BRIGHTCOVE

ON THE EVOLUTION OF VIDEO AND STREAMING

Yuriy Reznik
Brightcove, Inc.

FOKUS Media Web Symposium
June 22, 2022

Outline

A brief look at the history of video

- Review some interesting facts in history of video overall
- Why we are using frames, scan orders, pixels, YUV color spaces, etc.?
- What was before streaming?

Evolution of streaming

- Early systems
- ABR streaming before HTTP
- ABR streaming with HTTP
- Evolutions ABR systems

What may come next?

- New forms of “video”
- In pursuit of lower delays
- Back to purposedly built video (or metaverse) networks?

Evolution of video

Evolution of video technologies

BRIGHTCOVE

THE PAST:

Invention of camera, still image photography, color reproduction, film, moving pictures

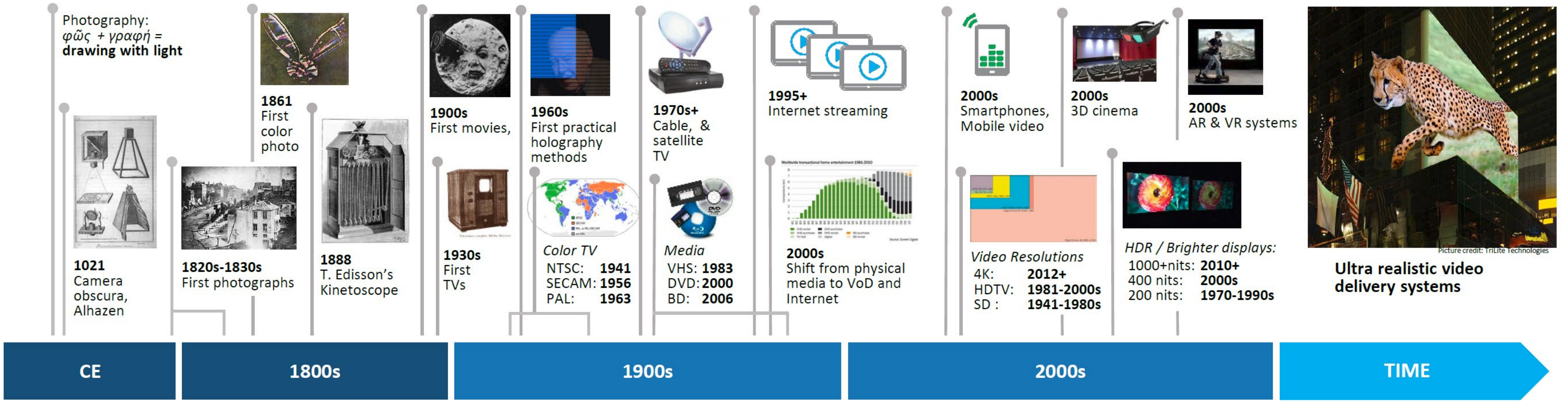
THE PRESENT:

New delivery methods: TV, recordable media, digital compressed formats, Internet streaming, mobile.

Increasing degree of realism: immersive video, 3D (holography, stereoscopic rendering, etc.)

THE FUTURE:

Recording & reproduction systems making rendered video undistinguishable from reality.



Everything we know about video are the results of human inventions

- Cameras, photographs, film, CCDs, digital media formats, displays, TVs, compression algorithms, streaming, etc.
- But as time progresses, we often forget what, why, and for which reason was initially invented.

Examples of some early decisions

Frames and framerates

- 24fps → first film projectors (T. Eddison & Co., 1930s)
- 25/30fps → first B&W TV receivers, synchronized by 50/60Hz AC (1940s)
- 29.97fps → NTSC (1953), fitting chroma in same band as allocated for B&W TVs

Lines and scan orders

- 1880 – Maurice Leblanc's patent
- 1931 – first CRT tubes and CRT-based TV systems (V. Zworykin et al, RCA).
- 1937 – 240 lines TV systems
- 1941 – 441 lines TV systems
- 1948 – 525 and 625 lines TV systems (**all interlaced!**)

YUV color spaces

- Designed in 1938(!) for backwards compatibility with B&W TV systems
- Luma = "intensity" in earlier systems, "chroma" = complementary channels
- Variants: YPbPr, YDbDr, YIQ, YCbCr, etc.

