Berlin 10/10/2016

Efficient Trick Modes in MPEG-DASH Adaptive Streaming with GStreamer

Visla Systems Ltd
Wojciech Przybyl

- New to gst community

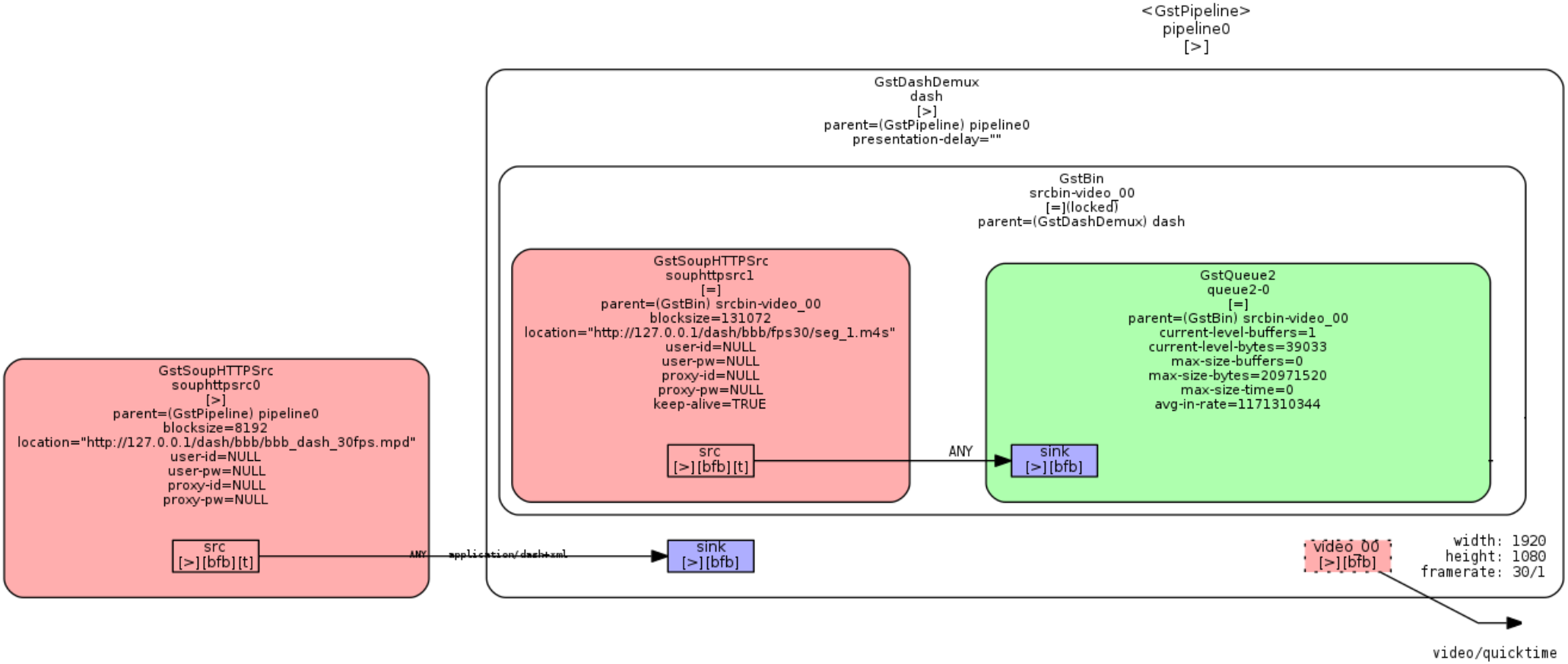
- 10+ years experience in Linux & Media products development

