



ISOBMFF: short-comings and evolutions

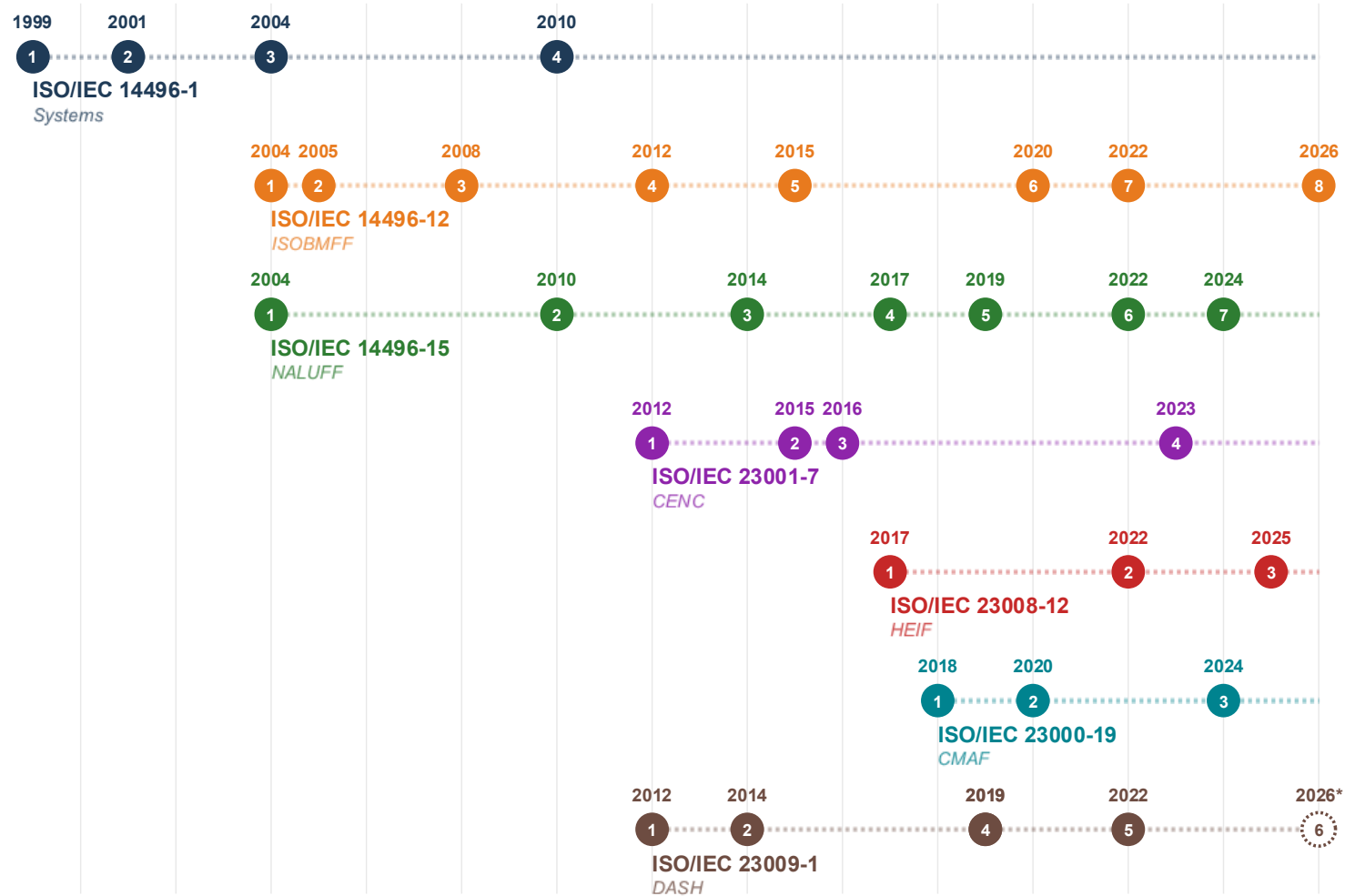
Cyril Concolato

Workshop « Media Streaming Service – What's next »

July 14, 2026

Disclaimer: Personal/MPEG chair views, not company views

Where are we now? The ISO/BMFF Ecosystem



How it started ?



How it's going?



What's next?



Why was ISO/BMFF successful?

