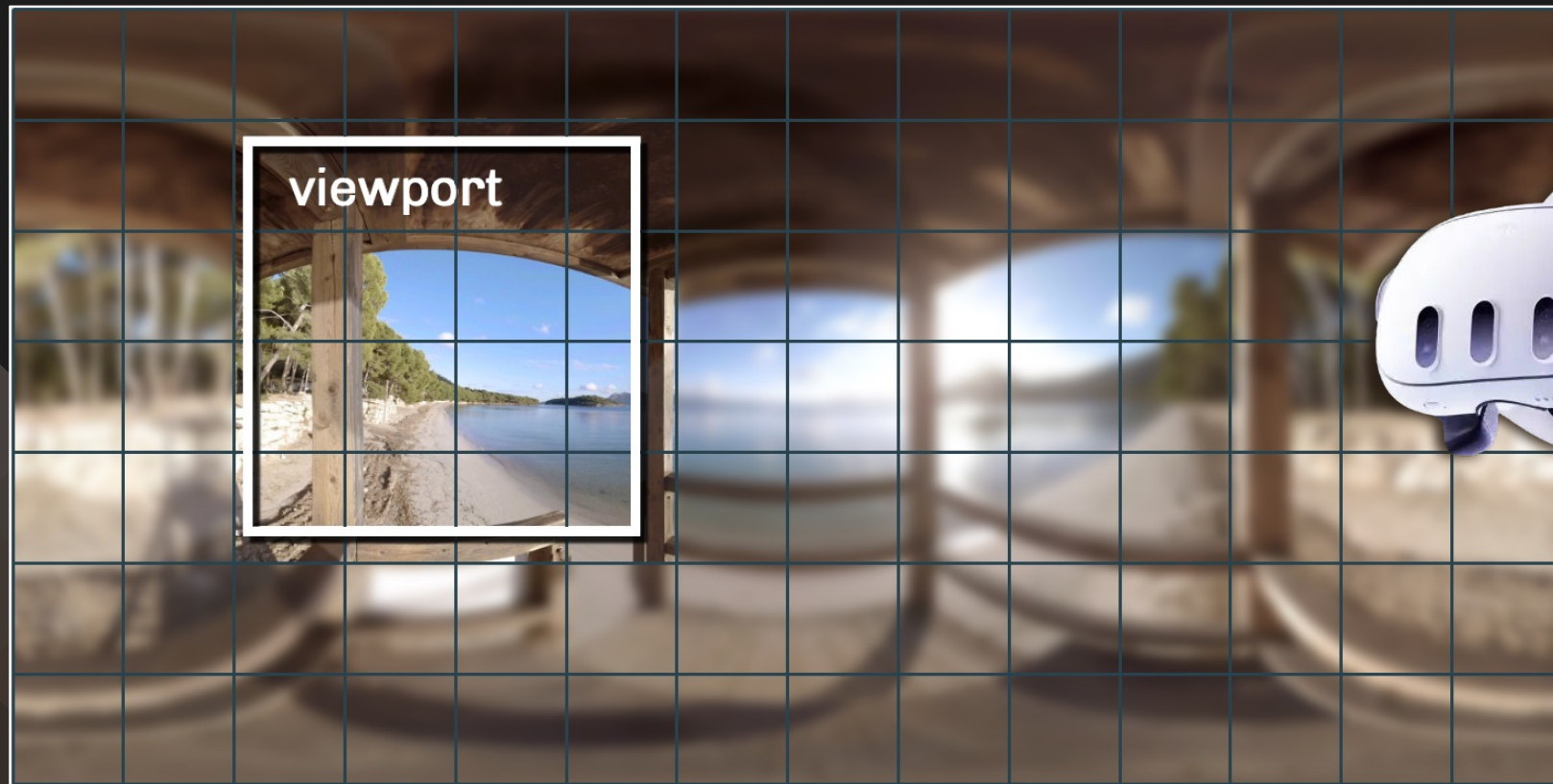
*Controlling the Entire Player
Stack for Better Performance
and Less Errors*

Rob Koenen, Ray van Brandenburg

Tiledmedia

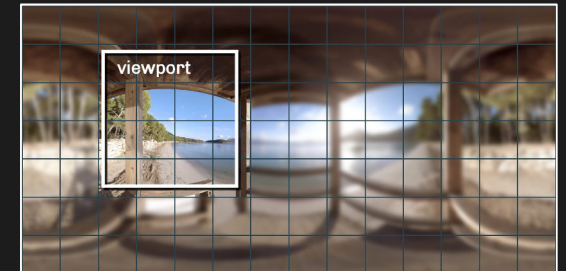


Necessity is the Mother of Invention



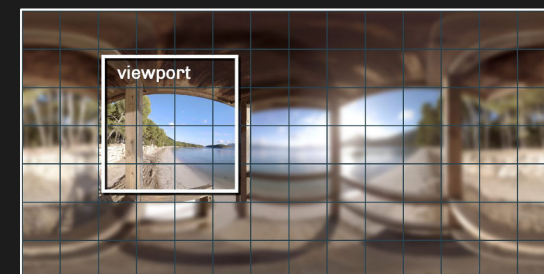
Necessity is the Mother of Invention

- Tiled streaming of 8K+ VR360 yields hard problems:
 - Coordinating retrieval & buffers across 25+ feeds
 - Frame-level sync across all tiled feeds, including multiple GOP versions and ABR levels
 - Retrieving, decoding and displaying new tiles within 1-3 frames in response to head motion



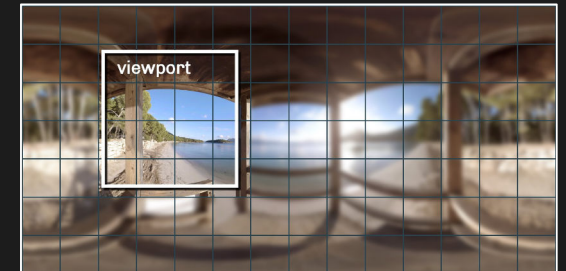
Necessity is the Mother of Invention

- Tiled streaming of 8K+ VR360 yields hard problems:
 - Coordinating retrieval & buffers across 25+ feeds
 - Frame-level sync across all tiled feeds, including multiple GOP versions and ABR levels
 - Retrieving, decoding and displaying new tiles within 1-3 frames in response to head motion
- Only option: build a player completely from scratch



