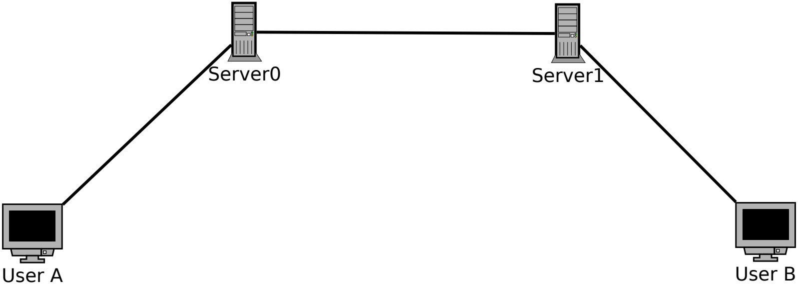
Why Video calls on a mobile device don't just work

Sjoerd Simons

Collabora Ltd.

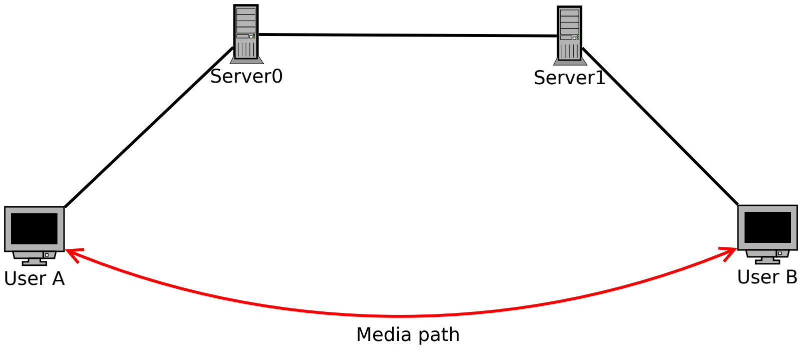
Linux Plumbers 2011

Signalling path



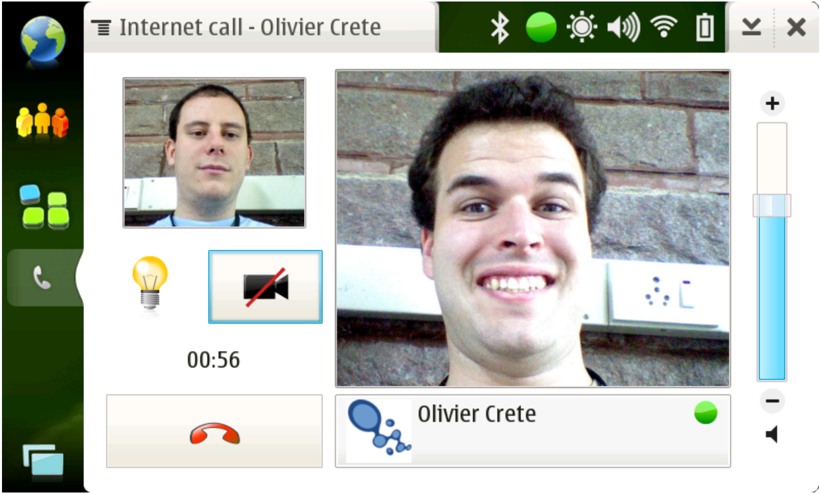
- Must be reliably
- May have high latency
- May be low bandwidth.