XYZ → XYZ→RGB matrix multiplication → YUV gamma → R'G'B' → Y'UV matrix multiplication → Y'UV

24 fps framerates

Framerate adopted in film movie projectors. 1930s. T. Eddison. Note: first film cameras were hand-cranked!

Scan orders

Maurice Leblanc, "Etude sur la transmission électrique des impressions lumineuses", La Lumière Électrique, **Dec 1, 1880.**

YUV color space

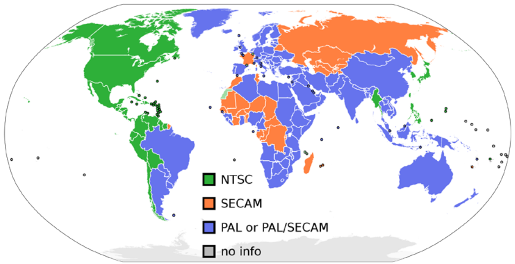
Invented in 1938 by Georges Valensi as a mean to make color TV system compatible with B&W TV receivers. Y channel in YUV was meant to be B&W TV signal.

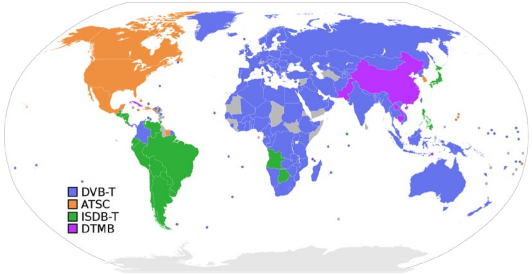
BRIGHTCOVE

What was before Streaming?

Video broadcast systems

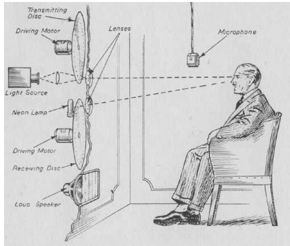
- Terrestrial, DHT satellite, Cable, hybrid.
- Several generations (from analog NTSC/PAL/SECAM in 1950a to digital ATSC/DVB/ISDB/TDMB in 1990s) been deployed
- They all used **purposedly built video distribution networks and receivers** to deliver video to the masses





Video conferencing systems

- 1927 – AT&T's first demo of video phone
- 1959 – AT&T's Picturephone (180p, 40kbps)
- 1976 – NTT, Mitsubishi AtariTel (48kbps)
- 1982 – CLI video phone system (first digital!)
- 1986 – PictureTel – first successful system
- 1990s – H.324 & H.323-based systems
- **Objective: 2-way real-time communication!**

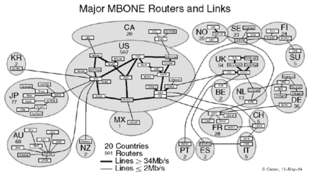


Evolution of Internet Streaming

First streaming systems

1993: MBONE

- Virtual multicast network connecting several universities & ISPs
- RTP-based video conferencing tool (vic) is used to send videos
- 1994 Rolling Stones concert – first major event streamed online



1995: RealAudio, 1997: RealVideo

- First commercially successful mass-scale streaming system
- Proprietary protocols, codecs: PNA, RealAudio, RealVideo
- Worked over UDP, TCP, and HTTP ("cloaking" mode)
- First major broadcast: 1995 Seattle Mariners vs New York Yankees

1995+: VDOnet, Vivo, NetShow, VXtream, …

- Many vendors have tried to compete in streaming space initially
- Vivo & Xing got acquired by Real, VXtreme by Microsoft
- By 1998, 3 main vendors remained: Real, Microsoft and Apple

1998: RealSystem G2

- First ABR streaming system

First innovations in streaming

Introduction of pre-roll delay

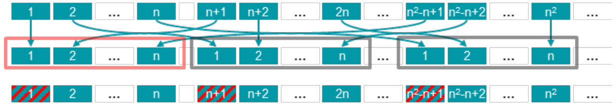
- Many early systems (Vivo, VDOnet, etc.) have tried to use H.324 / H.323- video conferencing stacks for streaming. But they worked very poorly!
- The first important discovery and deviation in the design of streaming systems from video conferencing was the *introduction of a much longer initial delay!*