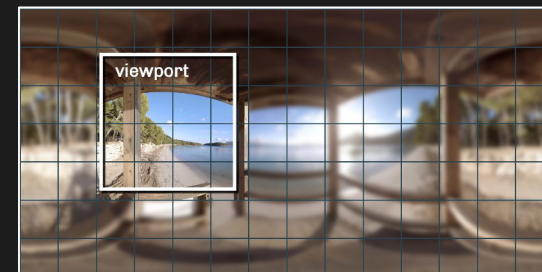
Necessity is the Mother of Invention

- After tackling VR, built Multiview in the same Multi-Stream Player
- Similar requirements : network retrieval efficiency, (massive) scalability, cross-feed ABR & buffer management, frame-level sync

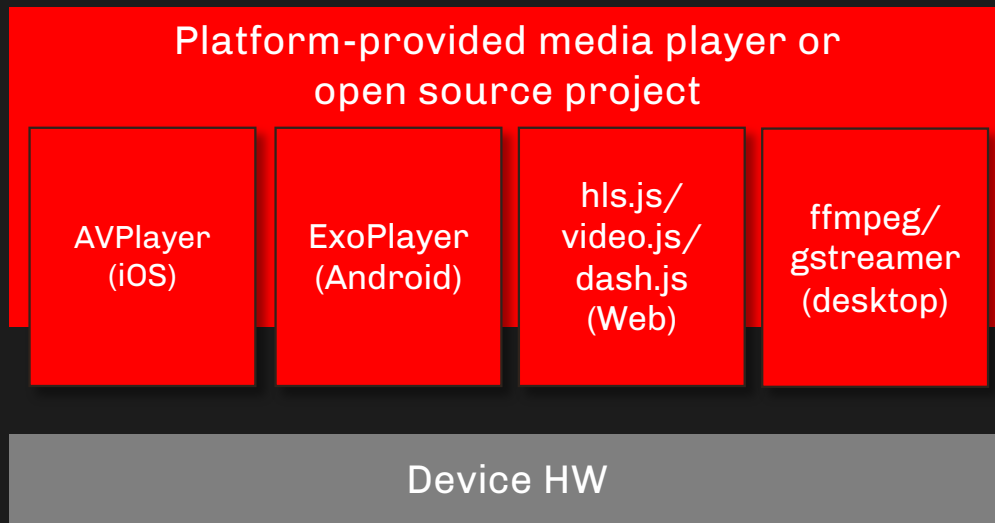


Necessity is the Mother of Invention

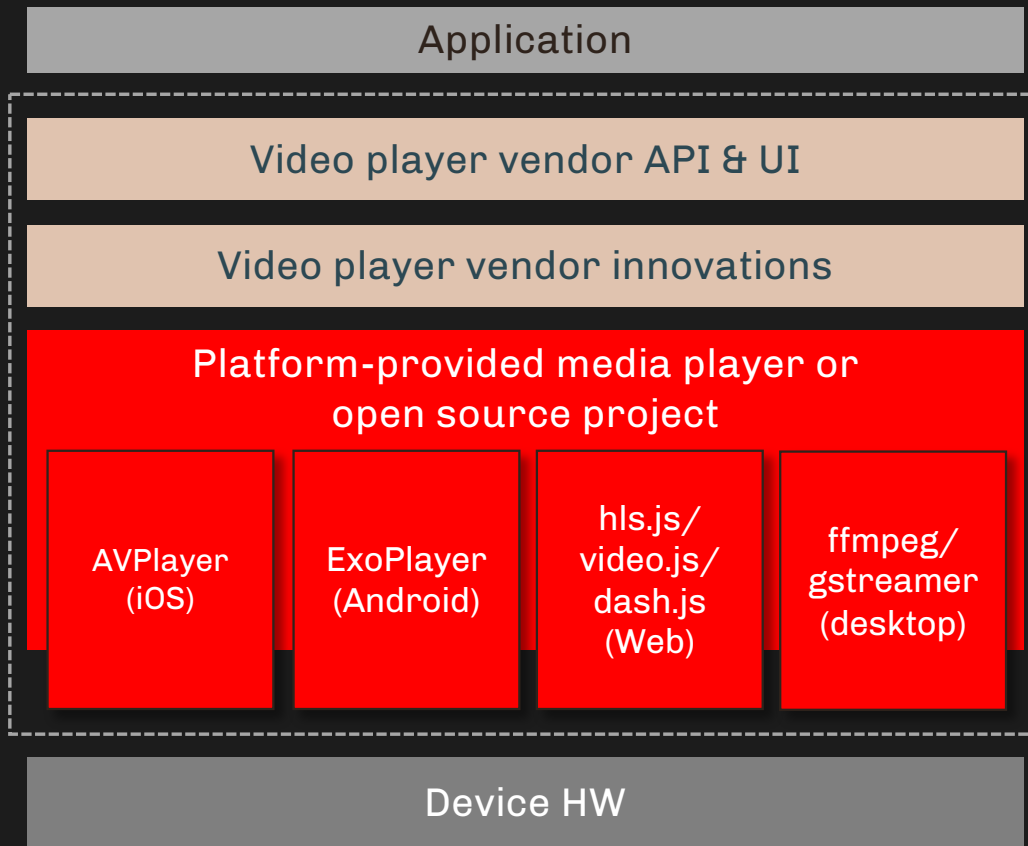
- After tackling VR, built Multiview in the same Multi-Stream Player
- Similar requirements : network retrieval efficiency, (massive) scalability, cross-feed ABR & buffer management, frame-level sync
- Working with launching multiview customer taught us that controlling the full stack also provides significant advantages for “conventional” streaming - sometimes to our own surprise.
- That’s the focus of today’s talk