Add Media path

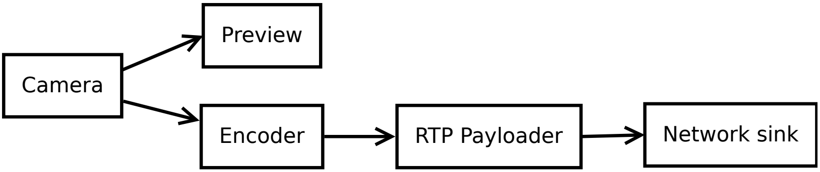


- May be unreliably
- Must have low latency
- Must have good bandwidth

What does it look like



Sending side

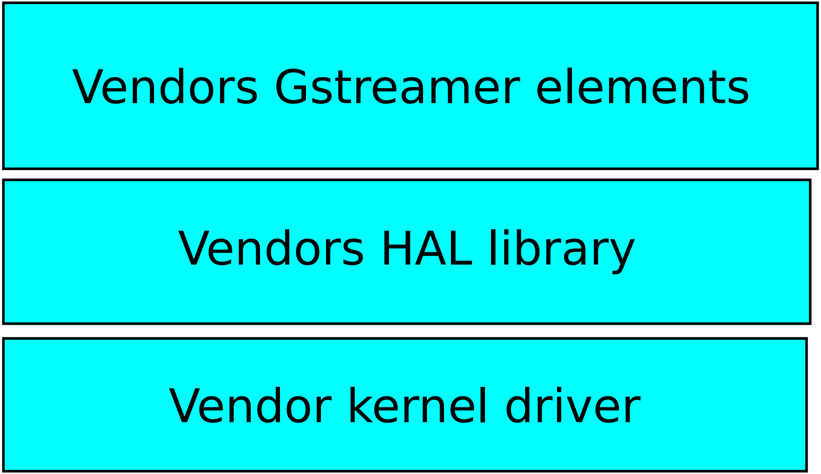


Receiving side

Seems simple enough right..

- Decoders are mostly/only tested for playback use-cases
- Encoders are mostly/only tested for camera/encoding-to-file use-cases
- Gstreamer elements are mostly/only tested in specific pipelines
- Requirements for VoIP are very different

Vendor specific elements



The Obvious differences

- Needs to Encode and Decode video at the same time
- Encoder and Decoders needs to be able to support profiles suitable for VoIP (e.g. baseline profile H.264)
- Decoder needs to cope with lost frames/corrupted frames/etc.

Types of video frames

- Keyframes contains a complete images
  - Also known as I-frame or Intra-coded frame.
- Delta frames contain only the difference from previous frame
  - Also known as P-frames or predicted frames.

and as a result..

- Keyframes are either bigger or lower quality.
  - But we want constant bitrate streams, so low-quality keyframes and improve image quality over time.
- Can't decode delta frames if the keyframe is missing.

Referencing headaches

- Encoder should to be able to disable automatic keyframe generation
- Encoder needs to generate a new keyframe on request.

Referencing headaches

- Decoder need to cope with missing (key)frames.
- Decoder should indicate when (key)frames are missing.
  - Then we can send slice loss indication (or similar) to the sender

Latency is the enemy

- Maximum latency to allow synchronization for live music: 25ms.
  - This is equivalent to sitting about 8 meters apart..
- Usable latency for calls: 200-300ms

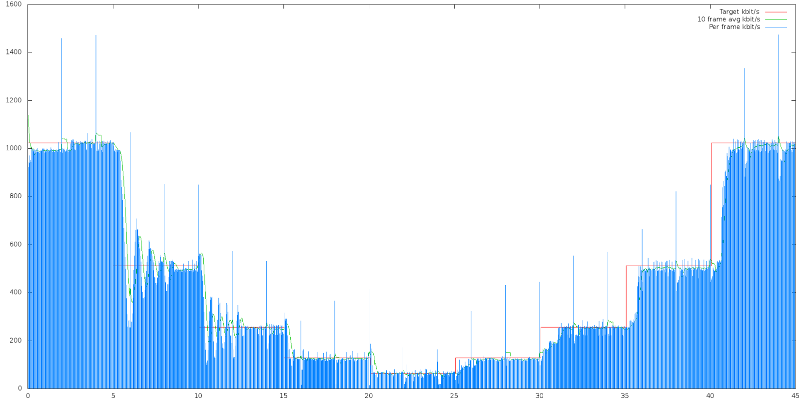
latency is the enemy

- Decoder should add no/minimal latency..
- Encoder should add minimal latency
  - Often needs to be configured to do so

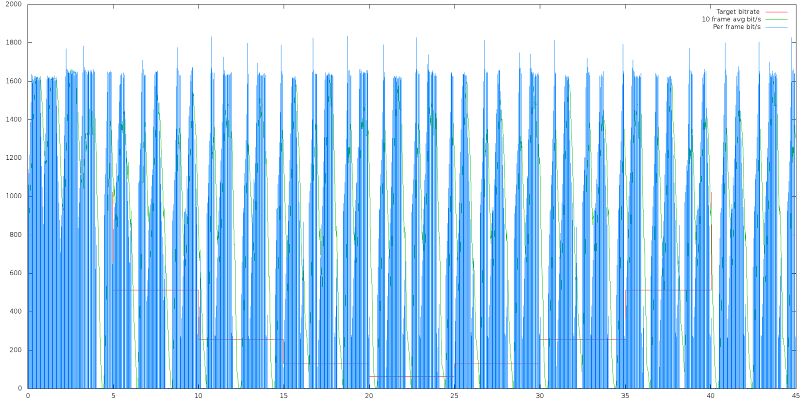
Bandwidth Adaptation

- Farsight is gaining capabilities to adapt to the available of network bandwidth via TFRC or in future similar mechanisms.
- So the Encoder needs to be able to switch bitrate on the fly

