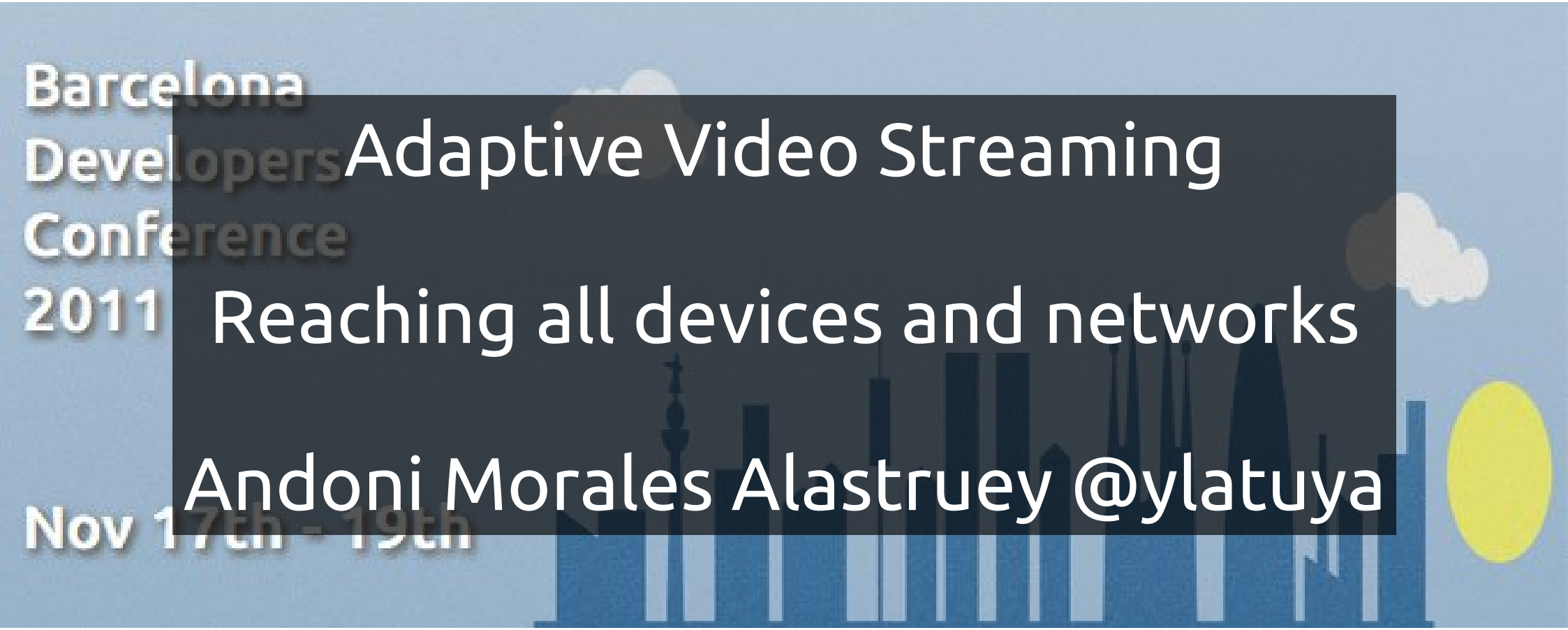
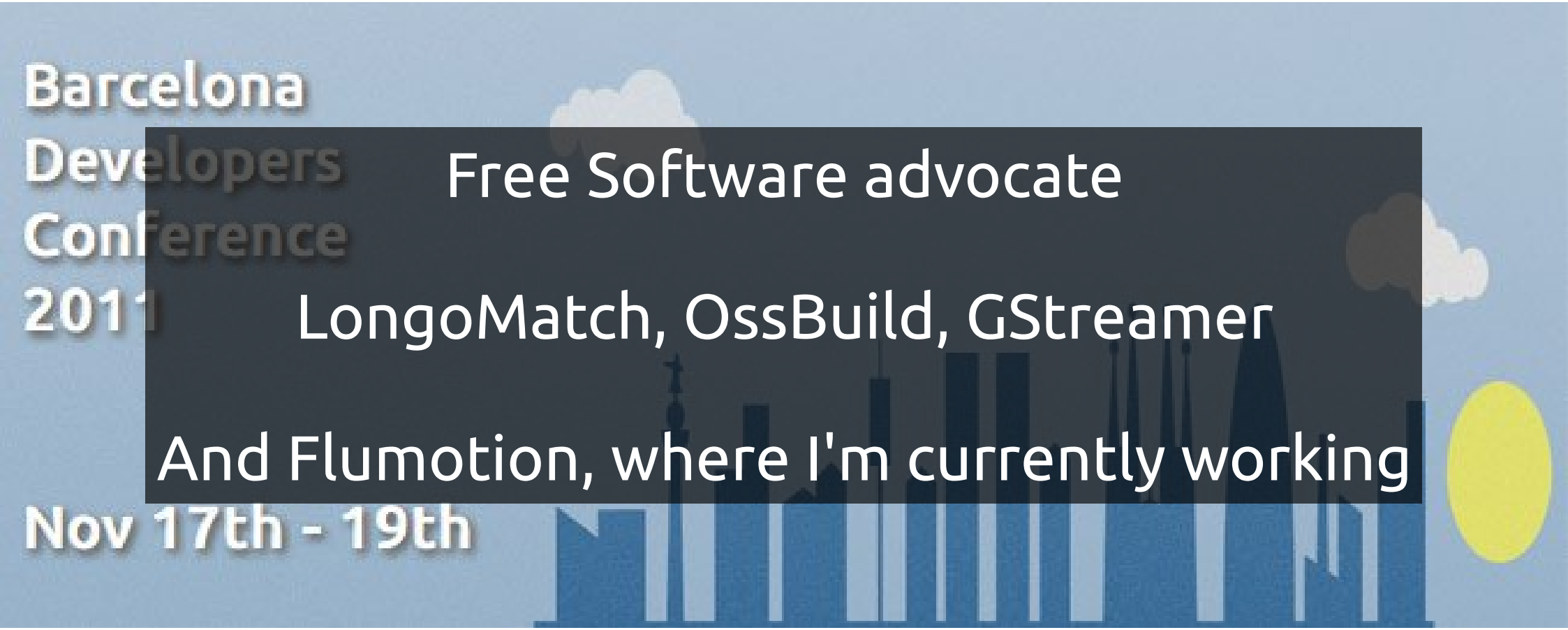