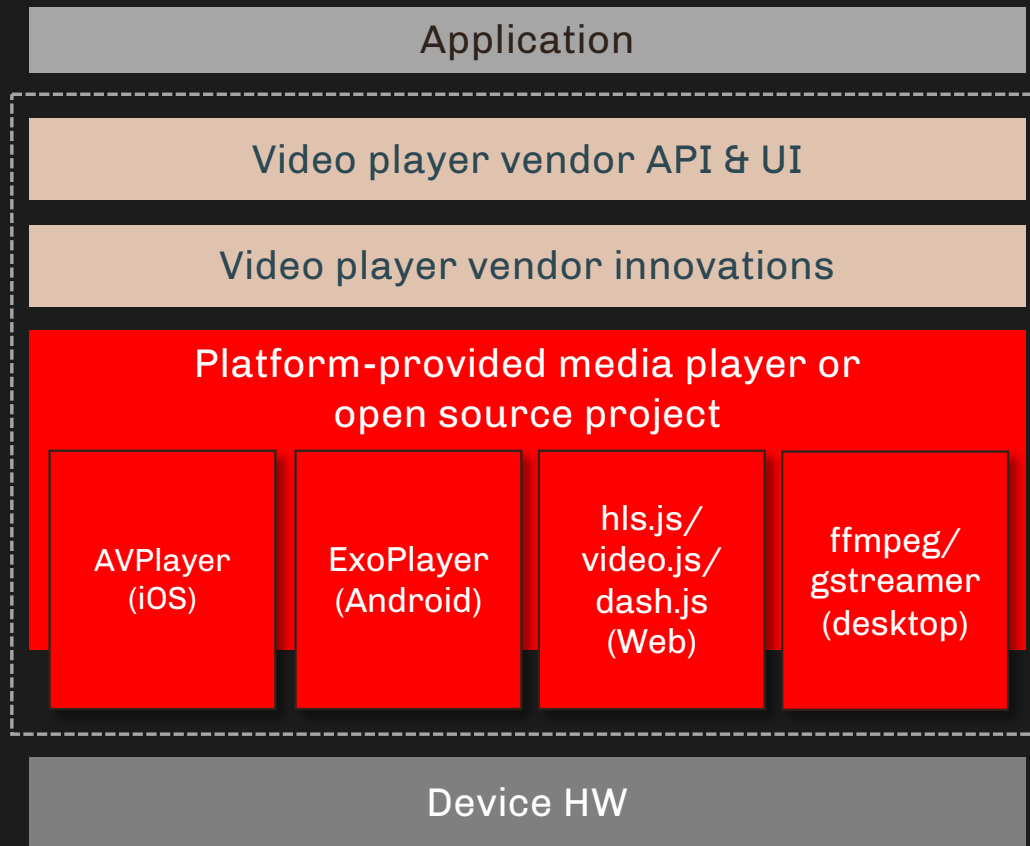
Traditional Approach



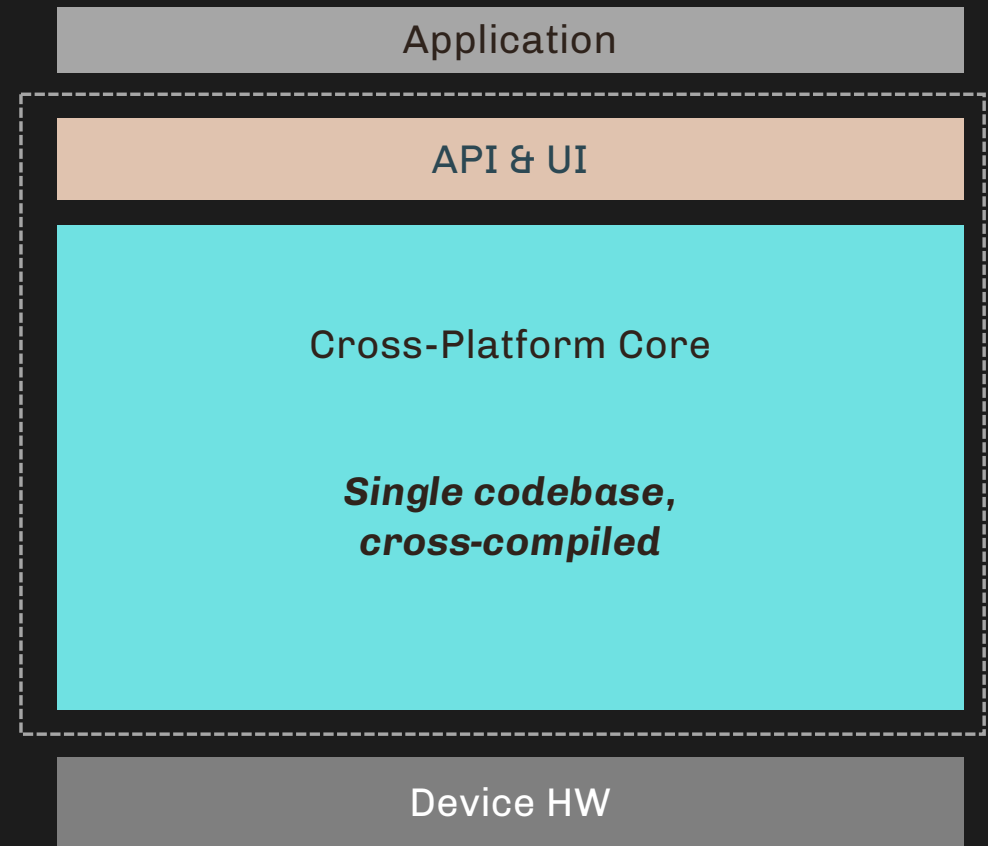
Traditional Approach



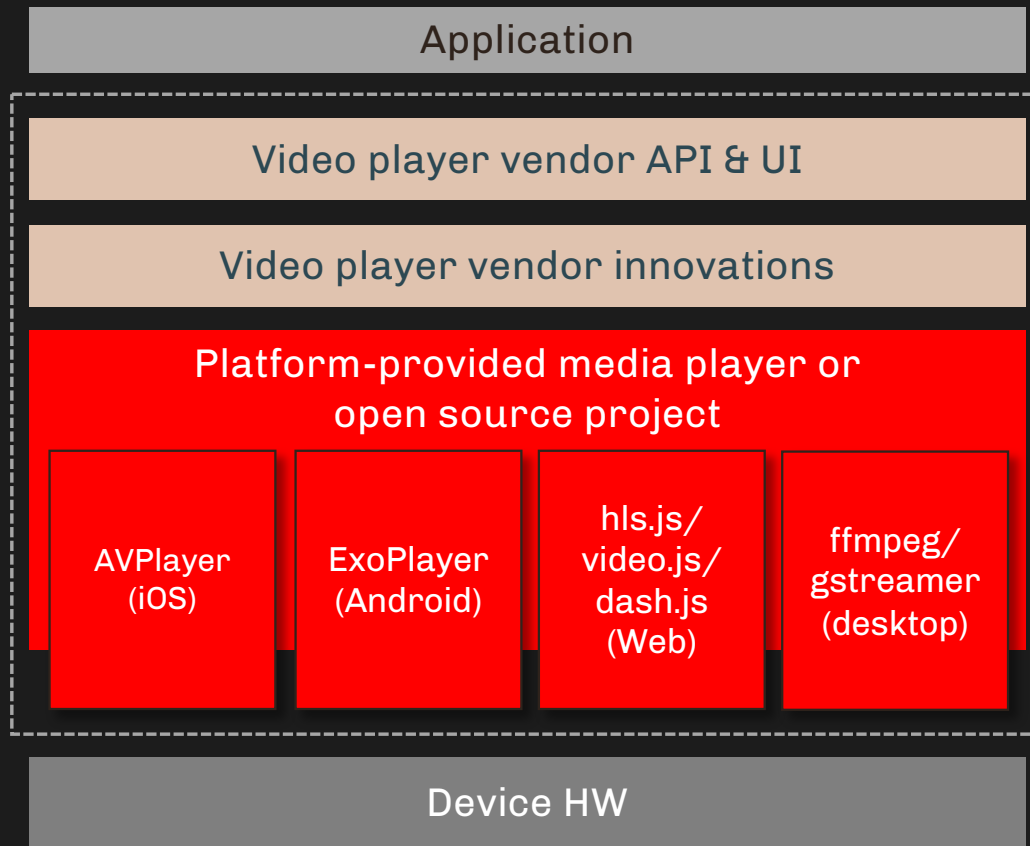
Traditional Approach



Build From Scratch

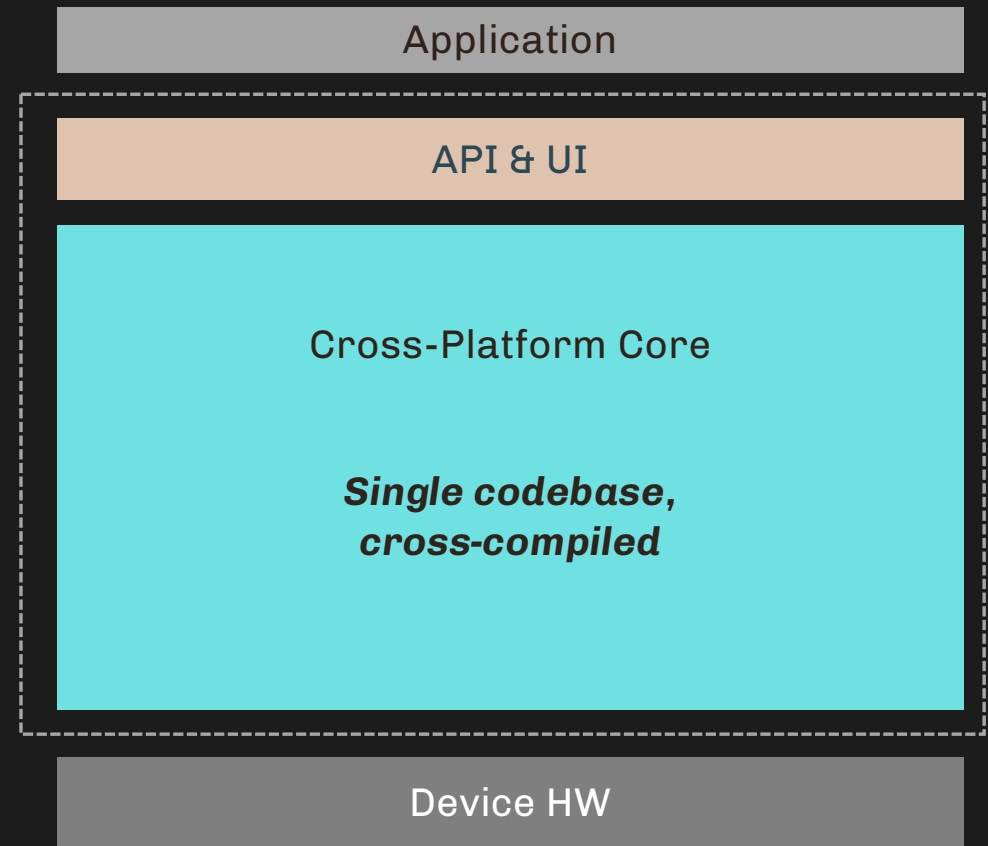


Traditional Approach



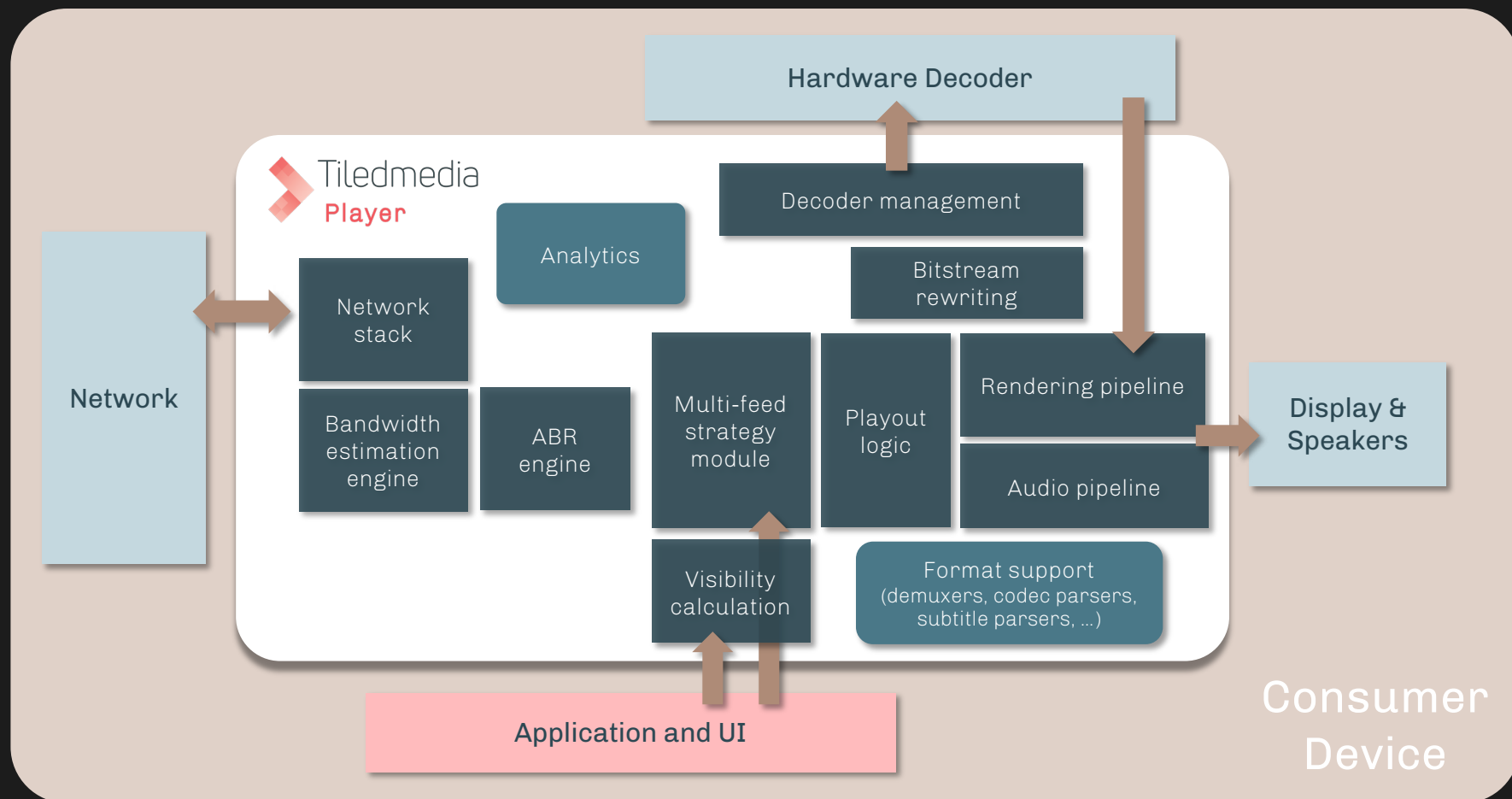
Limited Control and Platform-Specific

Build From Scratch



Full Control across all platforms

Player Architecture



Impact of Building a Player From Scratch

- Platform Support
- Format Support
- Error rates
- Analytics
- ABR
- Innovation and Integration
- Player Size