Original uses of pre-roll delay / buffer

- Leaky bucket: reducing probability of stalls with network bandwidth fluctuations
- Reordering of out-of-order received UDP packets
- Limited retransmissions (ARQ) – unlimited ARQ or TCP was simply non-practical !
- Interleaving / multiple-description coding of audio

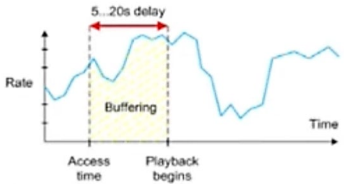
Interleaved packetization (RealAudio, 1995):

- 20-ms audio frames after encoder:
- UDP packets:
- Effects of loss of a packet:

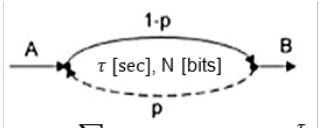


- Missing audio frames were by-directionally predicted/synthesized during decoding.
- This worked remarkably well even with heavy (5-10%) packet loss rates!.

Initial delay:



Expected delay & throughput in a system with unlimited retransmissions:

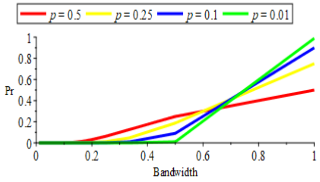


$$\bar{T}(A,B) = \sum_{i\geq 0}(1-p)p^i\,(1+i)\tau = \frac{\tau}{1-p}$$

$$\bar{R}(A,B) = \sum_{i\geq 0}(1-p)p^i\,\frac{N}{(1+i)\tau} = \frac{N}{\tau}\frac{(1-p)}{p}\log\left(\frac{1}{1-p}\right)$$

$$\Pr\left(R = \frac{N}{(1+i)\tau}\right) = (1-p)p^i$$

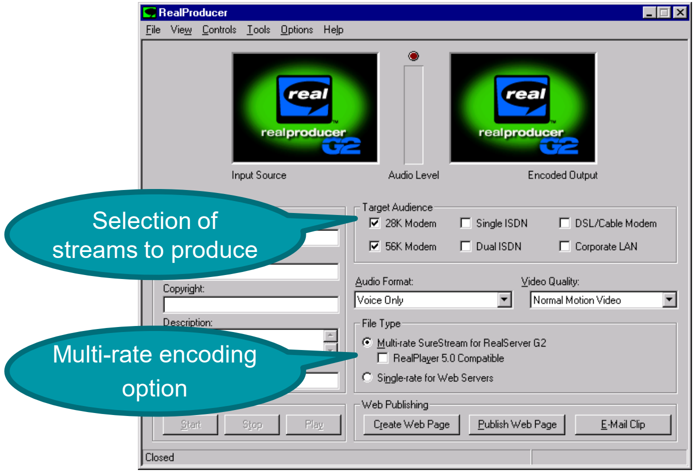
Bandwidth distribution:

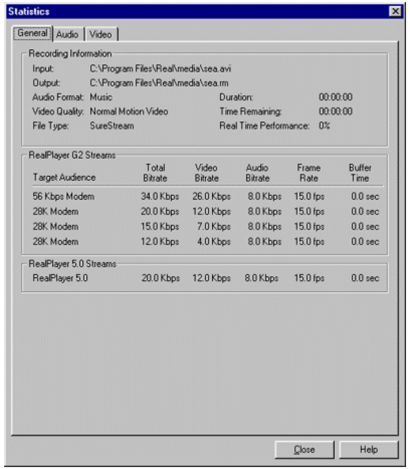


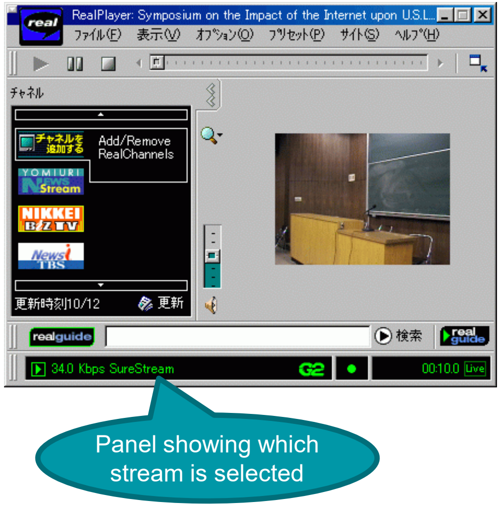
First ABR Streaming System

1998: RealSystem G2: “SureStream”