Adaptive Video Streaming

Reaching all devices and networks

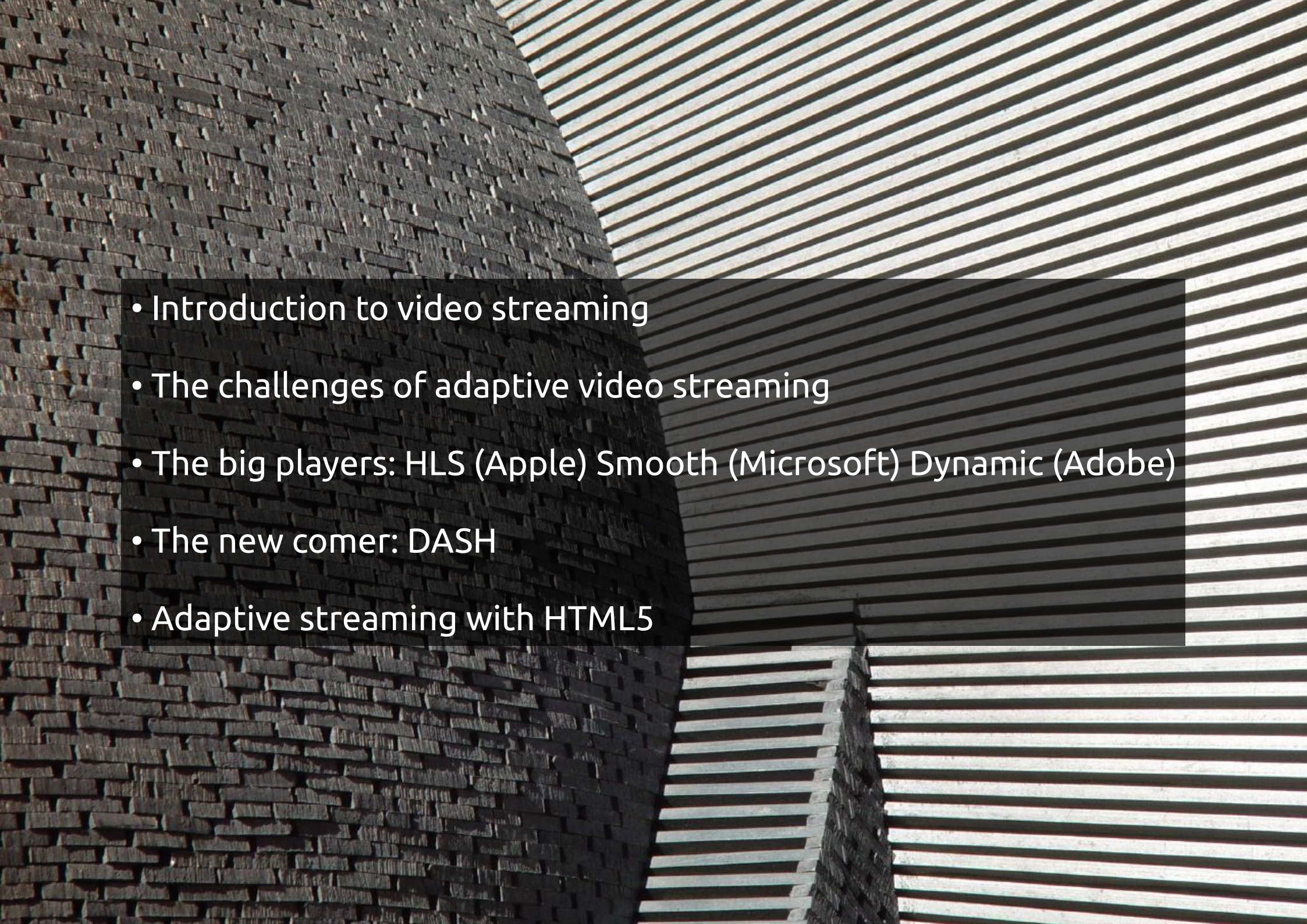
Andoni Morales Alastruey @ylatuya

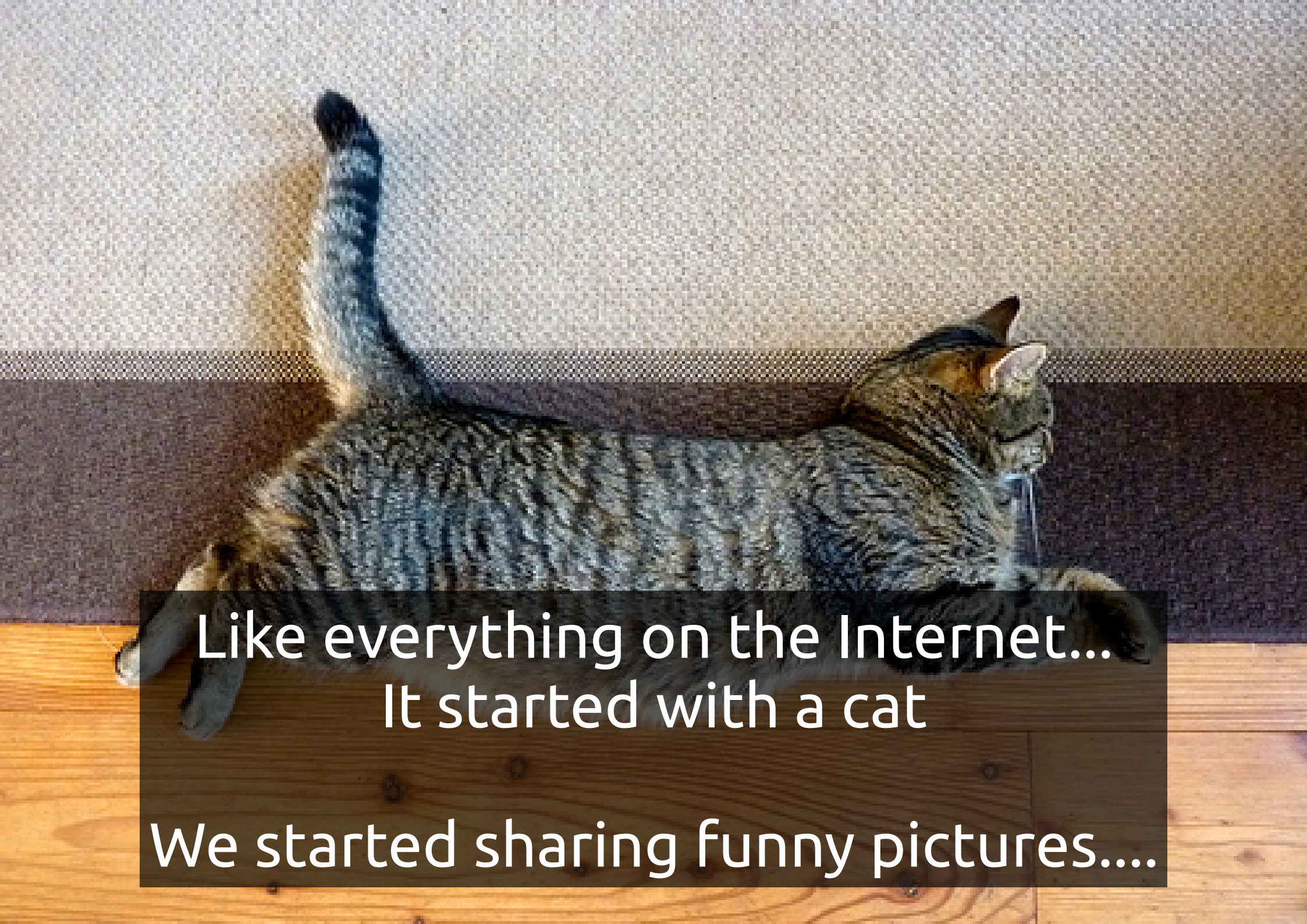


Free Software advocate

LongoMatch, OssBuild, GStreamer

And Flumotion, where I'm currently working


• Introduction to video streaming
• The challenges of adaptive video streaming
• The big players: HLS (Apple) Smooth (Microsoft) Dynamic (Adobe)
• The new comer: DASH
• Adaptive streaming with HTML5


Like everything on the Internet...
It started with a cat
We started sharing funny pictures....

And that's how video started too...

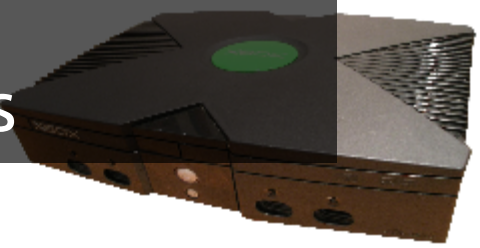
ADSL

FFTH

LAN

INTERNET

- A few years ago the ecosystem of video streaming was already very big:
  - ✔ Several networks
  - ✔ Several operating system
  - ✔ Several codecs
  - ✔ Several browsers
- Flash and H264 consolidate themselves very quickly.
- The downlink of the client was the bottleneck