The right start

- Underlying format (MOV) from an established company (Apple) with strong products (iTunes, iPod, iPhone ...)
- Co-developed with powerful standards:
 - Video codec – AVC
 - Audio codec – AAC
 - Mobile applications – 3GP



Why was ISO/BMFF successful?

The right technical principles

- Solid multimedia foundations
 - Logical vs physical
 - Synchronization concepts
 - Indexing and random access
 - Codec and media type agnosticism
 - Encryption and DRM support
 - Delivery agnostic but flexible
- Extensible, box-based architecture
- Open to (almost) any contributions



Why was ISOBMFF successful?

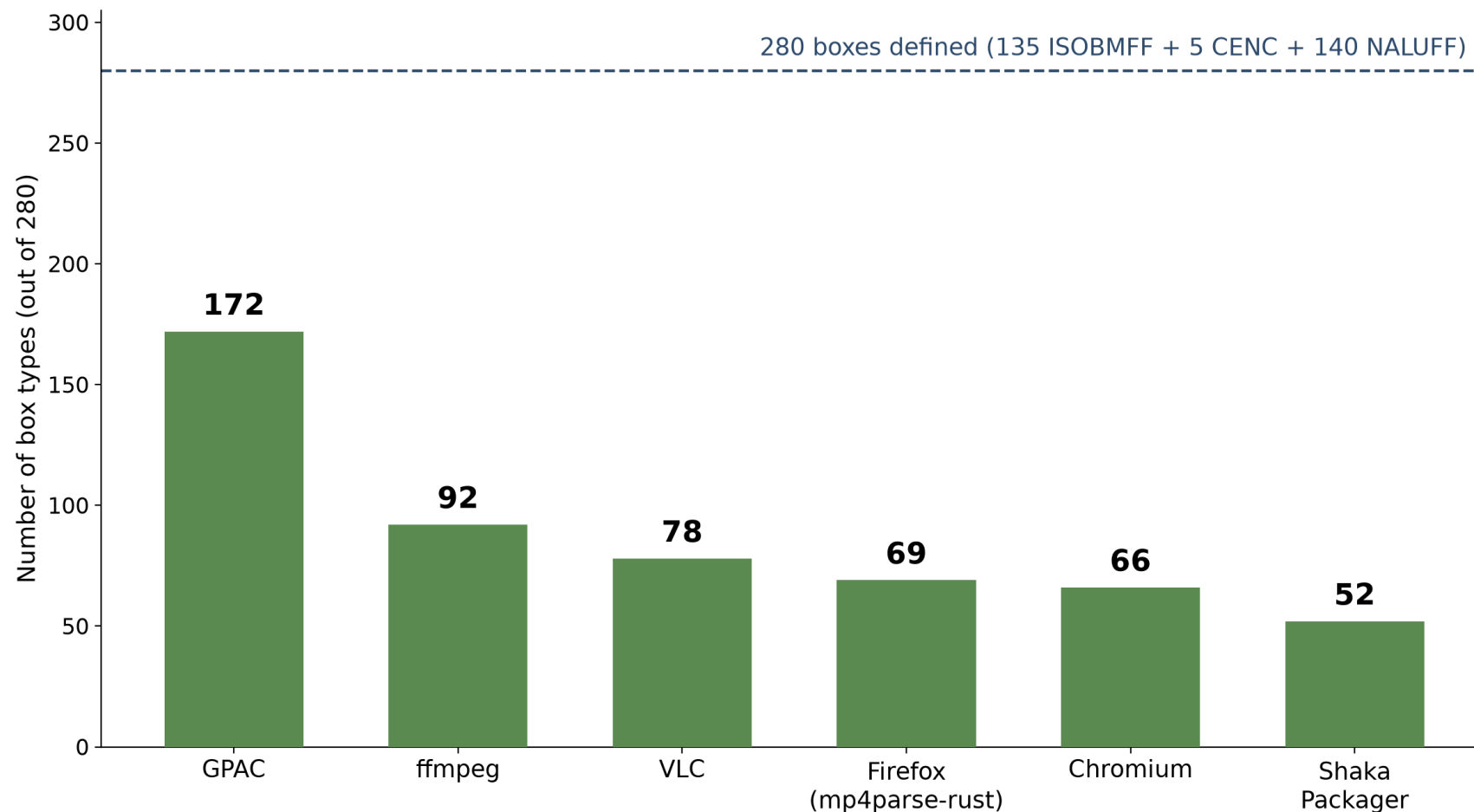
Support for many use cases, evolving over time