- First commercially successful ABR streaming system
- Encoder: Encoded streams Player







Related publications & patents

- B. Girod, et al, “Scalable codec architectures for Internet video-on-demand,” ACSSC, pp. 357 – 361, 1997.
- G. Conklin, et al, “Video Coding for Streaming Media Delivery on the Internet," *TCSVT*, 11 (3), pp. 20-34, 2001.
- US Patents: 6314466, 6480541, 7075986, 7885340

First streaming standards

1998: RTSP – Real-Time Streaming Protocol

- Session protocol for packet-bases streaming
- Main contributors: RealNetworks, Netscape, Columbia University
- Uses as foundation for most streaming systems of 1998-2008 era

2000: ISMA – Internet Streaming Media Alliance

- Forum created by Apple, Cisco, Kasenna, Philips, and Sun
- ISMA 2.0: RTSP+RTP+RTCP + H.264 and HE-AAC codecs
- ISBMFF with hint tracks is employed for storage of encoded streams
- ISMA 2.0 was supported by many servers and clients of that era

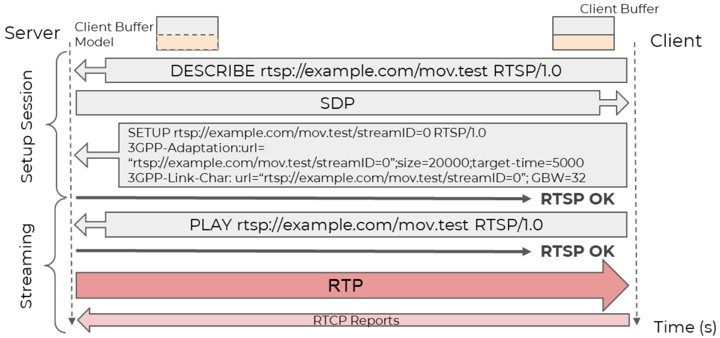
2006: 3GPP PSS – Packet Switched Streaming

- Describes RTSP+RTP+RTCP ABR adaptive streaming system with several standard video, audio and speech codecs
- 3GPP version of RTSP/RTP-based stack

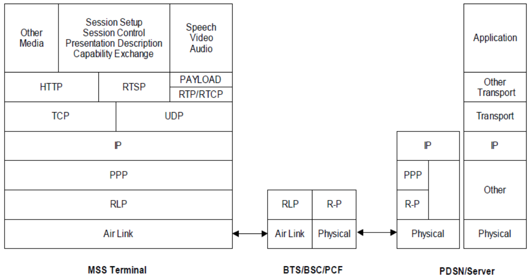
2006: 3GPP2 MSS – Multimedia Streaming Services

- Similar to 3GPP PSS, but differs in speech codecs & network stack

Session setup and streaming phases:



Full protocol stack in 3GPP2 MSS:



Why Today's streaming use HTTP?

Networks have improved!!

- When streaming started, 28k and 56k modems were the common connections available
- But by mid-2000s consumers moved to Cable, DSL, or other high-speed connections
- Bitrates were up 5-100x, latencies were 4-10x down, packet losses were under 1-2%
- This relaxed requirements dramatically!
- Progressive downloads become feasible alternatives to streaming!

CDNs become ubiquitous

- By mid-2000s Akamai, Limelight and other CDNs were well deployed all over
- CDNs provided much better density and reach than RTSP-based delivery networks (RBN, etc.)

Other practical & business reasons

- The space was fragmented: Real, Microsoft, Apple, and then Adobe used significantly different implementations of their stacks. Even codecs and file formats were different! RTSP and ISMA offered only "baseline" level of interoperability!
- RTSP systems were complex: servers and clients were extremely complex, error concealment was a major pain, etc.