WIFI

3G

INTERNET

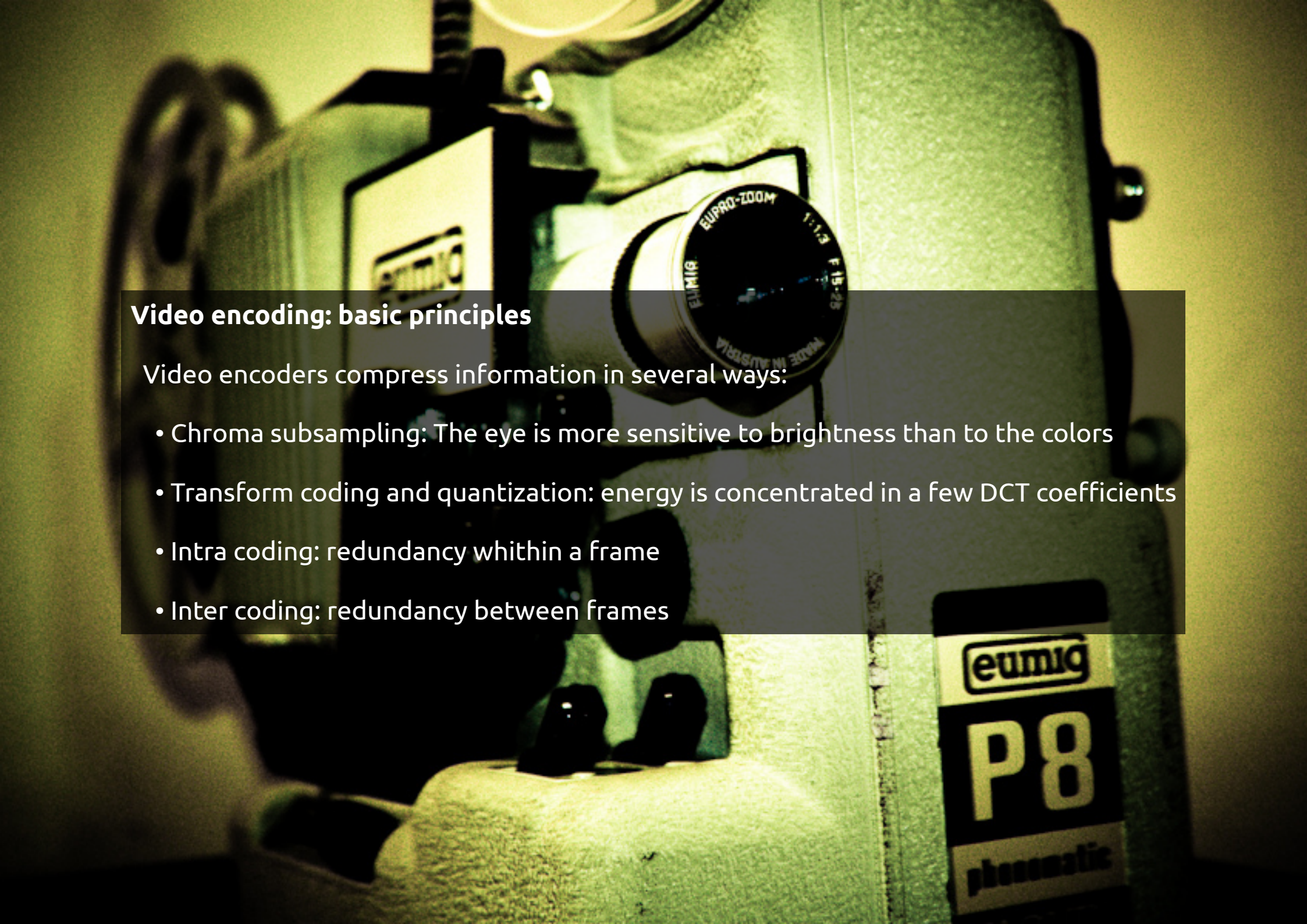
WIFI

FTTH

ADSL

Nowadays the ecosystem is even bigger and more sparse

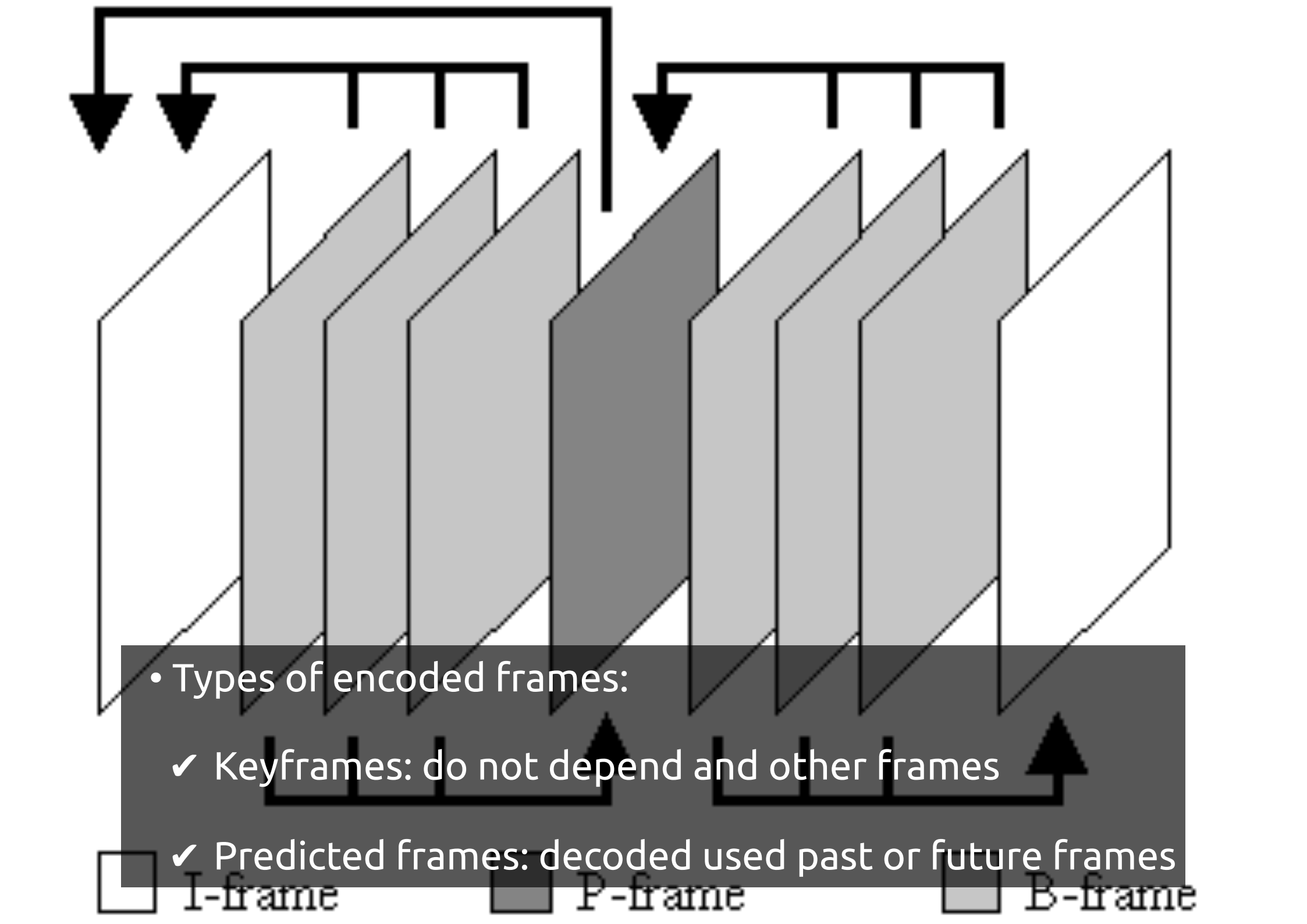
- Different networks require different bitrates
- Different devices:
  - ✔ Need to encode with different picture sizes
  - ✔ Hardware limitations for encoding profiles



Video encoding: basic principles

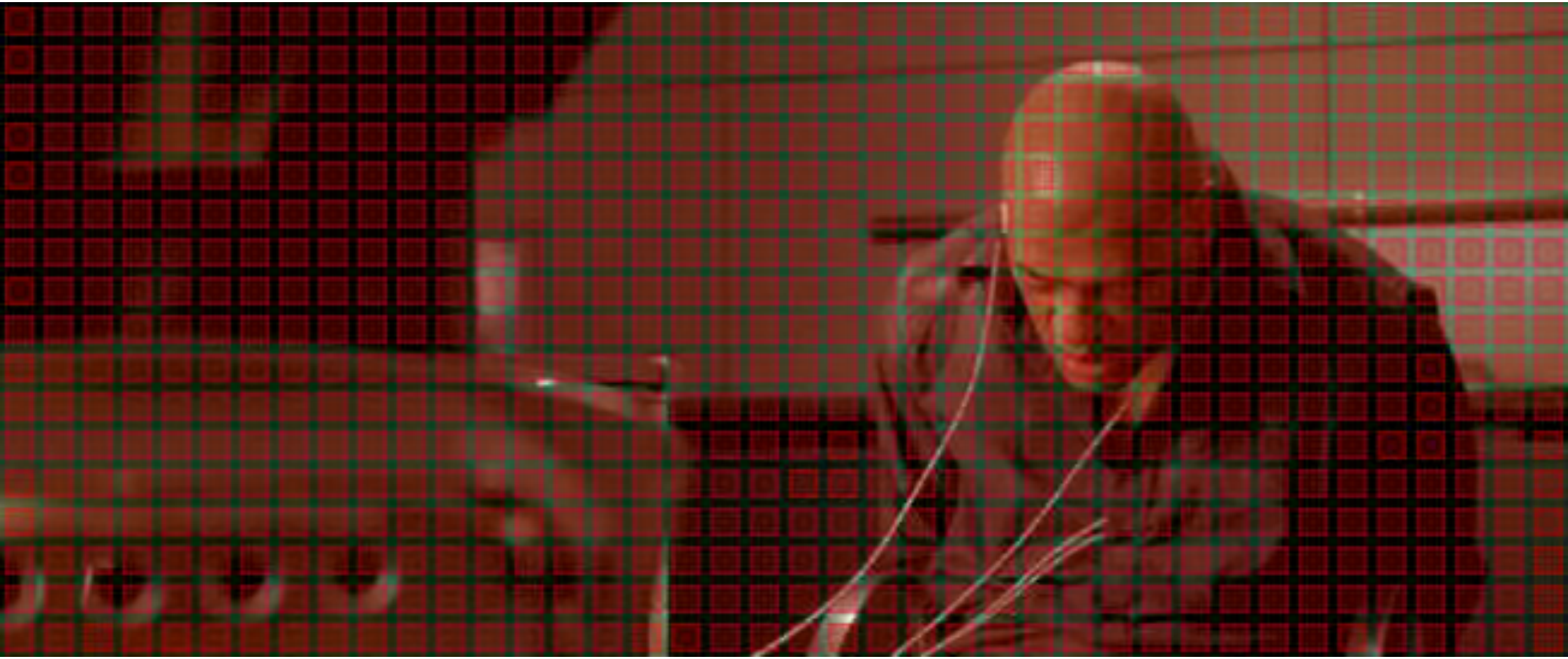
Video encoders compress information in several ways:

- Chroma subsampling: The eye is more sensitive to brightness than to the colors
- Transform coding and quantization: energy is concentrated in a few DCT coefficients
- Intra coding: redundancy whithin a frame
- Inter coding: redundancy between frames



- Types of encoded frames:
  - ✔ Keyframes: do not depend and other frames
  - ✔ Predicted frames: decoded used past or future frames

Keyframes vs predicted frames

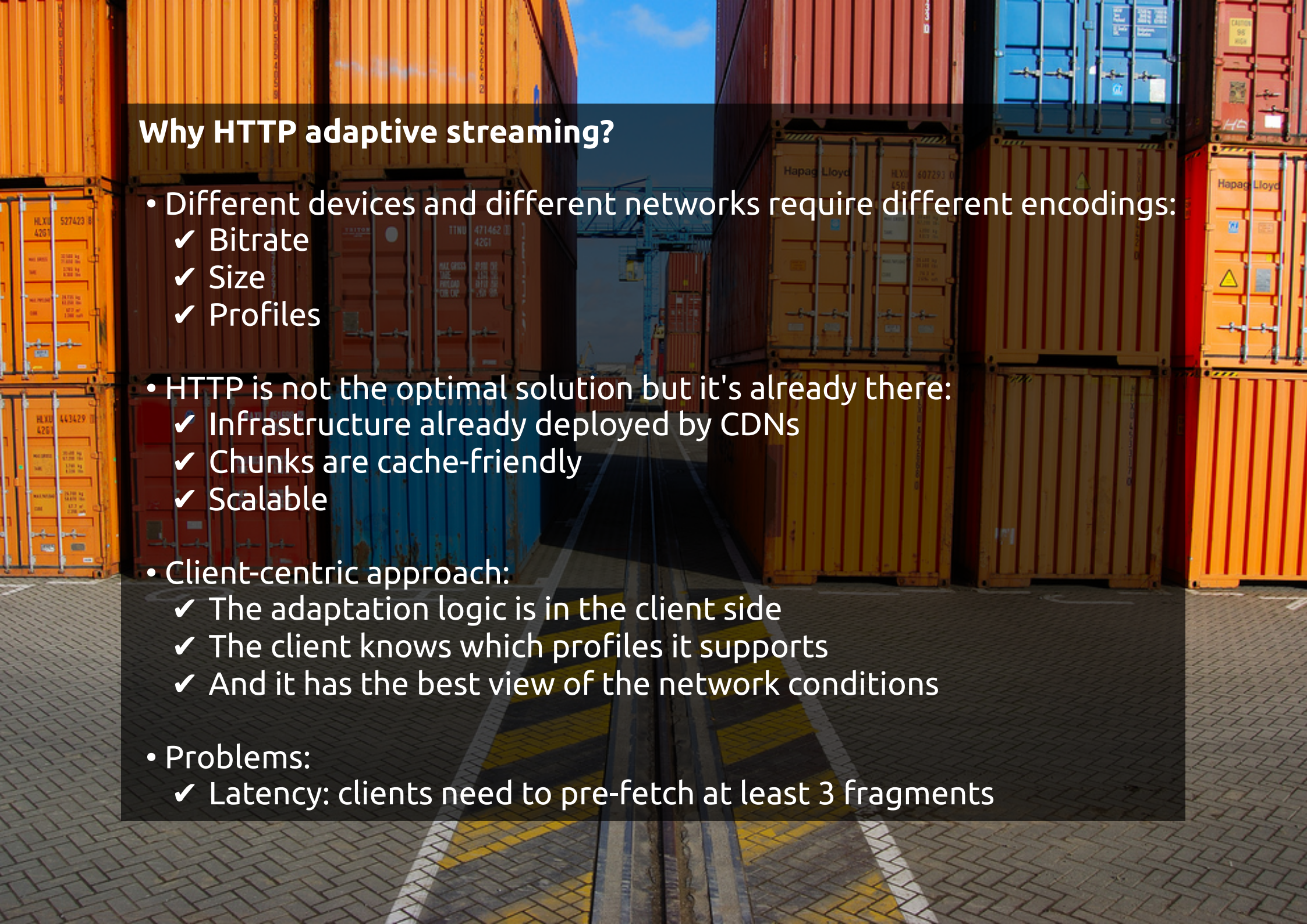


- Keyframes do not depend on past or future frames
- An encoded video stream can be decoded starting from a keyframe