Adapting to available bandwidth (unconfigured x264enc)



Adapting to available bandwidth (some dsp h264 encoder)



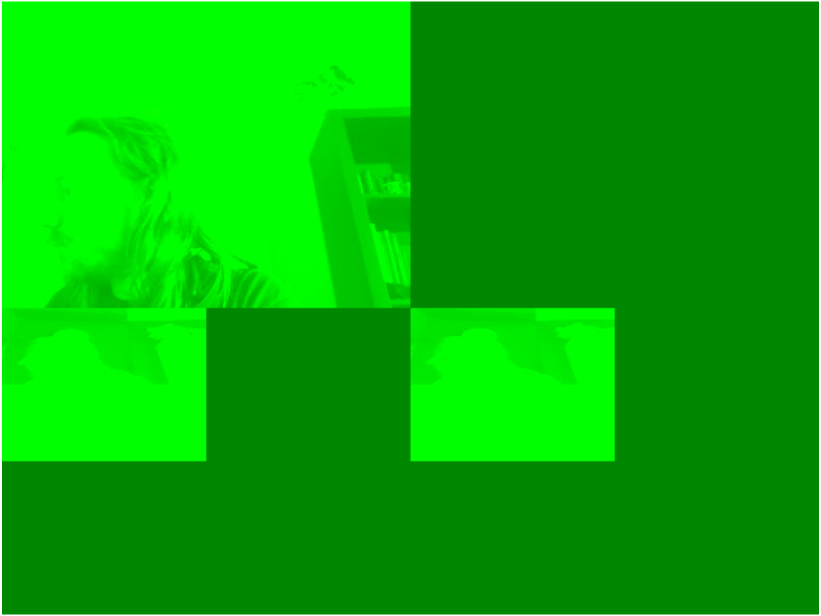
Adapting to available bandwidth (framerate)

- Encoder needs to be able to cope with a change in framerate
  - Can normally purely be done on the Gstreamer level

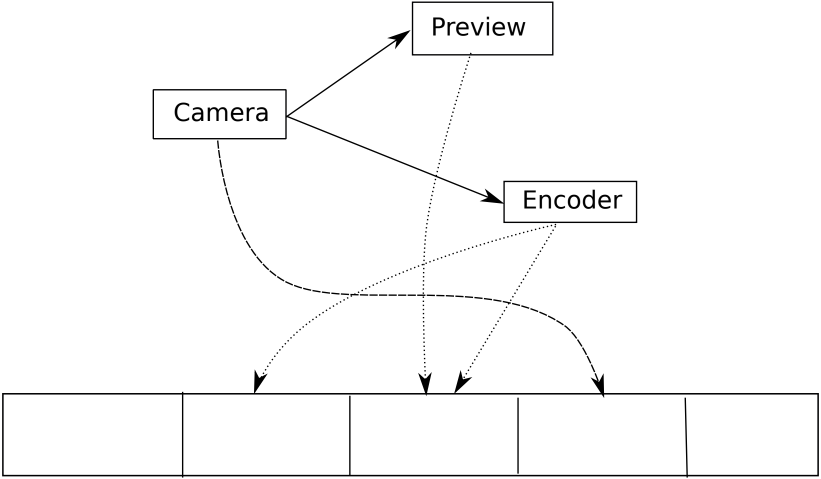
Adapting to available bandwidth (resolution)

- Encoder needs to be able to switch resolution on the fly
  - can be done by simply resetting the underlying hw encoder
- Decoder needs to be cope with this!

Decoder failing on resolution update



And now we still need to make it fast



So how to make this better

- Make people aware of our requirements (This talk).
- Better testsuites
  - gst-qa-system (aka Insanity)

Questions?