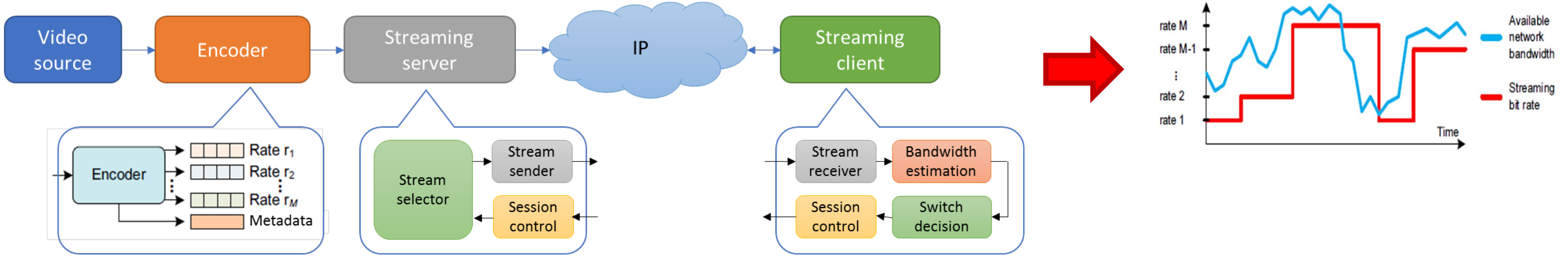
And… one day a much simpler solution was found

- Store encoded media streams in 5-10sec chunks on a web server… pull them using HTTP GET, catenate, and play
- About same delays, no packet losses or retransmissions, and with good enough networks – it may just work…

ABR systems & their evolutions

How first ABR system worked?

RTP/RTSP-based ABR streaming architecture:



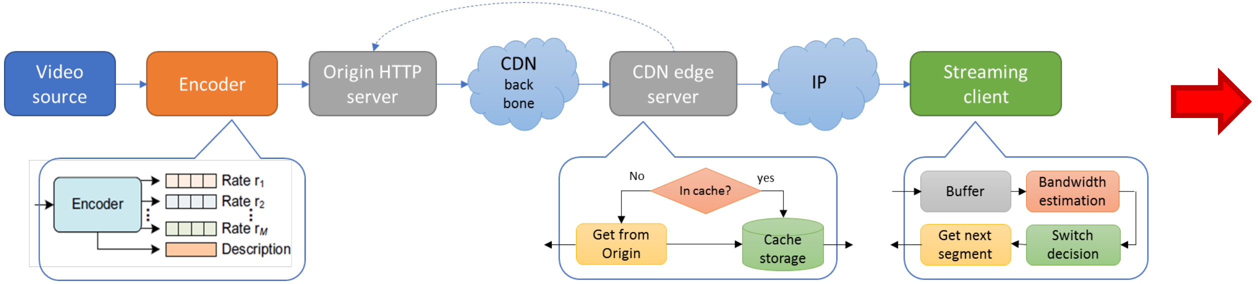
- Public internet is used for delivery
- RTSP protocol was used for session control, and UDP (plus RTP or proprietary transport) were used for sending the data
- Stream adaptation was done by server, but with most clients – it was client-driven: client was sending requests to switch
- Server was also responsible for retransmissions, injecting extra FEC packets, etc.
- Everything was sent in “packets”

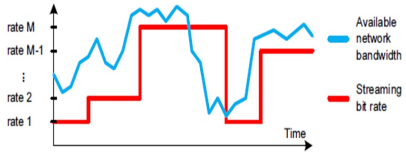
Important design elements:

- **Only one stream** was sent of over IP for delivery to each client!
- Multiple renditions were stored only on the (origin) streaming server, and transmissions of such “stacks of streams” to other servers was not even envisioned.
- This was all before CDNs and relay networks for streaming!

HTTP-based ABR Streaming

Modern-era HLS/DASH streaming architecture:





Key differences from RTSP/UDP streaming:

- instead of streaming server, a regular HTTP server is used as origin
- stream switching is trivialized to HTTP GET operations originating from streaming client
- the scaling and delivery is delegated to CDN, which caches content on the edge servers, reducing the load on the origin...

Important new factors:

- This works well when the content is *"popular"* and it becomes *cached* in the edge cache
- If content is not popular, and not stored at the edge cache – it becomes pulled from the origin server (in which case CDN only adds latency and increases cost of delivery)
- In other words – *optimistically CDN helps*, but in the worst case – it does not!