- Local files
- RTP Streaming
- Progressive Download
- HTTP Streaming – the DASH/CMAF successes
- Images

ISOBMFF issues

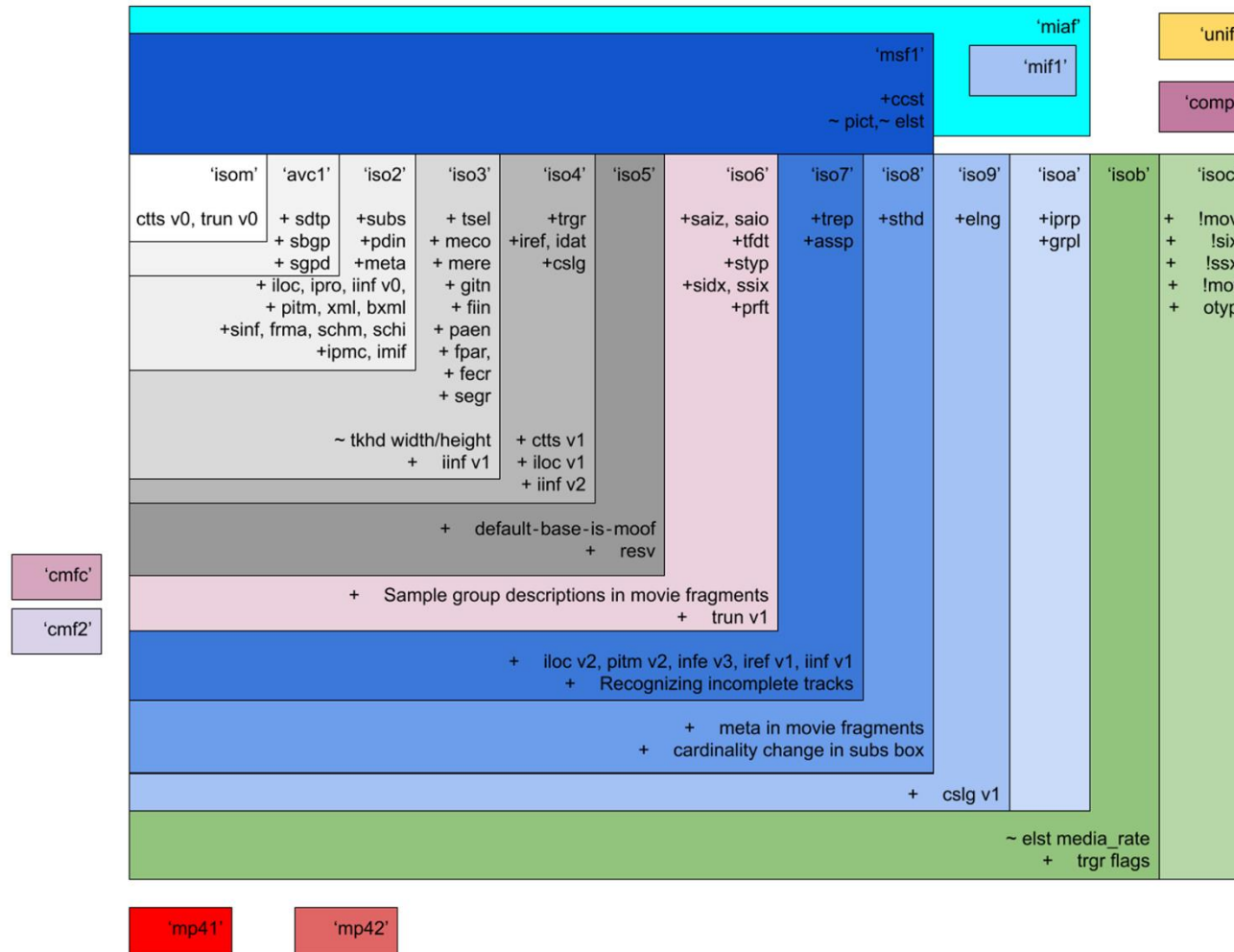
Too many boxes, too little implemented

**ISOBMFF + NALUFF + CENC box-type coverage by open-source project
(exact match count, of 280 defined boxes)**



Exact-match count from box-by-box regex search of each project's fetched source (GitHub, master branch)

Too many brands, too little interoperability



What could be better with ISOBMFF?