But video needs to be muxed with audio

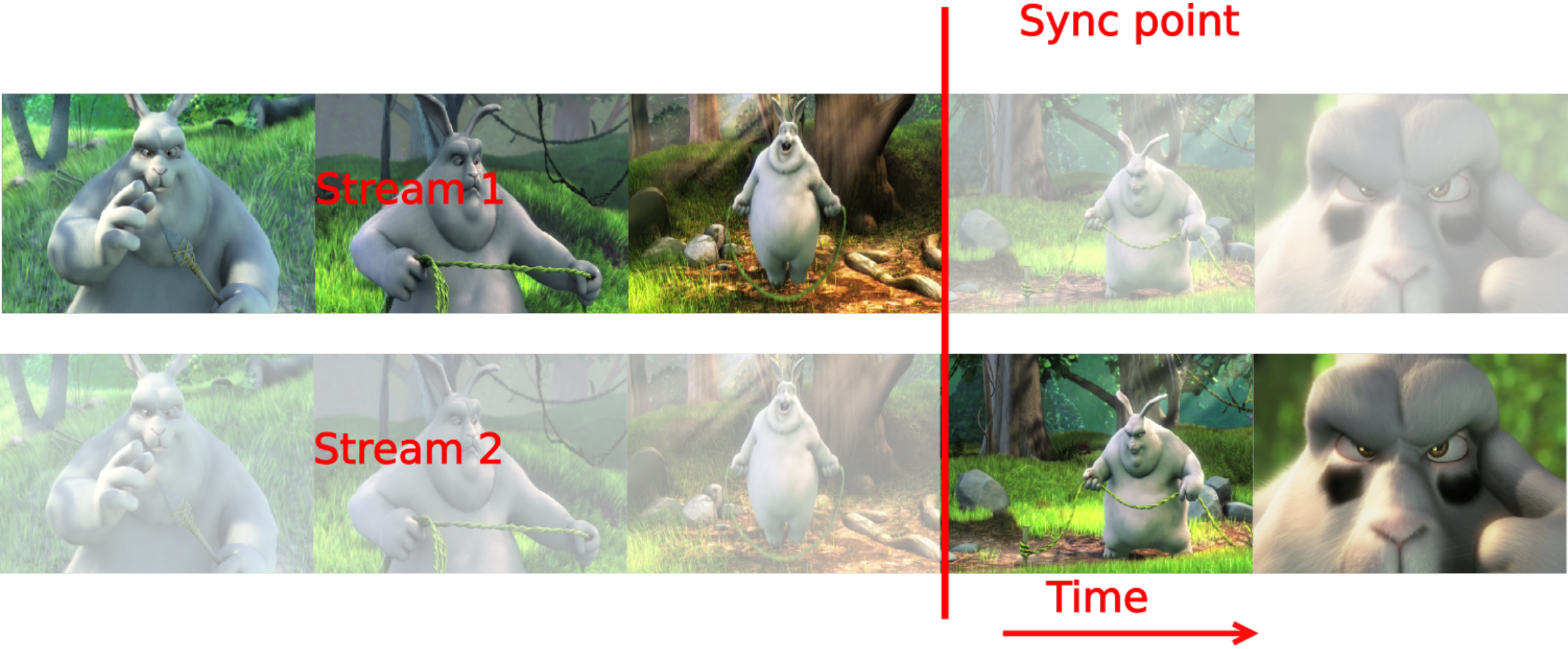
- Pure streamable formats: MPEG-TS, OGG
- Non-streamable formats: AVI, MP4
- Mixed formats: ASF, FLV, WebM, FMP4


Streamable: MPEG-TS
Video
A
Video
A
PAT
PMT
Video
A
Headers
Tracks
Codecs Info
Samples offset
Keyframes Index
VIDEO
AUDIO
Non-streamable: MP4
Headers
Tracks
Codecs Info
Keyframes Index
VIDEO
AUDIO
Both: WebM

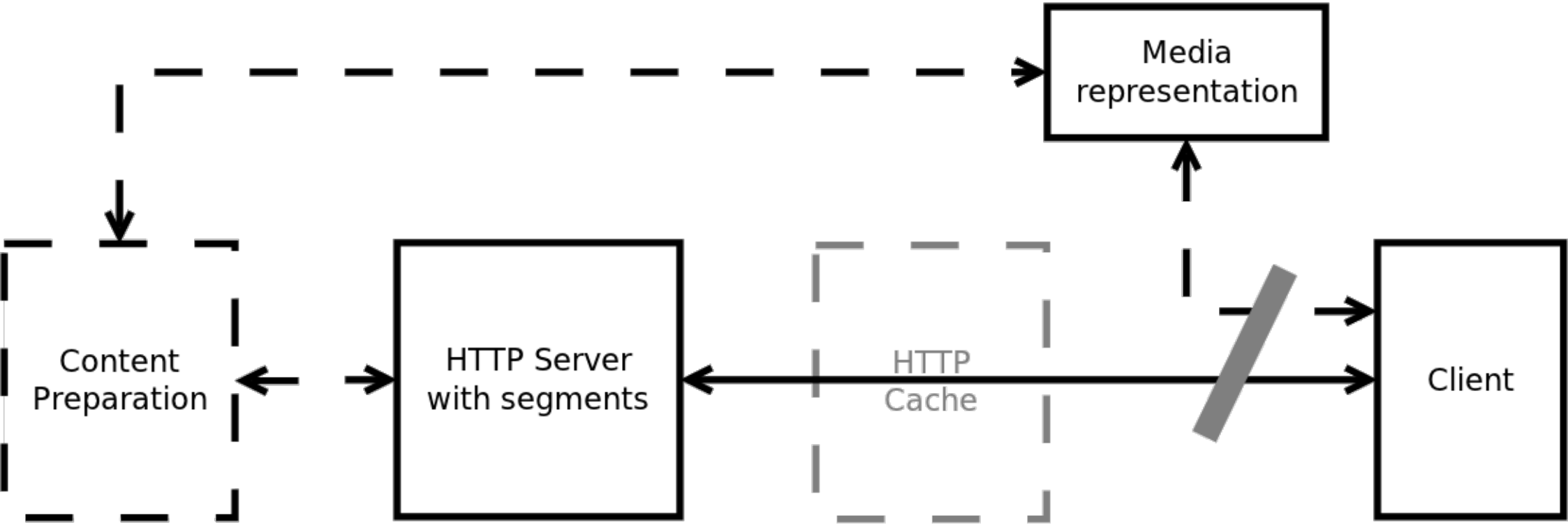


Why HTTP adaptive streaming?

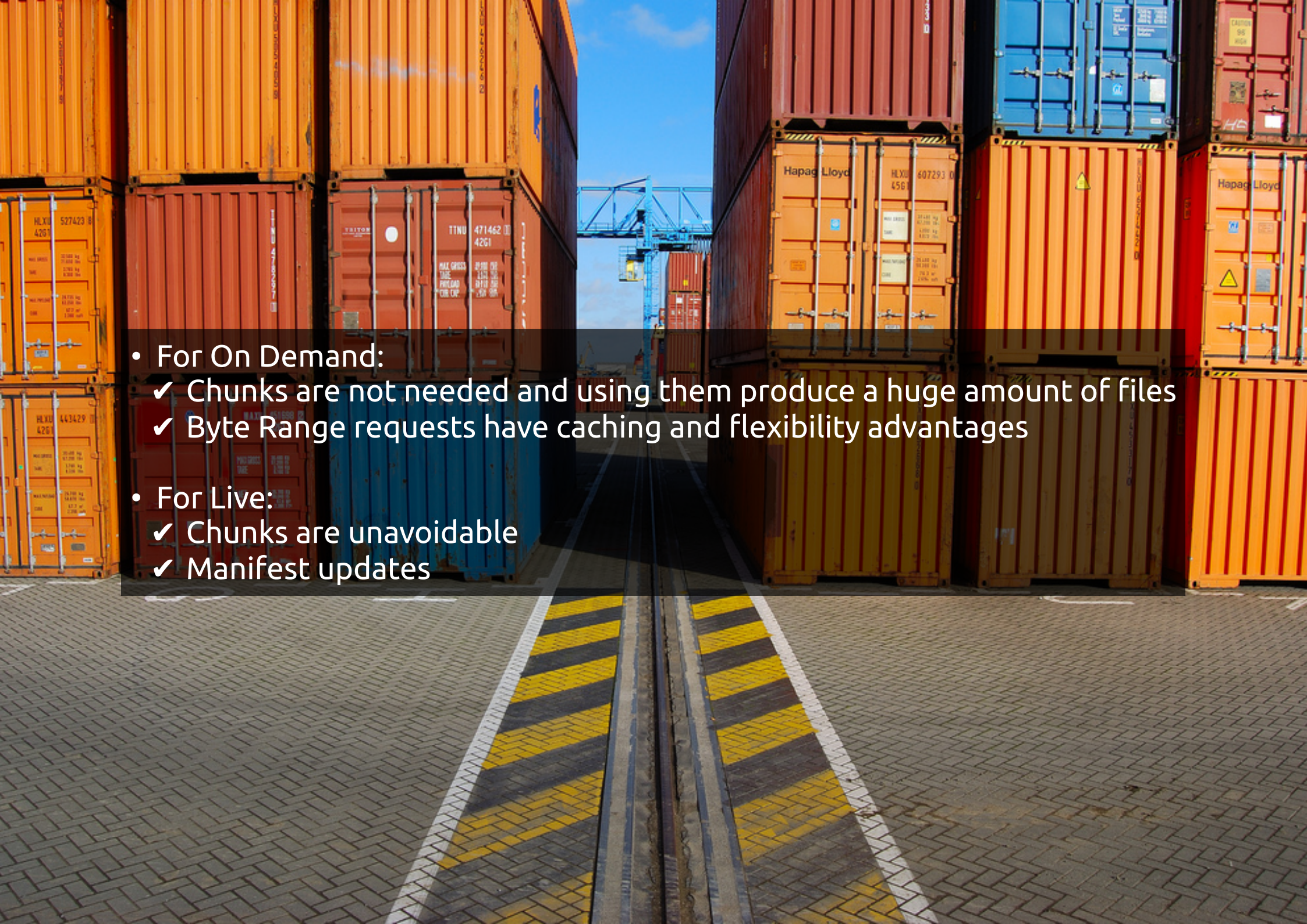
- Different devices and different networks require different encodings:
  - ✔ Bitrate
  - ✔ Size
  - ✔ Profiles
- HTTP is not the optimal solution but it's already there:
  - ✔ Infrastructure already deployed by CDNs
  - ✔ Chunks are cache-friendly
  - ✔ Scalable
- Client-centric approach:
  - ✔ The adaptation logic is in the client side
  - ✔ The client knows which profiles it supports
  - ✔ And it has the best view of the network conditions
- Problems:
  - ✔ Latency: clients need to pre-fetch at least 3 fragments