Disconnect between ABR and CDN models

Key issues:

- ABR systems fundamentally need **several encoded versions of the content:**
  - Multiple streams are needed to achieve better network adaptation and minimize the visibility of stream switches.
  - Multiple streams are also needed to support different delivery formats (HLS, DASH, MSS, etc.) and DRM systems.
  - Support for multiple video codecs (H.264, HEVC, AV1, and VVC) also results in a creation of multiple streams

- However, once multiple streams are created, and different client start pulling different versions of then – such streams start **“competing” for the CDN edge cache disk space.** This results in mode CDN cache misses, and higher load on origin server. This also increases delivery costs and makes whole system less reliable, less scalable, etc.

- In other words, while **ABR streaming concept promotes the creation of “more” streams, what CDNs need to be the most effective is “less”!**

CDN cache misses with multiple streams

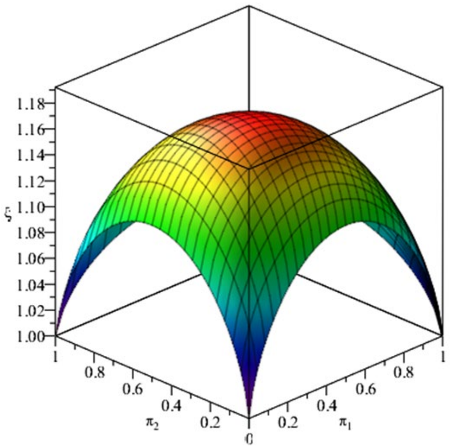
Analytic model:

- Asymptotically, the use of k streams increase CDN cache miss probability by a factor

$$\xi(\alpha,\pi) = \frac{p_{miss,k}(C,\alpha,\pi)}{p_{miss}(C,\alpha)} \sim \left(\sum_{i=1}^{k} \pi_i^{\frac{1}{\alpha}}\right)^{\alpha} = \|\pi\|_{\frac{1}{\alpha}}$$

- where: α is a parameter of content popularity model, and $\pi = \{\pi_1, \ldots, \pi_k\}$ are the usage probabilities of each stream
- *Y. Reznik et al, "On multiple media representations and CDN performance", MHV 2022*

Relative increase in cache miss probability in case of using 3 formats.



Observations:

- the worst impact happens when all formats are equally probable: $\pi_1 = \cdots = \pi_k$
- the higher is the asymmetry in usage of different formats (or renditions), the better it is from CDN efficiency standpoint: $\pi_i \to 1 \Rightarrow \xi(\alpha,\pi) \to 1$

Possible solutions / workarounds:

- Reduce the number of streams;
- Pick one "preferred" representation, and direct as many possible clients/devices use it
- Consider alternatives to "simulcast ABR": scalable coding, multiple description coding models, etc.

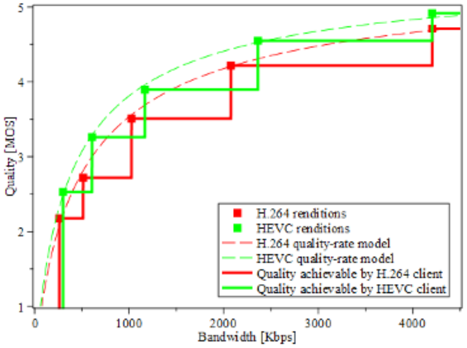
Multi-codec systems

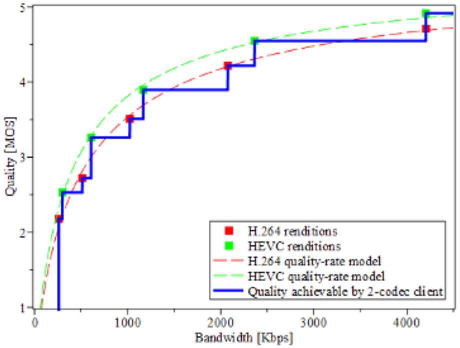
Multiple codecs bring more problems to CDNs:

- Even as newer codecs are getting better, adding new streams to CDNs may increase delivery costs instead of reducing them!
- Old streams must be retained for compatibility with older systems!