- SetTopBox & DVR development

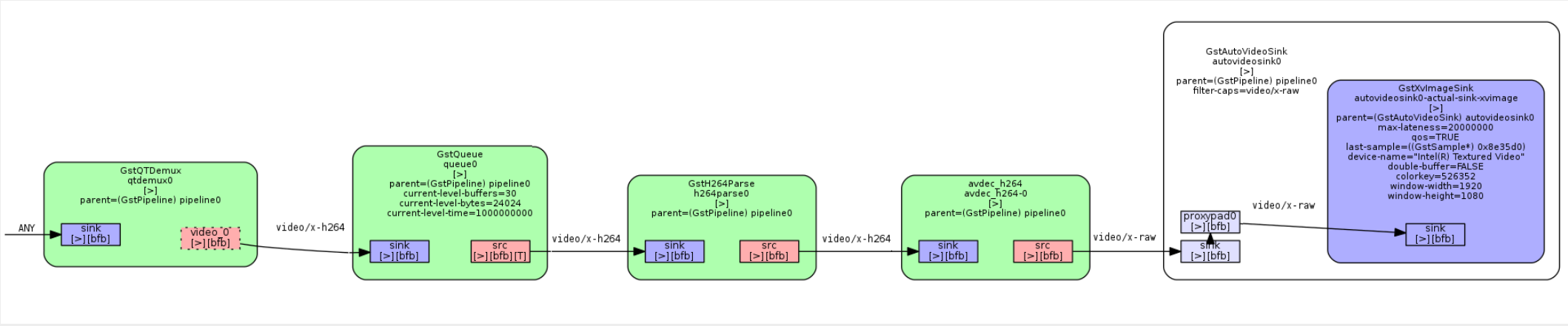
- SoC: ST / Broadcom / TI / Freescale

MPEG-DASH - Dynamic Adaptive Streaming over HTTP

- Codec agnostic: H.264, H.265, VP8, VP9, PCM, AAC, AC-3, DTS etc.
- Containers: MPEG-2 TS, ISOBMFF (MP4)
- Independent downloadable fragments 1-5 sec long