The stream is chunked in segments of 2~10 seconds using the keyframes boundaries
All stream must be aligned for a smooth transition



Architecture



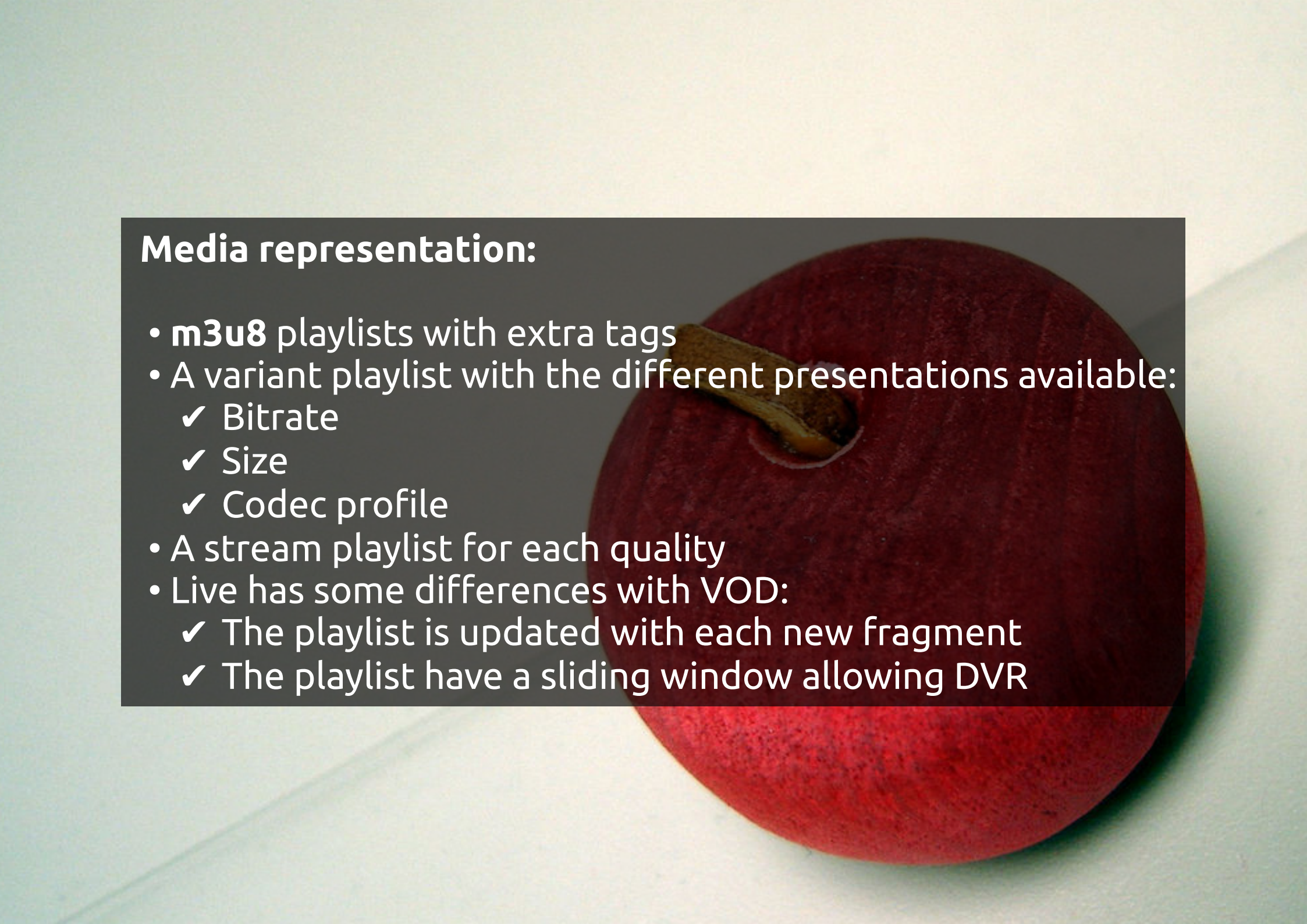
- For On Demand:
  - ✔ Chunks are not needed and using them produce a huge amount of files
  - ✔ Byte Range requests have caching and flexibility advantages
- For Live:
  - ✔ Chunks are unavoidable
  - ✔ Manifest updates

HTTP Live Streaming (HLS) :

- Developed by Apple for QuickTime and IPhone originally:
  - ✔ All IOS devices (IPod, IPhone and IPad)
  - ✔ Android >= 3.0
  - ✔ Desktop players: vlc, all GStreamer players
  - ✔ Browsers: WebKit Gtk browsers using GStreamer's backend

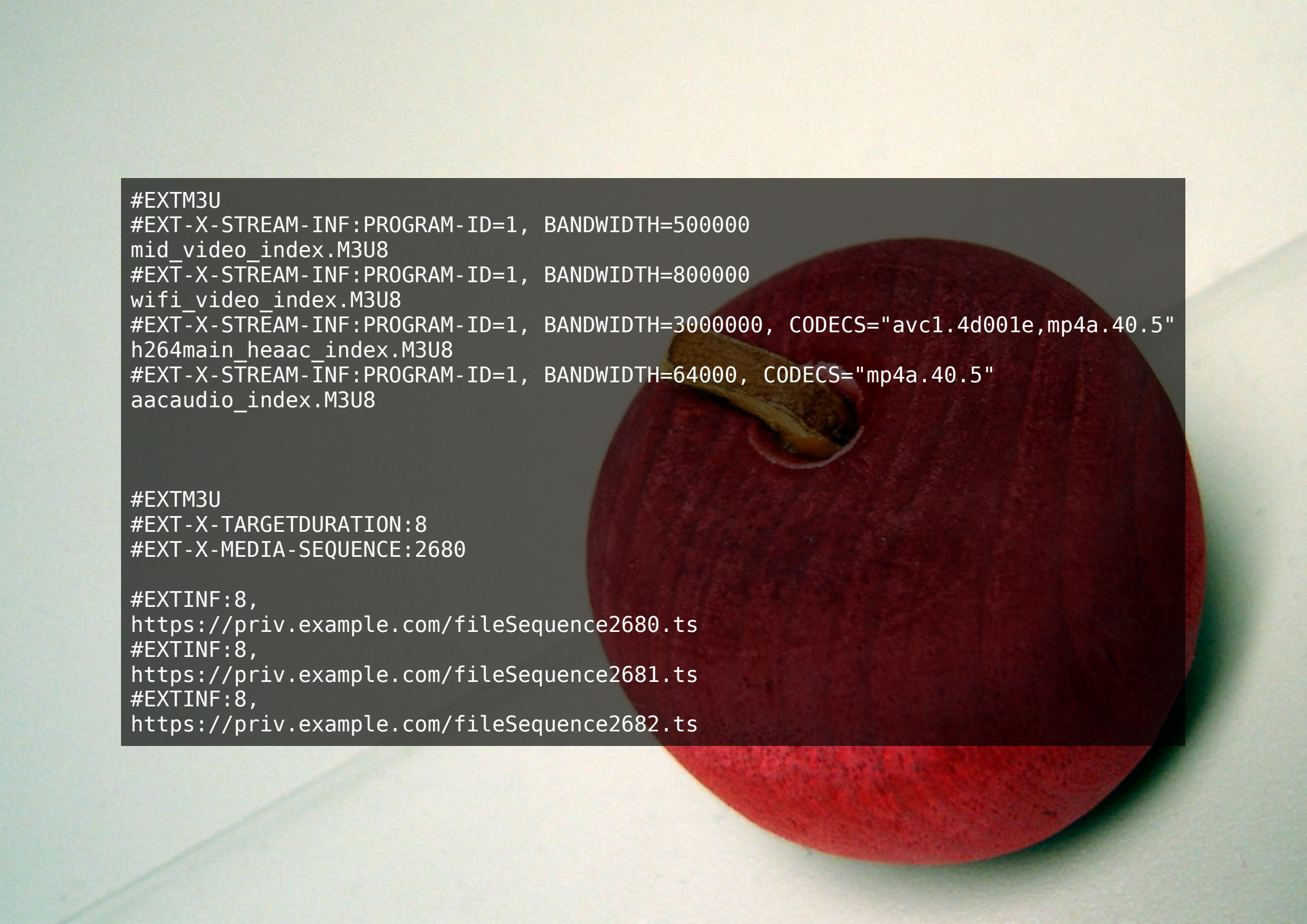
- Documented as an RFC Internet-Draft:
  - ✔ 7 revisions
  - ✔ Version 4 of the protocol