"The purpose of this document was to explore whether a simpler, more powerful/modern format was plausible, and how it might work out. [...] This [document] a format that would streamline better into today's environments, notably where bandwidth is rising, but latency [...] is staying steady, or even rising due to buffer delays."

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION

ORGANISATION INTERNATIONALE DE NORMALISATION

ISO/IEC JTC1/SC29/WG11

CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11

MPEG2012/**m24980**

24 April 2012, Geneva, CH

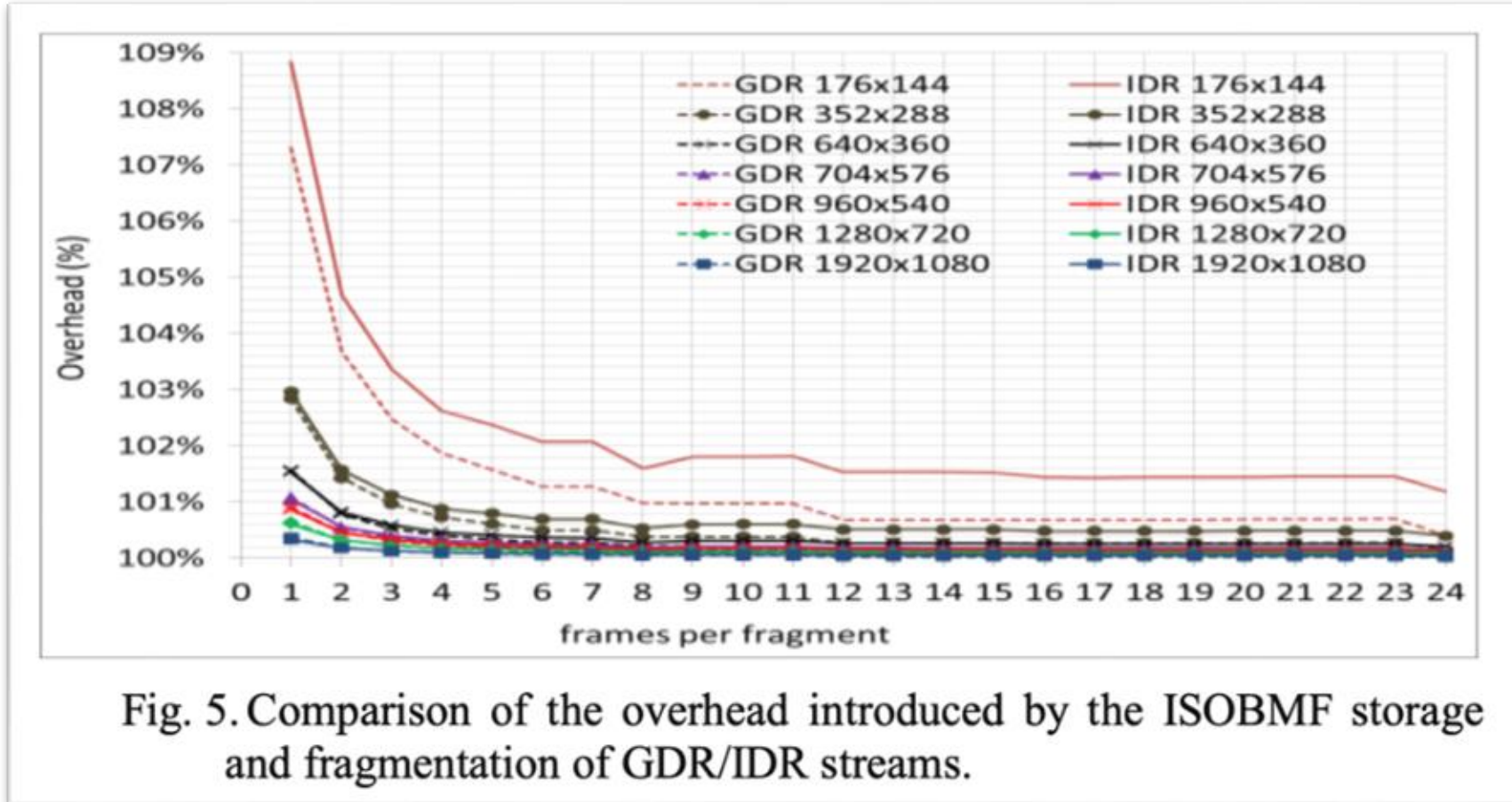
Title:	Exploring a new MPEG file format
Status:	Contribution
Authors:	David Singer
Organization:	Apple Inc.
Group:	Systems/File Format MMT