- Adaptive bitrate & framerate
- Streams download over HTTP 1.1



Simple MPEG-DASH pipeline in GStreamer



gst-launch-1.0 souphttpsrc ! dashdemux ! qtdemux ! queue ! h264parse ! avdec_h264 ! autovideosink

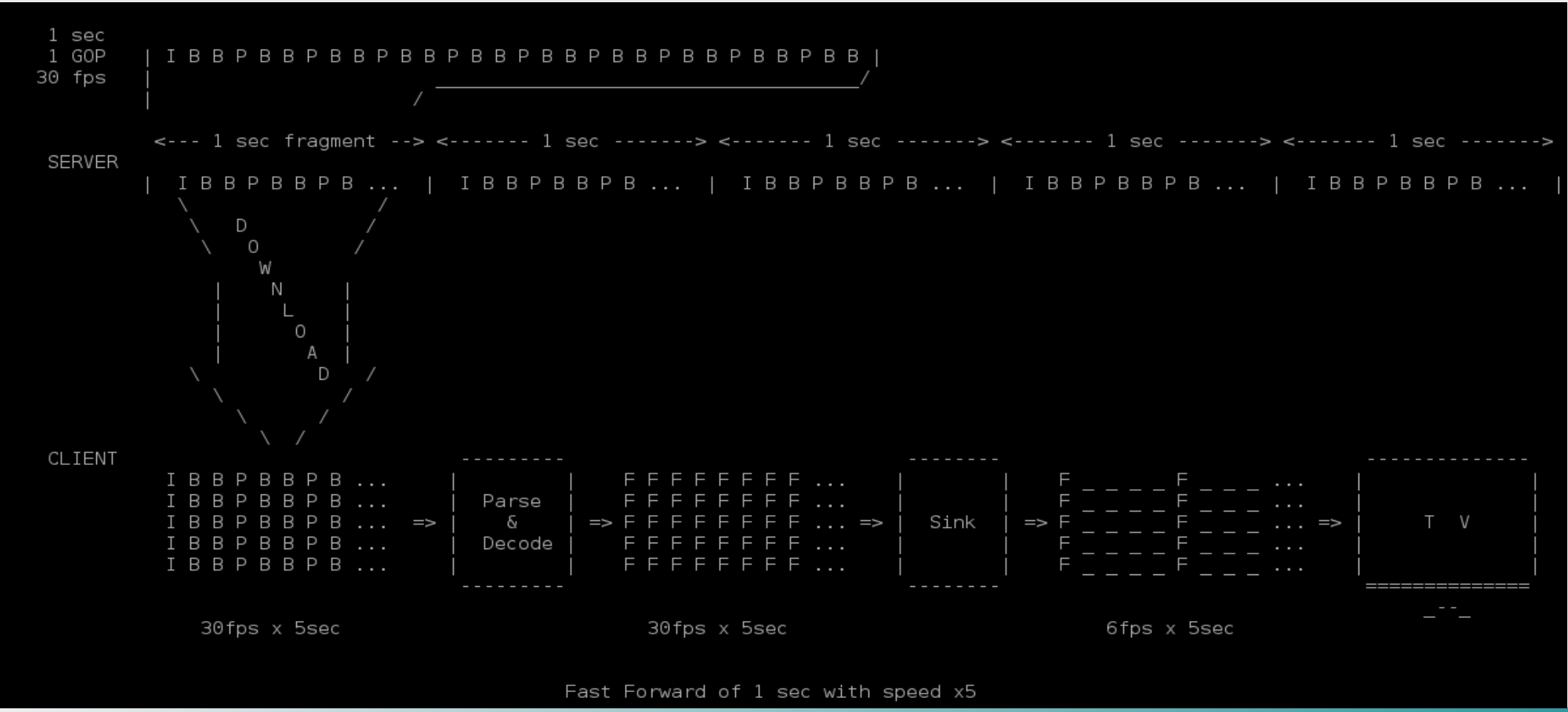
- Trick Modes: Speed and Direction != x1