- Format:
  - ✔ MPEG-TS as container
  - ✔ Codecs: AAC/MP3 for audio and H264 for video
  - ✔ Could also be used with Webm/VP8/Vorbis
  - ✔ Fragments of ~10 seconds



Media representation:

- **m3u8** playlists with extra tags
- A variant playlist with the different presentations available:
  - ✔ Bitrate
  - ✔ Size
  - ✔ Codec profile
- A stream playlist for each quality
- Live has some differences with VOD:
  - ✔ The playlist is updated with each new fragment
  - ✔ The playlist have a sliding window allowing DVR

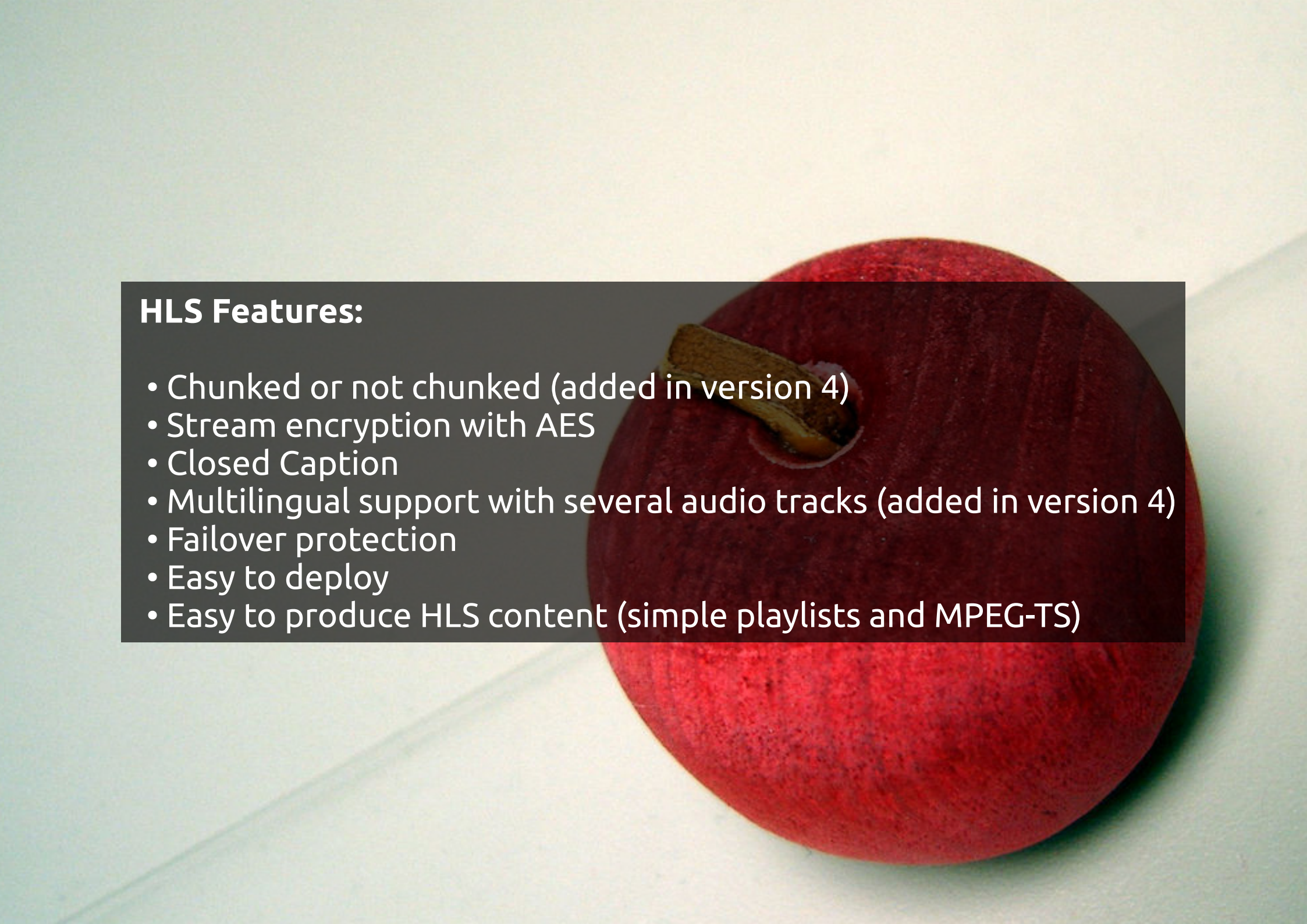


```
#EXTM3U
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=500000
mid_video_index.M3U8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=800000
wifi_video_index.M3U8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=3000000, CODECS="avc1.4d001e,mp4a.40.5"
h264main_heaac_index.M3U8
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=64000, CODECS="mp4a.40.5"
aacaudio_index.M3U8



#EXTM3U
#EXT-X-TARGETDURATION:8
#EXT-X-MEDIA-SEQUENCE:2680

#EXTINF:8,
https://priv.example.com/fileSequence2680.ts
#EXTINF:8,
https://priv.example.com/fileSequence2681.ts
#EXTINF:8,
https://priv.example.com/fileSequence2682.ts
```



HLS Features:

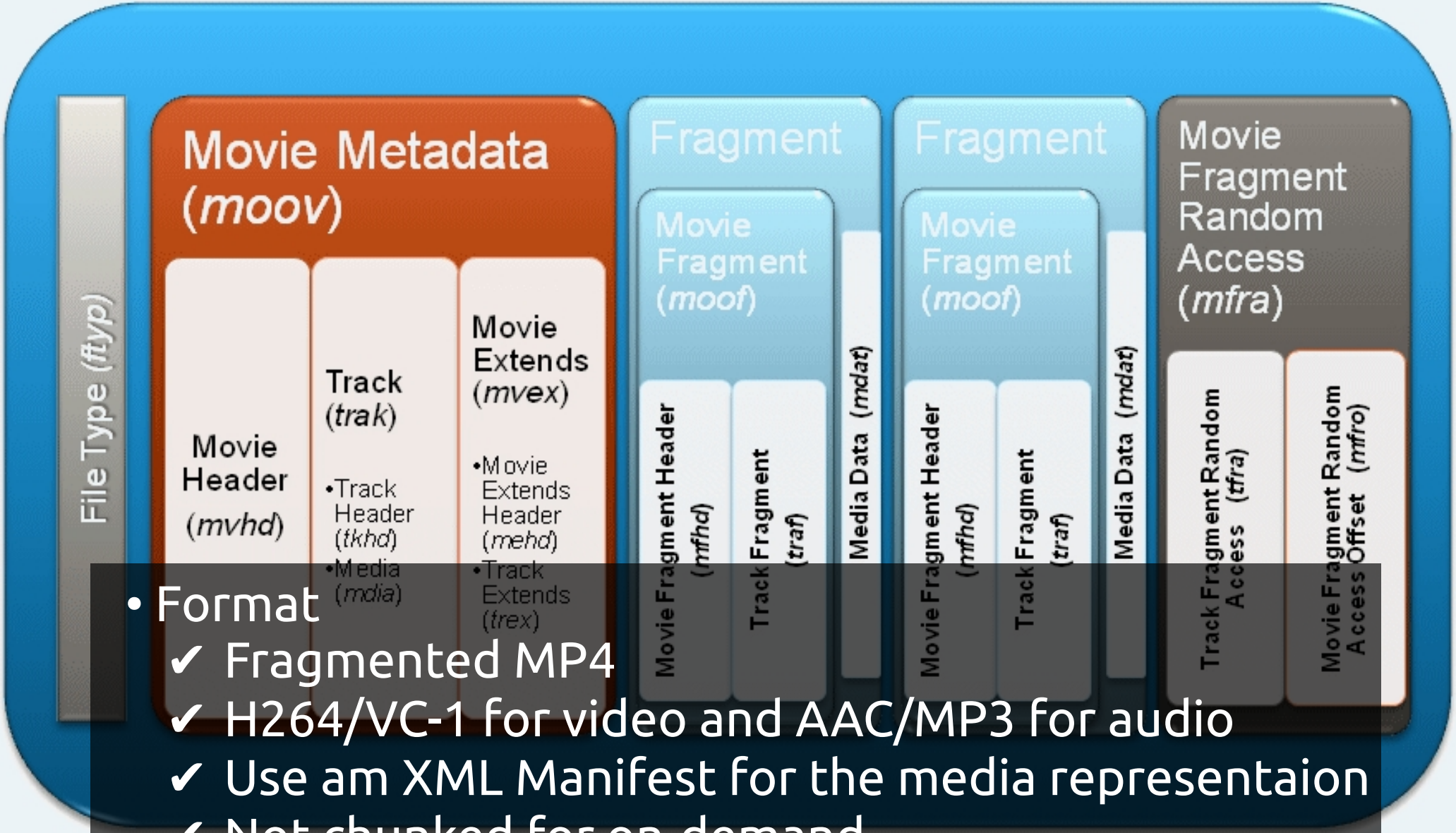
- Chunked or not chunked (added in version 4)
- Stream encryption with AES
- Closed Caption
- Multilingual support with several audio tracks (added in version 4)
- Failover protection
- Easy to deploy
- Easy to produce HLS content (simple playlists and MPEG-TS)