Format Support

- Single cross-platform player allows for massive reduction in number of packaging formats required to cover all platforms
- Typically, a single multi-DRM HLS ABR ladder suffices
- With a single subtitle/caption format
- And a single trickplay thumbnail format
- Any formats not yet supported only need to be implemented once

Format Support

- Single cross-platform player allows for massive reduction in number of packaging formats required to cover all platforms
- Typically, a single multi-DRM HLS ABR ladder suffices
- With a single subtitle/caption format
- And a single trickplay thumbnail format
- Any formats not yet supported only need to be implemented once
- Some packaging duplication may be required to support platform-specific DRM (Specifically with Widevine L1 and Fairplay)

Trickplay Example (Conventional Approach)

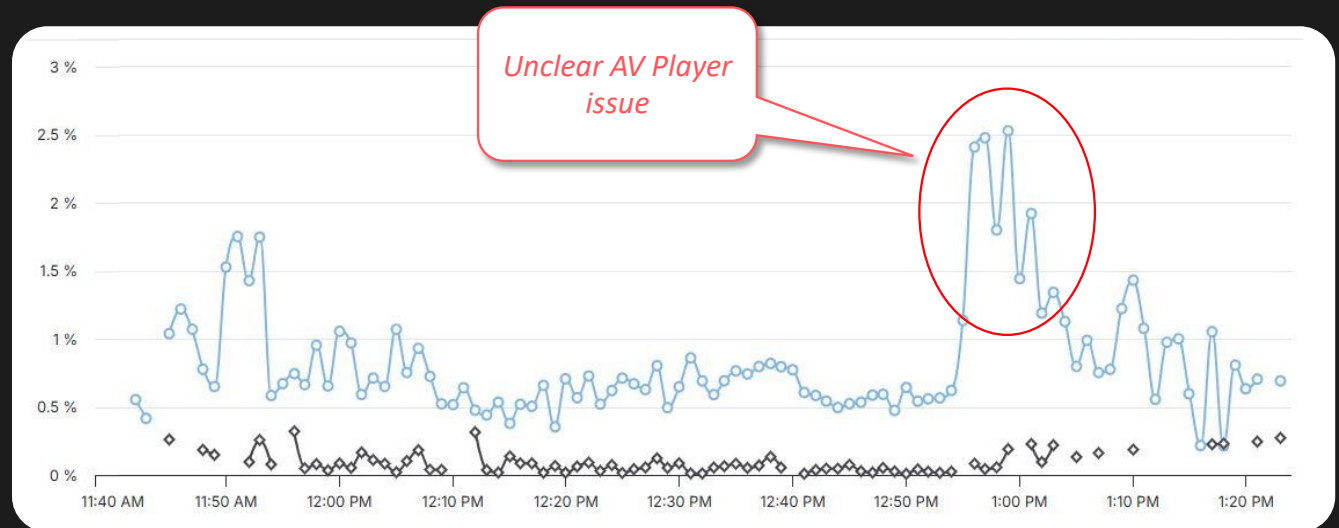
	WebVTT with JPEG/PNG + HLS/DASH	IOP Image Adaptation Set	HLS Image Media Playlist	HLS I-Frame playlist
Web	VoD only	X	VoD only	
Android	VoD only	X	VoD only	
iOS (Vendor UI)	VoD only			
iOS (System UI)				X
iOS (Custom UI)	VoD only			X
tvOS (system UI)				X
tvOS (custom UI)	VoD only			X