History repeats

Selected Key Clean-ups from David Singer (2012)

- "no-one needs decode times"
- "the whole design of stream access points, sample groups, and sync points should be made uniform"
- "we don't need old-style sample tables and movie fragment tables "
- "we don't need most of the 'composing' technology in the movie and track headers"
- "carouselizing of initialization information should be possible"
- "edit lists have a number of issues that need addressing"

Low Latency and Packaging Overhead (2014)



Movie Fragment Sizes for Typical Encodes (2018)

Table 1 - Video Movie Fragment Sizes

Min moof size (bytes)	Avg moof size (bytes)	Max moof size (bytes)	Bitrate at 2s/moof (kbps)
732	732	732	2.9
1128	1128	1128	4.5
511	1914	2299	7.6
585	2797	7725	11.1
727	2842	3609	11.4
781	3129	3897	12.5
625	3221	9217	12.9
889	3705	4473	14.8
943	3993	4761	16.0

Table 2 - Audio Movie Fragment Sizes

Min moof size (bytes)	Avg moof size (bytes)	Max moof size (bytes)	Bitrate at 2s/moof (kbps)
199	252	331	1.0
247	714	715	2.9
673	891	907	3.6
273	917	923	3.7
419	1209	1211	4.8

Compacting in ISOBMFF has also always been a concern

- CompactSampleSizeBox ('stz2') :
 - 4/8/16-bit per-sample sizes instead of stsz's fixed 32-bit table
- Compressed boxes:
 - !mov, !mof, !six, !ssx
- CompactSampleToGroupBox ('csgp') :
 - run-length-encoded sample-to-group mapping
- default_group_description_index in sgpd:
 - lets a fragment omit its own SampleToGroupBox entirely
- static_group_description / static_mapping flags in sgpd
 - declare up front that no per-fragment tables of a given type exist



LOCMAF (2026)

- Compact wire encoding for single-track CMAF chunks, specifically moof/traf/trun
- Initial chunk unchanged, subsequent chunks are delta-encoded
- Compatible with CENC/DRM (including regeneration of saio/saiz from senc)
- Escape mechanism for chunks that are not in scope

Workgroup: Media Over QUIC
Internet-Draft: draft-einarsson-moq-locmaf-01
Published: 5 July 2026
Intended Status: Informational
Expires: 6 January 2027

T. Einarsson
Eyevin Technology

Low Overhead CMAF for Media over QUIC (LOCMAF)

Abstract

This document specifies LOCMAF (Low Overhead CMAF for Media over QUIC), a compact packaging for low-latency CMAF media carried end-to-end as MoQ Transport (MOQT) Object payloads, with per-object overhead comparable to the Low Overhead Container (LOC). LOCMAF carries the CMAF chunk head metadata from a single moof (movie fragment) as a small set of tagged fields, while leaving the sample data (mdat) untouched. Boxes that may surround the moof in a CMAF chunk – styp (segment type), prft (producer reference time), and any number of emsg (event message) boxes – are carried verbatim, each through a generic box element (a genBox). The first Object of each MOQT group carries a full reference; subsequent Objects in the same group carry only the differences. The receiver reconstructs CMAF chunks that are decode-equivalent to the sender input, including the encryption metadata required by CMAF DRM (Common Encryption) pipelines, and a canonical byte-identical reconstruction – independent of the encoder's representation choices – is defined for conformance testing.

Low-overhead Image File Format

- Amendment 2 of HEIF 3rd edition
- Breaking change
 - No moov or meta top-level box
 - A compact 'mini' box
- Equivalence with MetaBox
- Significant overhead savings for small images

Table XX3 — Required boxes in a file under the 'mif3' brand

Hierarchy of boxes			Version	Box description
<u>ftyp</u>			-	file type and compatibility
<u>mini</u>			0	metadata and image data

```
aligned(8) class MinimizedImageBox extends Box('mini') {  
    bit(2) version = 0;  
  
    // flags  
    bit(1) explicit_codec_types_flag;  
    bit(1) float_flag;  
    bit(1) full_range_flag;  
    bit(1) alpha_flag;  
    bit(1) explicit_cicp_flag;  
    bit(1) hdr_flag;  
    bit(1) icc_flag;  
    bit(1) exif_flag;  
    bit(1) xmp_flag;
```

What's next?