Max
Frame Rate (24) fps
0
Now Downloading Bit Rate
2750 kbps
Limit Max Bit Rate
4930 kbps
Smooth Streaming
4.93M
3.675M
2.75M
2.04M
1.52M
1.13M
845K
630K
470K
350K

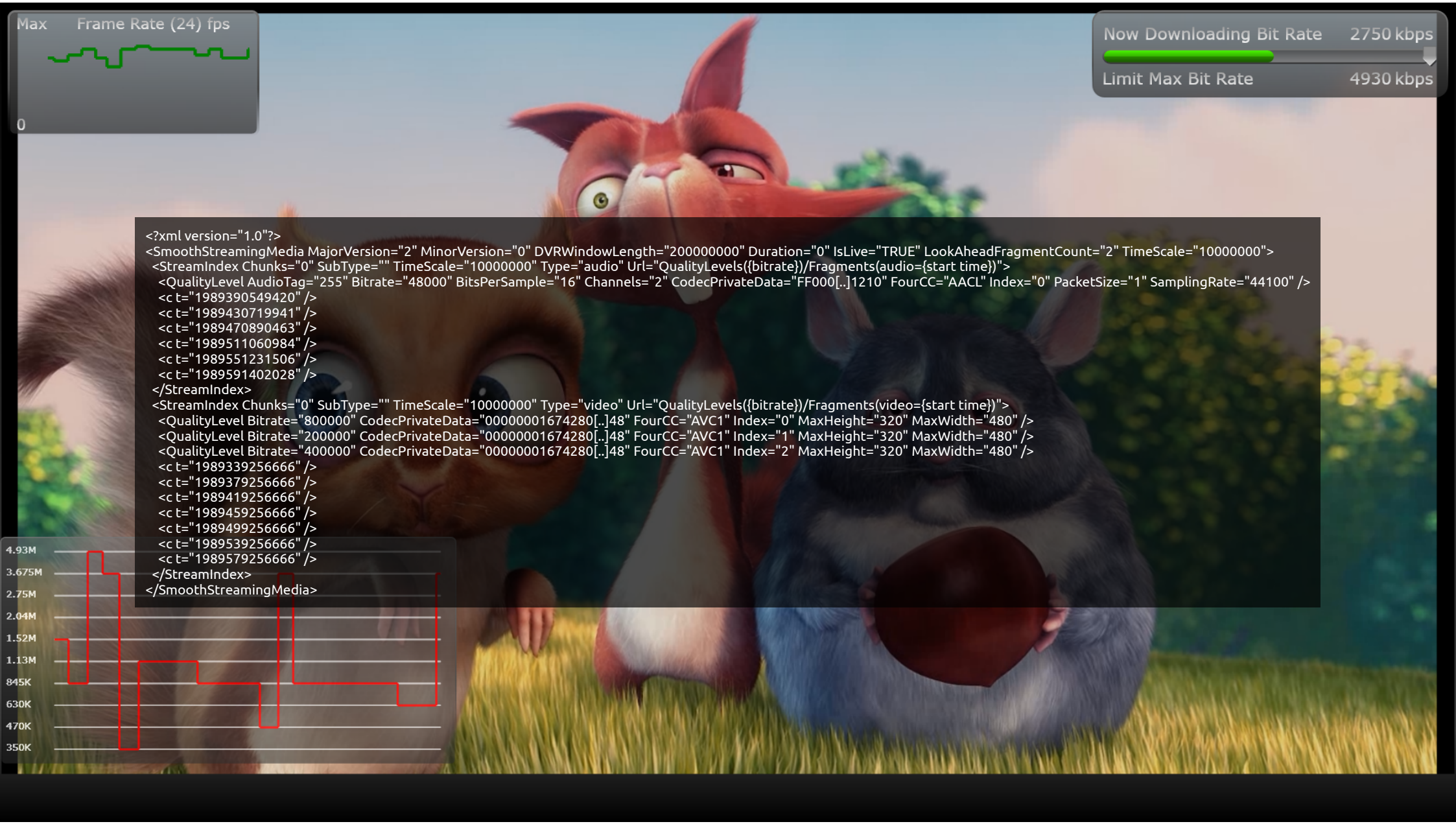
Smooth Streaming:

- Developed by Microsoft:
  - ✔ Microsoft provides a sample player (requires Silverlight)
  - ✔ Custom players can be developed using the open media player framework
  - ✔ Windows phone
  - ✔ Used for the first for on-demand video of the 2008 Summer Olympics

- Standardized as **Protected Interoperable File Format** (PIFF):
  - ✔ Extension of the ISO Base Media File Format (MPEG4 part 12, aka MP4)
  - ✔ Adds DRM support and some restrictions to the original standard
  - ✔ Released under the Microsoft Community Promise



- Format
  - ✔ Fragmented MP4
  - ✔ H264/VC-1 for video and AAC/MP3 for audio
  - ✔ Use am XML Manifest for the media representaion
  - ✔ Not chunked for on-demand
  - ✔ Live is tricky


Max
Frame Rate (24) fps
0
Now Downloading Bit Rate
2750 kbps
Limit Max Bit Rate
4930 kbps
<?xml version="1.0"?>
<SmoothStreamingMedia MajorVersion="2" MinorVersion="0" DVRWindowLength="200000000" Duration="0" IsLive="TRUE" LookAheadFragmentCount="2" TimeScale="10000000">
<StreamIndex Chunks="0" SubType="" TimeScale="10000000" Type="audio" Url="QualityLevels({bitrate})/Fragments(audio={start time})">
<QualityLevel AudioTag="255" Bitrate="48000" BitsPerSample="16" Channels="2" CodecPrivateData="FF000[..]1210" FourCC="AACL" Index="0" PacketSize="1" SamplingRate="44100" />
<c t="1989390549420" />
<c t="1989430719941" />
<c t="1989470890463" />
<c t="1989511060984" />
<c t="1989551231506" />
<c t="1989591402028" />
</StreamIndex>
<StreamIndex Chunks="0" SubType="" TimeScale="10000000" Type="video" Url="QualityLevels({bitrate})/Fragments(video={start time})">
<QualityLevel Bitrate="800000" CodecPrivateData="00000001674280[..]48" FourCC="AVC1" Index="0" MaxHeight="320" MaxWidth="480" />
<QualityLevel Bitrate="200000" CodecPrivateData="00000001674280[..]48" FourCC="AVC1" Index="1" MaxHeight="320" MaxWidth="480" />
<QualityLevel Bitrate="400000" CodecPrivateData="00000001674280[..]48" FourCC="AVC1" Index="2" MaxHeight="320" MaxWidth="480" />
<c t="1989339256666" />
<c t="1989379256666" />
<c t="1989419256666" />
<c t="1989459256666" />
<c t="1989499256666" />
<c t="1989539256666" />
<c t="1989579256666" />
</StreamIndex>
</SmoothStreamingMedia>
4.93M
3.675M
2.75M
2.04M
1.52M
1.13M
845K
630K
470K
350K

Features:

- The manifest is downloaded only once, event for live
- Multilingual audio
- Supports DRM

Cons:

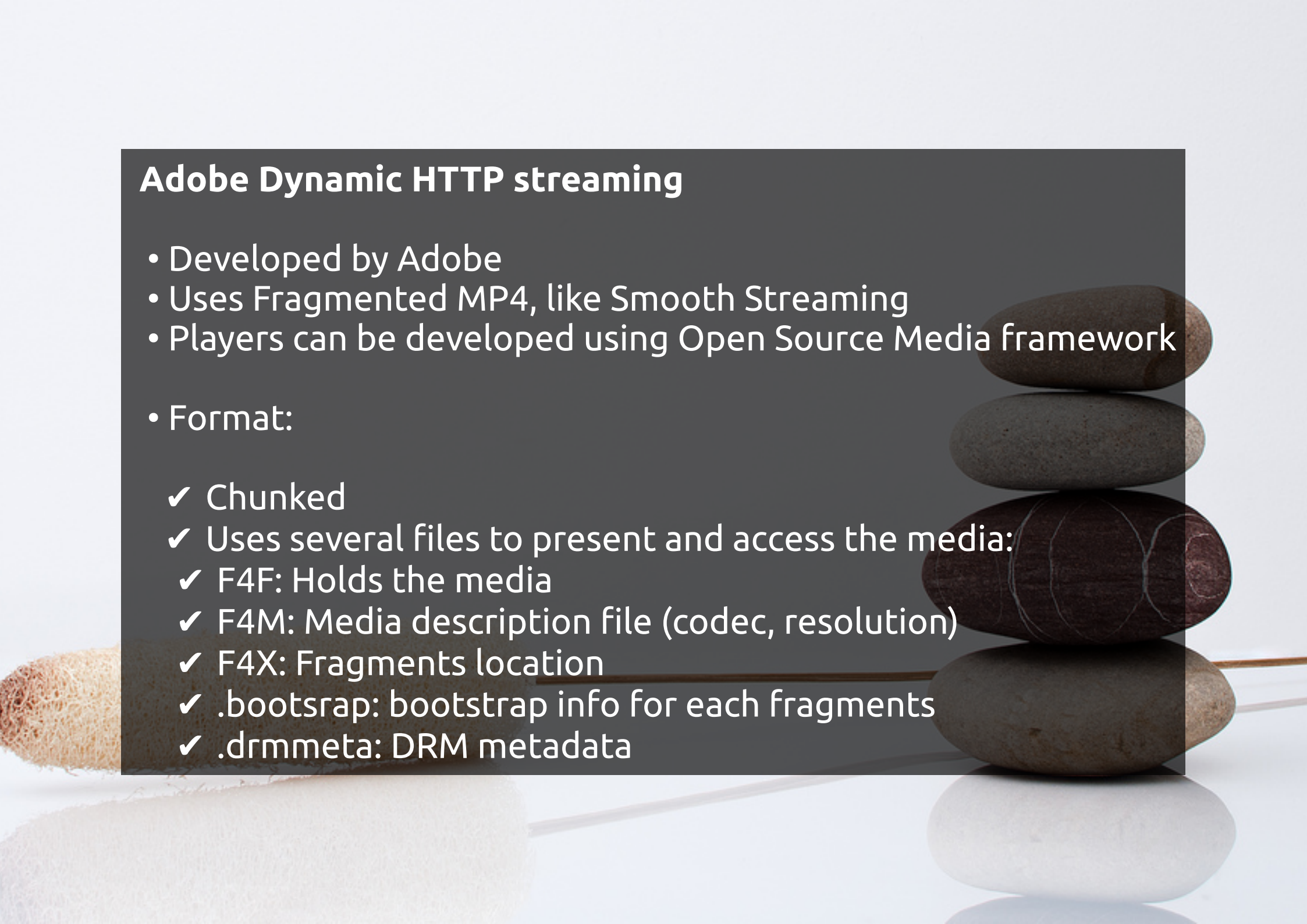
- Hard to produce live content
- Not widely supported (Silverlight and Windows Phone)