- Much more elaborate still for subtitle formats

Errors

- If you build the player yourself, you are accountable for all errors; cannot blame anyone else
- But you have full power to fix any player issues
- Error rates trending towards zero

Conventional player (blue line) vs. Tiledmedia Player (black line) in operational A/B field test with ~1M users worldwide



Debugging

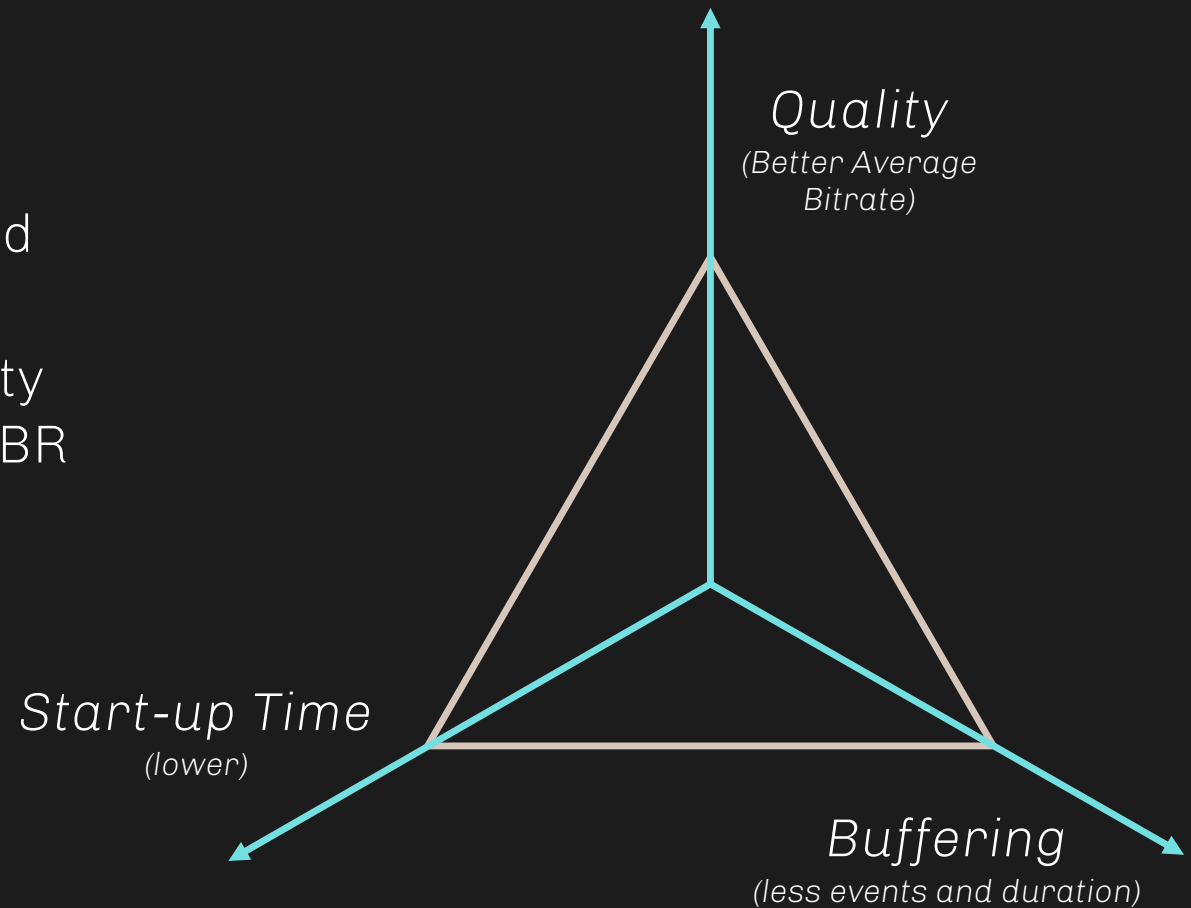
- Can fully understand player state (network, buffer, load, performance, ...)
- E.g., error or crash analysis
- “Core Log”
- Includes all user actions,
- ... which can be replayed in the player o recreate exact conditions

```
09:31.293Z INFO CORE Feed state LF-12 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-13 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-14 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-15 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-16 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-17 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-18 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-19 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-20 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-21 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-22 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Feed state LF-23 | (0/0).301958702->166 (287) | Down./Sched. 00:42.760/00:42.760 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.293Z INFO CORE Audio state | pf0-vt0-it0 | Down./Sched. 00:48.277/00:48.277 | pos 301958702->167 (cts 1739282121322)
09:31.293Z INFO CORE Scheduled switches | false
09:31.293Z INFO CORE Decoder status | 0: inf, frames (0) left in partition (0 partitions queue)
09:31.293Z INFO CORE Tile indices | 0 1 2 0 1 3 4 5 2 3 6 7 8 0 - |
09:31.294Z INFO CORE Tile slice type | p p p p p p p p p p p p p p -
09:31.294Z INFO CORE Tile physical feed | 1 1 1 24 24 1 1 1 24 24 1 1 1 0 -
09:31.294Z INFO CORE Network efficiency | Downloaded 2.4GB | Bitstream 2GB | Efficiency 81.23%
09:31.294Z INFO CORE Network bandwidth | Estimate 81Mbps | Estimate based on 2.2GB | Rate (10s/lifetime) 12.3Mbps/11.7Mbps
09:31.294Z INFO CORE Network perf. | BufferEvents 1 (avg. 0) | TTFB st.avg/lt.avg/min/max 239ms 251ms 16ms 3891ms | Cache hit ratio 0%/2% (100% present)
09:31.294Z INFO CORE Network requests-1 | Act. 0 | Cancel. 15 | Err. 0 | Opp. 0 | Total 4785 | 2.80rps
09:31.294Z INFO CORE Network requests-2 | Video-Generic 1080 | Video-Bckgrnd 0 | Video-Vpwr 0 | Index 0 | Audio 312 | Subtitles 0 | Manifest 3378 | Thumbnails 0 | Other 3:
09:31.294Z INFO CORE

09:31.294Z INFO CORE Sleeping until next frame {"time-to-sleep": "16.217333ms", "target-wait-time": "26.217333ms"}
09:31.298Z DEBUG CORE FrameRendered() called {"poid": 85559}
09:31.298Z DEBUG CORE FrameManager: FrameRendered {"doid": 85554, "poid": 85559, "cts": "483133h55m20.44s"}
09:31.310Z DEBUG CORE Received manifest update {"bundle": "pf0-b6", "time-since-most-recent-segment-added": "14.8751s", "head-seg-idx": 301957809, "tail-seg-idx": 301959473}
09:31.310Z DEBUG CORE Received manifest update {"bundle": "pf24-b2", "time-since-most-recent-segment-added": "6.1671s", "head-seg-idx": 301957809, "tail-seg-idx": 301959473}