- Gstreamer API: gst_event_new_seek()

- Judging Trick Modes
  - efficiency
  - user experience
  - complexity
  - cost

Simple Trick Modes

- Play it all but faster
- Download all
- Decode all
- Discard frames in a sink element


1 sec
1 GOP
30 fps
<--- 1 sec fragment --> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec ------->
SERVER
DOWNLOAD
CLIENT
Parse & Decode
Sink
T V
30fps x 5sec
30fps x 5sec
6fps x 5sec
Fast Forward of 1 sec with speed x5

- Local playback with low speed is OK

- What about the bandwidth?

- What about higher speeds: x30?

- Efficiency: None / Impossible

- User Experience: Perfect until it hits the limit

- Complexity: Simple

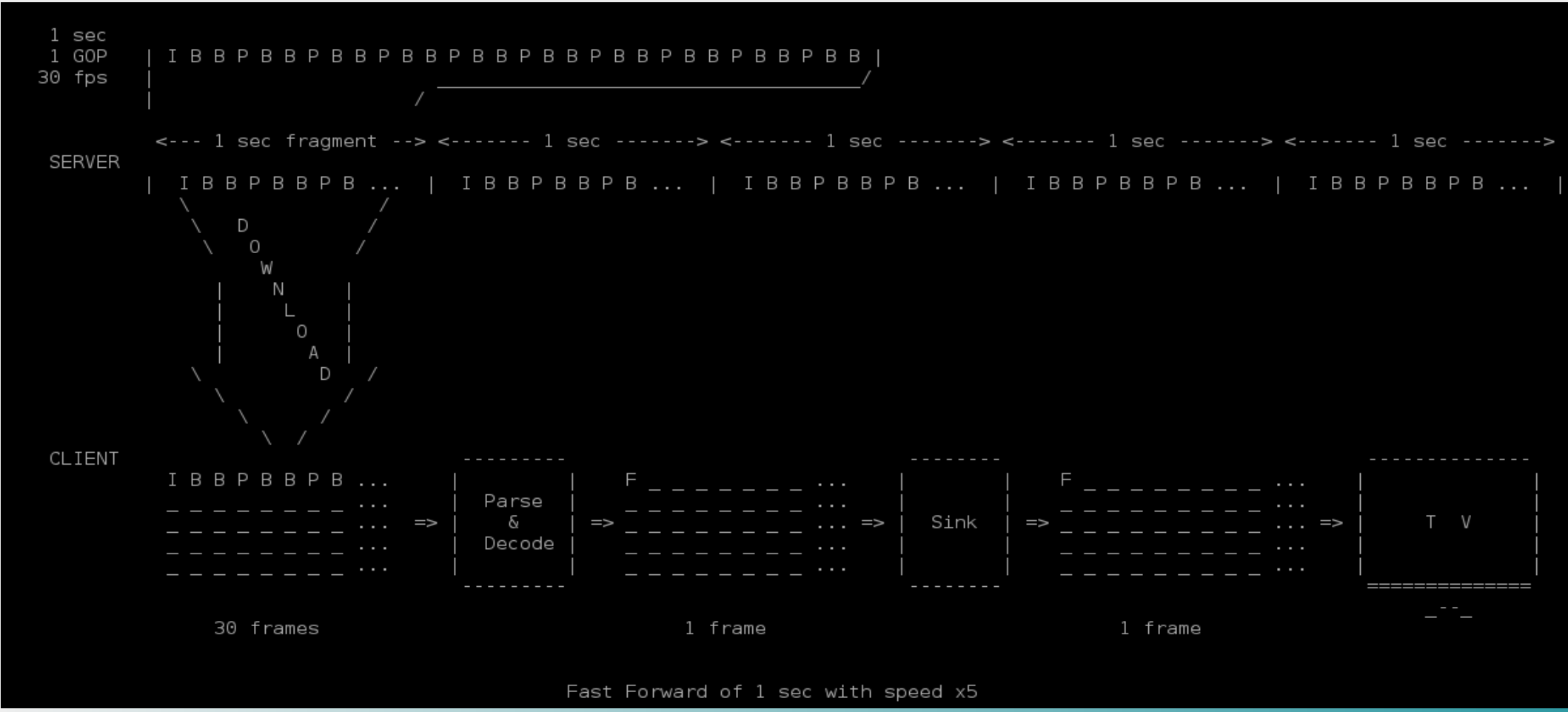
- Cost: Low

Pause & Seek Trick Modes

Pause Playback →

- Seek&Flush
- Download
- Decode
- Present

← Repeat


1 sec
1 GOP
30 fps
I B B P B B P B B P B B P B B P B B P B B P B B P B B P B B
<--- 1 sec fragment --> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec ------->
SERVER
I B B P B B P B ... | I B B P B B P B ... | I B B P B B P B ... | I B B P B B P B ... | I B B P B B P B ... |
DOWNLOAD
CLIENT
I B B P B B P B ...
Parse & Decode
F _ _ _ _ _ _ _ ...
Sink
T V
30 frames
1 frame
1 frame
Fast Forward of 1 sec with speed x5