Adobe Dynamic HTTP streaming

- Developed by Adobe
- Uses Fragmented MP4, like Smooth Streaming
- Players can be developed using Open Source Media framework

- Format:

  - ✔ Chunked
  - ✔ Uses several files to present and access the media:
    - ✔ F4F: Holds the media
    - ✔ F4M: Media description file (codec, resolution)
    - ✔ F4X: Fragments location
    - ✔ .bootsrap: bootstrap info for each fragments
    - ✔ .drmmeta: DRM metadata



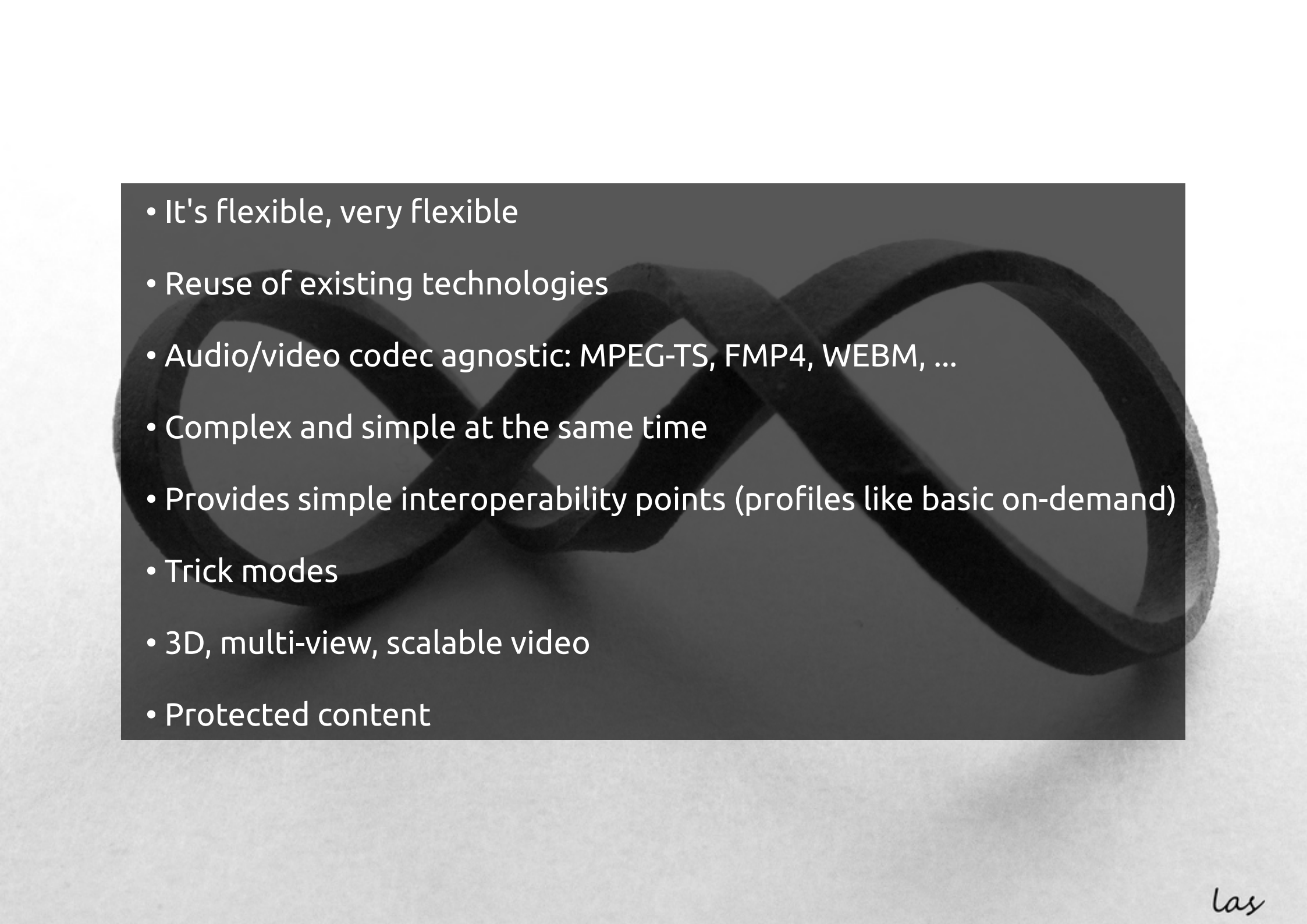

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0">
<id>myvideo</id>
<duration>253</duration>
<mimeType>video/x-flv</mimeType>
<streamType>recorded</streamType>
<baseURL>http://example.com"</baseURL>
<drmAdditionalHeader url="http://mydrmserver.com/mydrmadditionalheader"/>
<bootstrapInfo profile="named" url="/mybootstrapinfo"/>
<media url="/myvideo/low" bitrate="408" width="640" height="480"/>
<media url="/myvideo/medium" bitrate="908" width="800" height="600"/>
<media url="/myvideo/high" bitrate="1708" width="1920" height="1080"/>
</manifest>


Too many propietary solutions for the same thing!
A standard is needed


DASH to the rescue!

DASH: Dynamic Adaptive Streaming over HTTP

- MPEG DASH ISO/IEC 23001-6 is now the master specification
- Draft International Standard in January 2011 (v2)
- Expected to become an International Standard this month!



- It's flexible, very flexible
- Reuse of existing technologies
- Audio/video codec agnostic: MPEG-TS, FMP4, WEBM, ...
- Complex and simple at the same time
- Provides simple interoperability points (profiles like basic on-demand)
- Trick modes
- 3D, multi-view, scalable video
- Protected content

Segment Index in MPD only

```
<MPD>
  ...
  <URL sourceURL="seg1.mp4"/>
  <URL sourceURL="seg2.mp4"/>
</MPD>
```

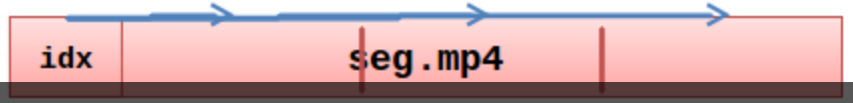

```
<MPD>
  ...
  <URL sourceURL="seg.mp4" range="0-499"/>
  <URL sourceURL="seg.mp4" range="500-999"/>
</MPD>
```

Segment Index in MPD + Segment

```
<MPD>
  ...
  <Index sourceURL="idx.mp4"/>
  <URL sourceURL="seg.mp4"/>
</MPD>
```

Segment Index in Segment only

```
<MPD>
  ...
  <BaseURL>seg.mp4</BaseURL>
</MPD>
```



The Media Presentation Description (MPD): contains metadata required by a DASH Client to construct appropriate HTTP-URLs and access Segments and to provide the streaming service to the user.

I've seen the FUTURE It's in the BROWSER

Media Source API:

- Support adaptive streaming natively in HTML5
- An extension to HTMLMediaElement that allows to pipe data into audio and video
- MD parsing, buffering, bitrate switching strategies should not be done by the browser
- Demo available for Chrome presented in the OVC12

```
interface HTMLMediaElement : HTMLElement {
  ...
  // URL passed to src attribute to enable the media source logic.
  readonly attribute DOMString webkitMediaSourceURL;

  // Appends media to to the source.
  void webkitSourceAppend(in Uint8Array data) raises (DOMException);
}
```


Questions?

Andoni Morales Alastruey <ylatuya@gmail.com>
Barcelona Developers Conference 2011
Special thanks to Xavi and all the people that made this conference possible :)

Acknowledgment: Mark Watson, Christian Timmerer and Christopher Müller
from which I based the DASH part of this presentation

https://github.com/ylatuya/Talks

HLS:
http://developer.apple.com/library/ios/#documentation/networkinginternet/conceptual/streamingmediaguide/Introduction/Introduction.html
http://tools.ietf.org/html/draft-pantos-http-live-streaming-07
http://www.thekuroko.com/http-dynamic-streaming-getting-started/

Smooth streaming:
http://alexzambelli.com/blog/2009/02/10/smooth-streaming-architecture/
http://go.microsoft.com/?linkid=9682897

Dash:
http://openetherpad.org/ovc11-standards-for-http-adaptive-streaming
http://www.slideshare.net/christian.timmerer/dynamic-adaptive-streaming-over-http-dash

Images:
http://www.flickr.com/photos/platinum/8499300/
http://www.flickr.com/photos/bibigeek/6198549521/
http://www.flickr.com/photos/21561428@N03/4991300232/
http://www.flickr.com/photos/elrentaplats/2006927827/
http://www.flickr.com/photos/joost-ijmuiden/4757944724/
http://www.flickr.com/photos/kevincochran/4024384525/
http://www.flickr.com/photos/davidlafourcade/5716913940/
http://www.flickr.com/photos/josefstuefer/5982121/
http://www.erg.abdn.ac.uk/future-net/digital-video/images/mpeg-ts.gif
http://img.tomshardware.com/us/1999/09/24/video_guide_part_3/ipb_frames.gif