09:31.311Z INFO CORE ----- State (running) after 85585 frames | PSI 37 -----
09:31.311Z DEBUG CORE videodecoder.go: sending frame to decoder {"doid": 85586, "poid": 85591}
09:31.311Z INFO CORE Output | Flags: RenderableFrame
09:31.311Z INFO CORE Video Frame <0> | rts (ms): 1713977, pts/dts (ms): 1739282121080/1739282121100, doid/poid: 85586/85591, playback-dur. (ms): 20, 3200x1152p50.00, pr
09:31.311Z INFO CORE Audio Frame | rts (ms): 1714241, pts (ms): 1739282121343, playback-dur. (ms): 21, srp: 301958702->168, track: pf0-vt0-it0, rate-change: fal
09:31.311Z INFO CORE Rendering Video | Enabled true | ShowFromPOID not-set | ShowUntilPOID not-set | VSyncRate: 60.00 | FrameRate 50.00 | Pattern RRRRRRRRRRRRRR...
09:31.311Z INFO CORE Playback Audio | Active true | Underrun false | SyncedAtCTS 483133h35m34.76s | ResumeAfterPOID: 26189 | ResumeAfterTime: not-set | LastDiscontinui
09:31.311Z INFO CORE Playout timing | Position/Length 01:25:47.020 / 02:39:50.400 (301959473->0) | Live Edge: false | Timeshift 1h13m34.331s
09:31.311Z INFO CORE (live) | Playing back 13:55:21 | E2E latency 01:14:10.217 | Live window 301957809-301959473
09:31.311Z INFO CORE (historical) | Running 28:33.988 | Buffering 00:00.383 | Paused -2562047:-47:-16.-512 | Total content time 28:31.700
09:31.311Z DEBUG CORE FrameManager: calculated timeout to pace decoder output {"timeout": "19.818166ms", "frame-count": 59288, "frame-duration": "20ms", "time-since-start":
09:31.311Z INFO CORE (performance) | Perf. comp. 00:00.000(0) | Dec. comp. 00:00.000 | VideoDecoderQueueSize: 1 (overflow total/recent: 0/0) | AudioUnderrun (total/rec
09:31.311Z INFO CORE CPU perf. status | Avg. processing time: 3.63ms/3.67ms | Late 0.06% | Very late 0.00% | Memory 259.9MB/527.4MB | #GR 77 | MF Queues (FF/Ev/AP) 2/0/0
09:31.311Z INFO CORE DisplayObjects | 103 104 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129
09:31.311Z INFO CORE (logical feed) | 0 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 1
09:31.311Z INFO CORE (size) | 1731x974: [129] | 577x324: [103] | 504x284: [104] | 356x200: [108] | 355x200: [106,107,109,110,111,112,113,114,115,116,117,118,119]
09:31.311Z INFO CORE (viewport) | FOV 91.51 x 60.00
09:31.311Z INFO CORE Feed state LF- 0 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 10%/L3/1.2Mbps, avl. 8.8Mbps) | ... L2 (pf0-b4@2.0M)
09:31.311Z INFO CORE Feed state LF- 1 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 58%/L0/6.7Mbps, avl. 47.8Mbps) | L0 (pf1-b0@6.7M) (A
09:31.311Z INFO CORE Feed state LF- 2 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.311Z INFO CORE Feed state LF- 3 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.311Z INFO CORE Feed state LF- 4 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.311Z INFO CORE Feed state LF- 5 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.311Z INFO CORE Feed state LF- 6 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.311Z INFO CORE Feed state LF- 7 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
09:31.311Z INFO CORE Feed state LF- 8 | (0/0).301958702->167 (287) | Down./Sched. 00:42.740/00:42.740 N | 0ms | (max 1%/L7/160kbps, avl. 1.1Mbps) | ... L6 (pf24-b0@0.3M)
```