Context: end-user bandwidth has grown roughly two orders of magnitude in 20+ years

Period	Typical fixed broadband	Typical mobile
~2003–2005	~1–3 Mbps (early DSL/cable)	~50–200 kbps (2.5G/EDGE)
~2010	~5–10 Mbps	~1–3 Mbps (3G)
~2015	~15–25 Mbps	~10–20 Mbps (early LTE)
~2020	~40–80 Mbps (rising fiber share)	~25–50 Mbps (mature LTE)
~2025–2026	~100–300 Mbps (fiber/DOCSIS 4.0 common in many markets)	~50–150 Mbps (5G, highly variable)

Order-of-magnitude figures, global-average trend consistent with Akamai State of the Internet, Ookla Speedtest Global Index, and ITU broadband reports — actual numbers vary widely by market and access technology; the point is direction and rough scale, not precision. Median growth: roughly 50–100x for fixed broadband over ~20 years.

Context: video codec efficiency has also improved by roughly an order of magnitude in 30+ years

Codec (year standardized)	~Bitrate vs. predecessor at equal quality
MPEG-2 (1994)	baseline
H.264/AVC (2003)	~50% of MPEG-2 for equal quality
HEVC/H.265 (2013)	~50% of H.264
AV1 (2018) / VVC/H.266 (2020)	~30–50% further reduction vs. HEVC (claims vary by content/encoder maturity)
AV2 / next-gen (in development, ~2026)	another ~30–40% reduction targeted vs. VVC/AV1, per early standardization results

Figures are the commonly-cited standardization-era bitrate-savings targets (MPEG/JVET, AOM); real-world gains depend heavily on encoder implementation maturity and content type, and typically lag the headline number for years after a codec ships.

Context: what's actually being streamed has changed just as much

- Mobile now carries the majority of video traffic in most markets — phones, not desktops or set-top boxes, are the primary playback device for a huge share of viewing
- Resolution/quality ceiling has risen sharply: HD to 4K mainstream delivery, wide-gamut HDR, high frame rate — each adding bits per pixel even as codecs improve
- New delivery patterns didn't exist 20 years ago:
 - short-form vertical video at massive scale,
 - live/social streaming,
 - cloud gaming (extremely latency-sensitive),
 - real-time video calling
- Device and network diversity has exploded: the same content now has to reach everything from gigabit fiber TVs to congested mobile links in transit — the spread between best-case and worst-case conditions is wider than it was 20 years ago, not narrower
- New use cases: personalization, ads splicing, hybrid delivery (push+pull) ...

Before optimizing anything: which metric are we actually moving?

- "Reduce overhead" is not one goal — it's shorthand for several distinct, sometimes competing metrics:
 - Storage / CDN egress cost — bytes at rest and bytes served; the classic target of LOCMAF-style wire optimization
 - Client-perceived latency — startup delay, rebuffering, live-edge latency; overhead matters here mainly via header parsing cost and packet count, not raw byte count
 - CPU and battery on the receiving device — parsing/box-traversal complexity, not just payload size
 - Memory — buffer sizes and parse-ahead requirements imposed by the container's structure
 - Network efficiency — packet count and per-request protocol overhead, which chunk-size choices affect independently of container-box overhead
 - Operational complexity — pipeline cost of producing and storing many renditions, largely orthogonal to per-chunk box overhead
- **Container-overhead reduction is a strong lever mainly on the first metric, and mainly at the low-bitrate/low-latency intersection identified above — it is a weak lever on most of the others**

Is now a good time to rework ISOBMFF, non-backward-compatibly?

- Lesson from adjacent industries: old formats don't die

Format	Introduced	Status today
MPEG-2 Transport Stream	1995	Still the dominant transport for broadcast, cable, and satellite delivery worldwide, three decades on
JPEG	1992	Despite HEIF, AVIF, and others all offering better compression, still by far the most common image format on the web

- Every encoder, packager, CDN, player, DRM system, analytics pipeline, and archive has MP4-shaped assumptions baked in
- A non-backward-compatible format demands: new or updated tooling across the entire chain, dual-format support during a transition period, conversion of legacy libraries, and re-certification