- Efficiency: Moderate

- User Experience: Moderate

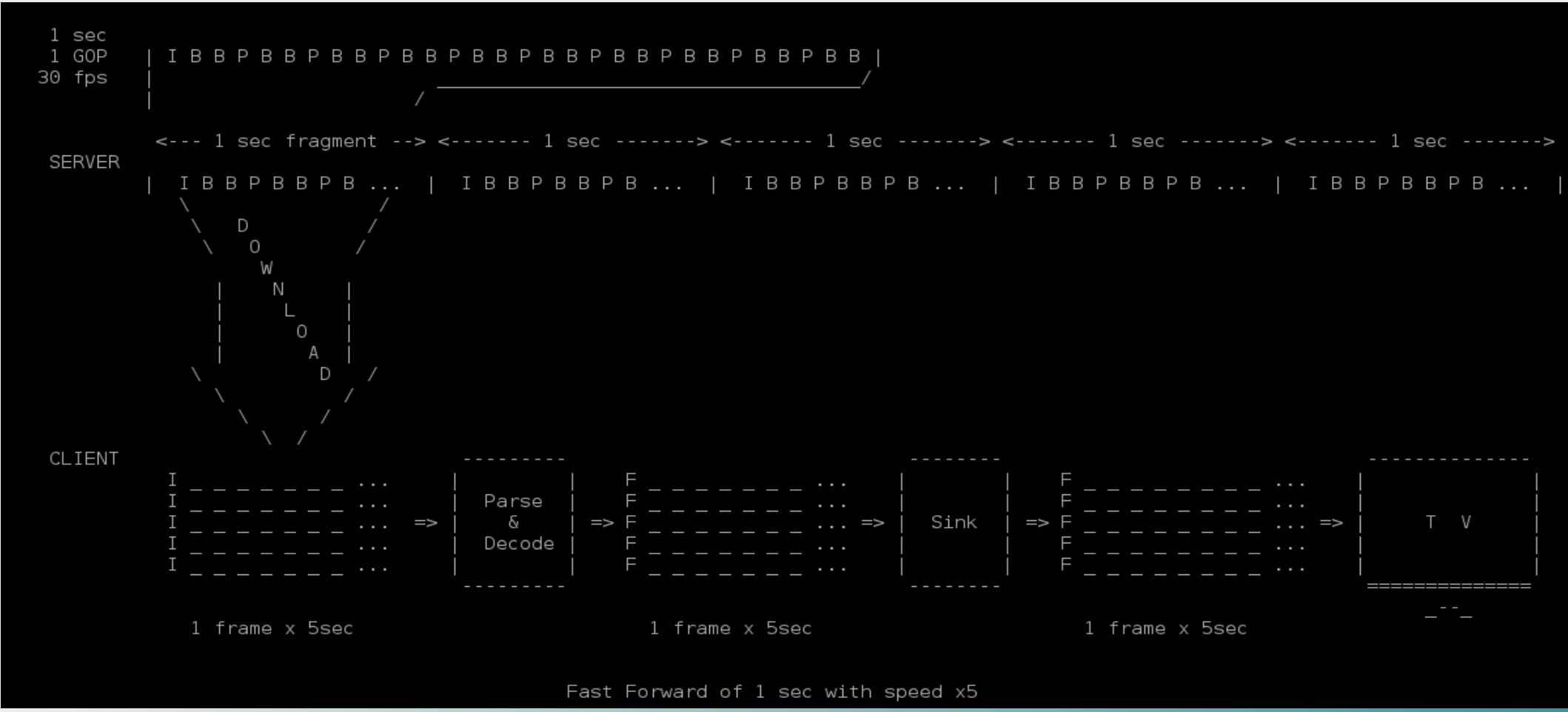
- Complexity: Moderate

- Cost: Low

Key frames only Trick Modes

- Download key frames only

- Perform the rest as in Simple Tick Mode


1 sec
1 GOP
30 fps
<--- 1 sec fragment --> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec ------->
SERVER
DOWNLOAD
CLIENT
Parse & Decode
Sink
T V
1 frame x 5sec
1 frame x 5sec
1 frame x 5sec
Fast Forward of 1 sec with speed x5

- Every DASH fragment starts with a key frame

- Add sidx&ssix box and use HTTP GET RANGE request

- Should we add more I frames when reencoding?