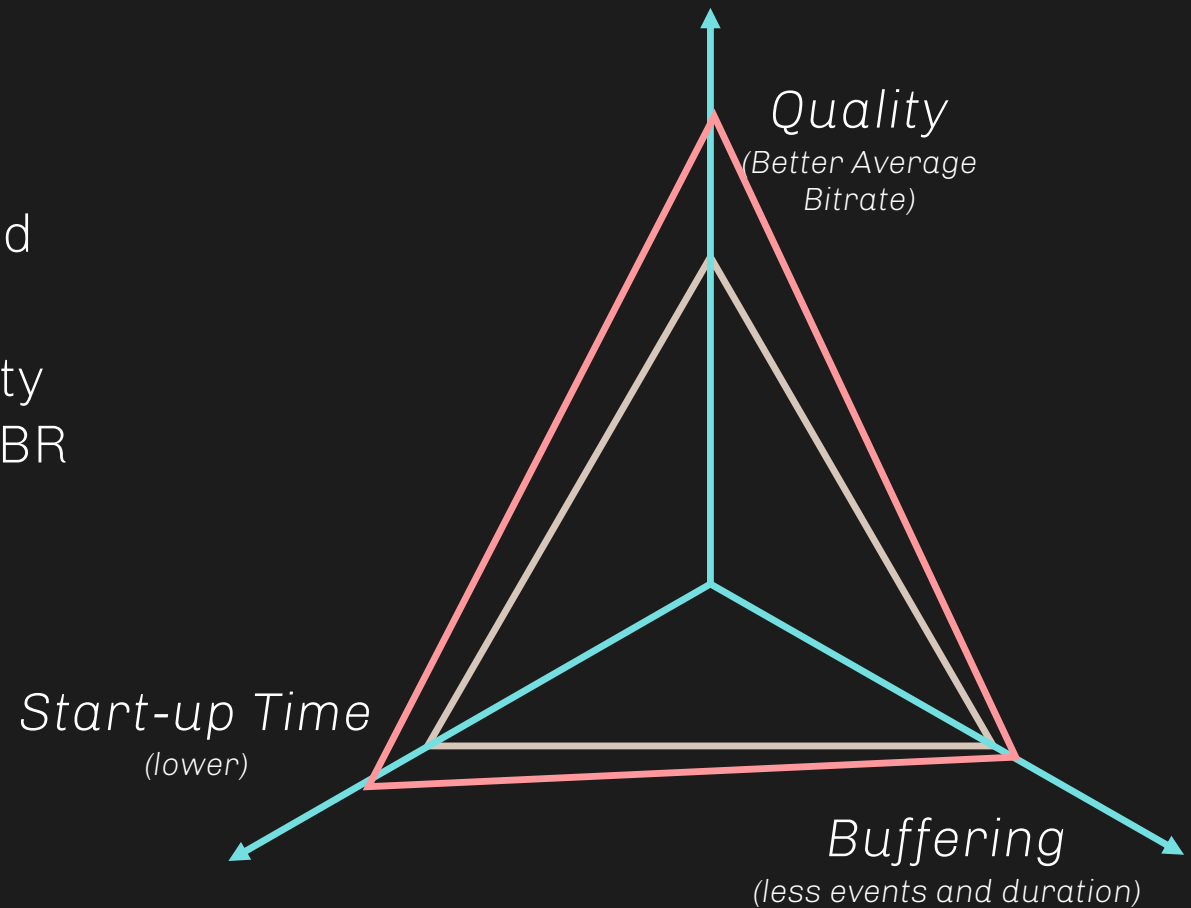
ABR Logic

- Ability to build, control and optimize ABR logic
- Including the unique ability to devise *cross-stream* ABR logic in a multi-stream scenario



ABR Logic

- Ability to build, control and optimize ABR logic
- Including the unique ability to devise *cross-stream* ABR logic in a multi-stream scenario



Nightly Test Runs

- Many Scenarios, such as:

- Stable Quality



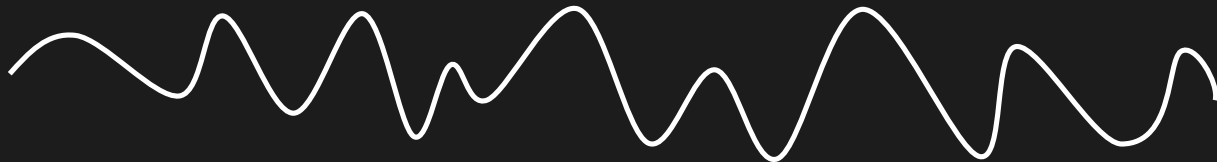
- Out of Data



- Tunnel



- Wave



Example: Lowering Startup Times Without Compromising on ABR Quality & Buffering

	Industry Reference	Tiledmedia Player Run 1	Tiledmedia Player Run 2
Buffer events	2	2	0
Average Startup (seconds)	4.47	2.23	2.90
Average Bitrate (kbps)	1,614	1,546	1,787

Disclaimer: We use these tests to optimize our own ABR logic, so we do our very best to create a valid test with representative conditions. We recognize that it's extremely hard to establish objective metrics. The results should be interpreted as an indication of the "tuneability" of a player that's under full control, not as a definitive comparison between players.

Analytics

- Can measure *all* aspects of experience
- Nothing hidden from view by platform player
- Metrics identical across all platforms
- Bonus: true multi-stream metrics, fully correlated and including aggregate figures

Innovation & Integration

- With full control over the entire stack, easier to build and optimise features to a level that's not otherwise possible
- Not blocked by features that are available on one platform but not another
- Innovation & integration need to be done only once, cross-platform
 - E.g.: cross-platform integration of an ad framework (Yospace): a few working days
- Supporting advanced use cases in the *Player* makes *App* lighter & simpler
 - E.g.: App feed switching logic from 10,000 lines with a conventional player to 20 lines for one customer

Innovation & Integration

- With full control over the entire stack, easier to build and optimise features to a level that's not otherwise possible
- Not blocked by features that are available on one platform but not another
- Innovation & integration need to be done only once, cross-platform
 - E.g.: cross-platform integration of an ad framework (Yospace): a few working days
- Supporting advanced use cases in the *Player* makes *App* lighter & simpler
 - E.g.: App feed switching logic from 10,000 lines with a conventional player to 20 lines for one customer
- Can't inherit functionality from platform players (but maintenance easier once built)

Platform Support

- Every platform requires a platform-specific port

Platform Support

- Every platform requires a platform-specific port
- Once ported, *all* features and benefits are of the single core are available on the new platform

Player Size

- A player that is not a thin layer around something in the OS is inherently larger in size

Player Size

- A player that is not a thin layer around something in the OS is inherently larger in size
- Compensated by having more control over the player memory footprint
- Player can be configured to not exceed some maximum RAM footprint, taking its own size into account, and adapt memory management accordingly (e.g. manage forward content buffer, historical data, etc.)



mhv/2025

Thanks!

Thanks to Thomas Keller for the
nightly test runs