Smart multi-codec ABR ladders:

- ABR ladder generation with 2+codecs and interleaved bit-allocation ➔ saves the total number of streams needed
- *Y. Reznik, et al, “Towards Efficient Multi-codec streaming”, NAB 2022:*



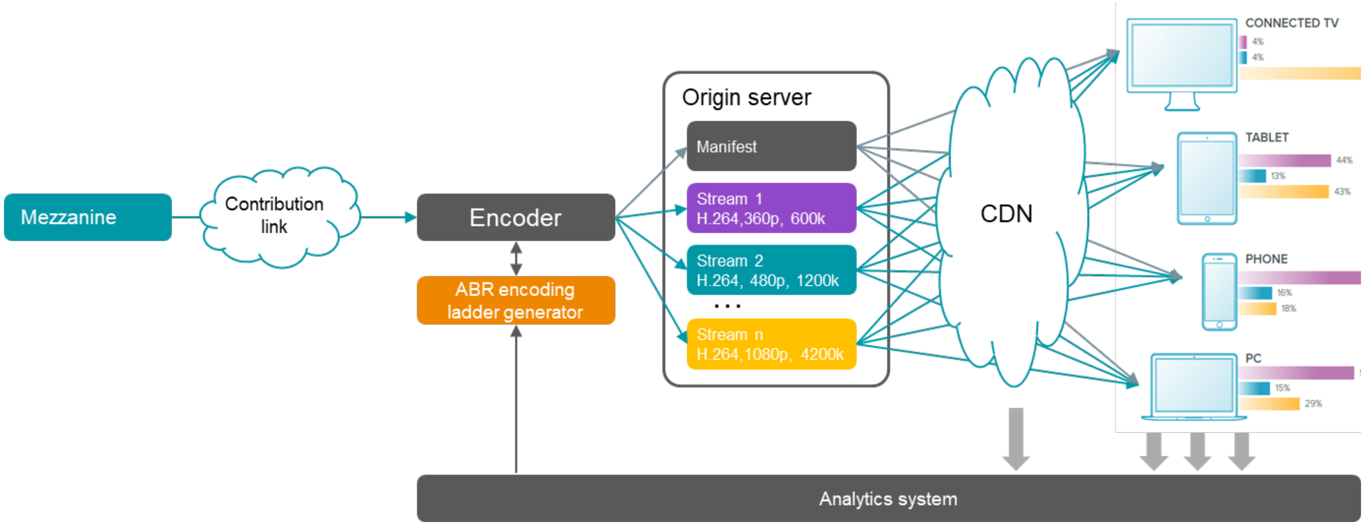


Is this the ultimate solution?

- Indeed no! Codecs fragmentation is a human-created problem!
- Better technical solution: force convergence to the same codec, and make it scalable or MD-capable!

CDN-aware ABR ladder construction

With ABR systems, the ladder design emerges as key for end-to-end optimization:



ABR ladder design techniques:

- Per-title or “content-aware” → take into account only properties of content
- Playback statistics or “networks-aware” → take into account playback statistics as basis for optimization
- “Context-aware” → take into account both properties of content, as well as its popularity and CDN- and network-related statistics

Y.Reznik, et al, "Optimal design of encoding profiles for ABR streaming“, Packet Video, 2018

Y.Reznik, et al, “Optimizing Mass-Scale Multiscreen Video Delivery," SMPTE Motion Imaging Journal, vol. 129, no. 3, 2020

What may come next?

BRIGHTCOVE

Future evolutions

New forms of video

- SD->HD->UltraHD, SDR->HDR, 30 degrees -> 360 degrees
- 2D/single view->stereoscopic->multi-view->light field representations
- Real world -> metaverse(s)
- Dependencies: displays, cameras, graphics stacks, and only then delivery systems

Towards lower delays

- HLS/DASH: 10-30sec, LL-HLS/DASH: 3-6 sec
- Back to UDP: WebRTC: 200-500ms
- Cross-layer Phy->App stacks: 30-100ms (subject to distance, topology, etc.)
- Extreme low-delay case:
  - If ultra-ultra-low delay (~30ms) becomes achievable, then we don’t need much bandwidth!
  - All we need to send is *about 1-2 degrees spot at each moment! [foveated video, eye-tracking-based systems]*
  - Perceptually perfect transmission can be accomplished at about 700kbps or less

Back to purposedly built video networks?

- Streaming started by the idea of sending video over networks designed for data
- But nowadays video is already consuming over 80% of Internet bandwidth!
- Internet is becoming a “video-first” network.. or “metaverse-first”.

THANK YOU