- qtdemux is behind dashdemux

- Efficiency: High

- User Experience: Moderate / Poor

- Complexity: High

- Cost: Low

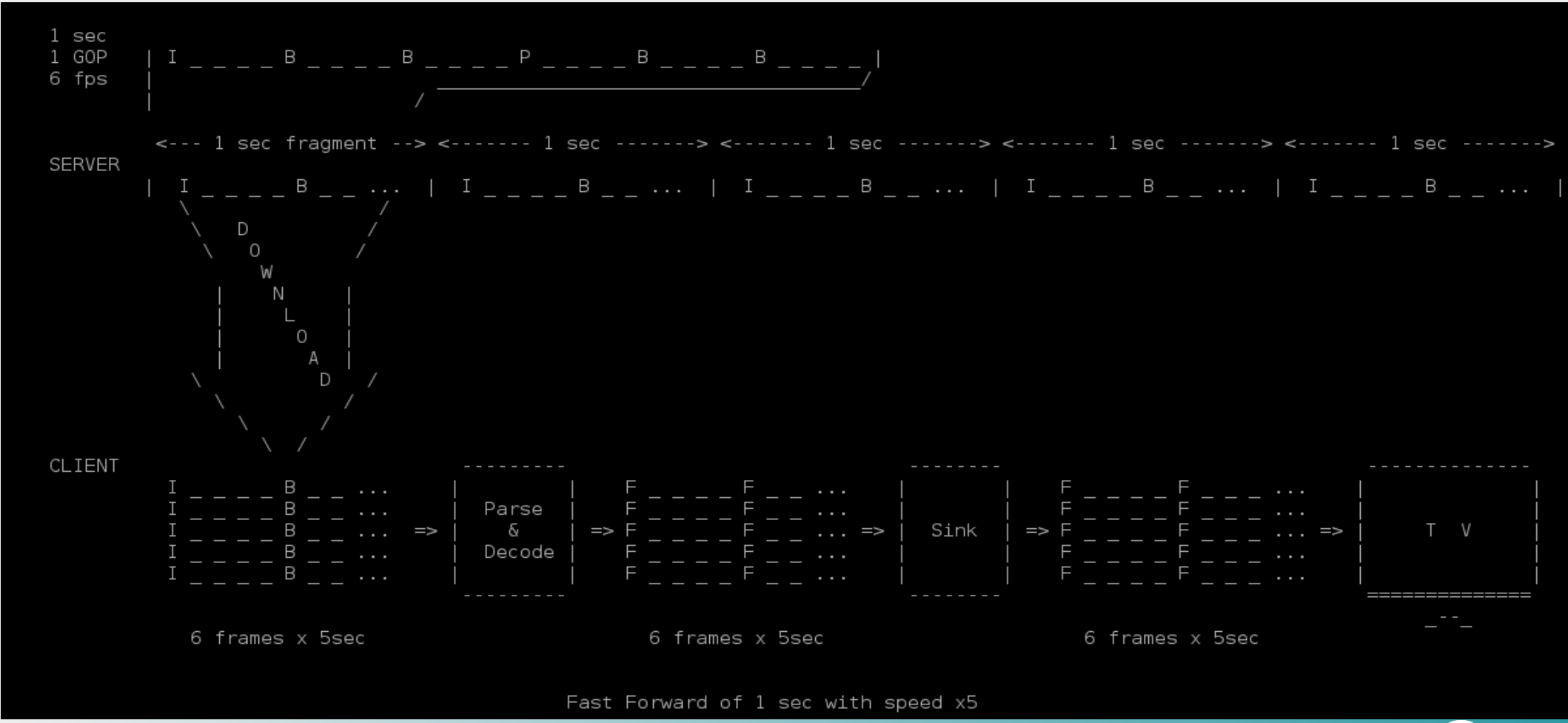
Efficient Trick Modes

- Download separate (Sub)Representation per speed

  holding only&all the frames which are going to be presented

- Perform the rest as in Simple Trick Mode

- Frames discarding is done during stream encoding


1 sec
1 GOP
6 fps
<--- 1 sec fragment --> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec -------> <------- 1 sec ------->
SERVER
CLIENT
Parse & Decode
Sink
T V
6 frames x 5sec
6 frames x 5sec
6 frames x 5sec
Fast Forward of 1 sec with speed x5

- Bandwidth & Processing Power for speed x5 is the same as at speed x1

- Image quality stays (almost) the same

- Bitrate per frame at speed x5 is the same as at speed x1

- Image quality stays (almost) the same
- Why almost?
    - True for speed x1 vs x5
    - Almost true for speed x1 vs x30 → Stream is I frames only

Status

- Proof of Concept stage done

- Choosing correct (Sub)Representation might be tricky

- Manifest's attributes:
  - @bandwidth
  - @frameRate
  - @MaxPlayoutRate

- Efficiency: Perfect

- User Experience: Perfect

- Complexity: Moderate

- Cost: High

Cost = (Storage+Encoding+Packaging+Encryption) * N bitrates * M fps substreams
+Delivery

DEMO

Questions and (hopefully) Answers

Thank you

Wojciech Przybyl
wojciech@visla.co.uk