W3C WebRTC WG / Media WG / MEIG Joint Meeting

TPAC 2022, 15 September 2022 15:30 - 17:30

Safety reminders

While attending TPAC, follow the health rules:

- Authorized masks are required indoors at all time (no exceptions). If you need to remove your mask during the meeting, keep it short
- Daily test is expected

Respect each other's needs for physical distancing and other accommodations or precautions due to health concerns

https://www.w3.org/2022/09/TPAC/health.html

Code of conduct

- Appreciate and accommodate our similarities and differences, be inclusive
- Have empathy when discussing sensitive issues
- Treat everyone with respect
- Be honest, be truthful
- Be aware of how much time is taken up
- Be sensitive to language differences
- Respect confidentiality and privacy

Refer to https://www.w3.org/Consortium/cepc/

Meeting logistics

- We use IRC for minute taking and to manage the speaker queue. Please join https://irc.w3.org/?channels=#mediawg
  - Type **`present+ Your_Name`** to record your attendance
  - Type **`q+`** to join the speaker queue
- Please volunteer to scribe
  - See IRC Guide: https://www.w3.org/wiki/IRC
- Zoom remote participants, please mute unless you're speaking.
  - To try out WebCodecs over RTCDataChannel (not RTP!) join via a browser.
- Please avoid side conversations in the meeting room (those get picked up by microphones)

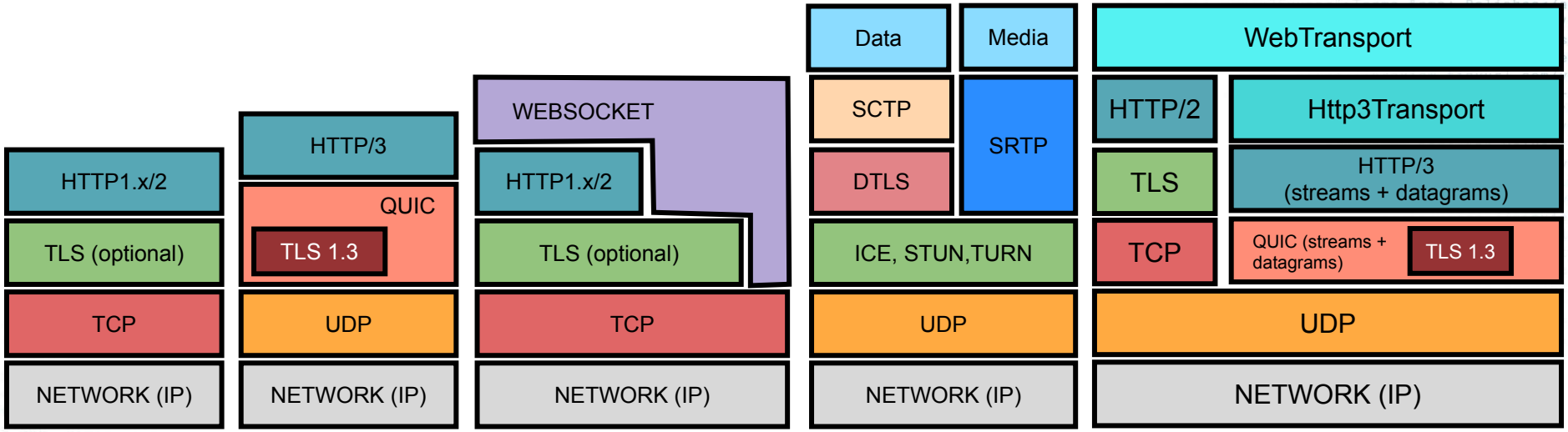
Introduction

- The Pandemic Challenge
- Protocol Evolution
- Next Generation Web media APIs

The Pandemic Challenge

- A new wave of technology reached the mass market during the pandemic:
  - Podcasting
  - Video conferencing
  - Video streaming services (live, video-on-demand)
  - Game streaming
  - IoT devices (doorbells/security, exercise equipment, robots, smart speakers)
  - Large scale webinars, classes, “town hall” meetings (100K+ viewers)
  - Online events (auctions, conferences, sporting events, concerts)
  - Interactive entertainment (co-watching, “together mode”, etc.)
- Compiled in WebRTC-NV Use Cases, WebTransport Use Cases
- The new use cases blur the lines between “streaming” and “realtime communications” and create a new challenge:
  - Can we develop web APIs (and protocols) useful for both “streaming” and RTC applications (and combinations of both?)

Protocol Evolution (IETF)



Source: WebTransport WG, TPAC 2022

Next generation Web media APIs

Overcome the “tyranny of OR”. Multi-threaded applications can deliver ***both*** low-latency ***and*** large scale, through low-level access to building blocks:

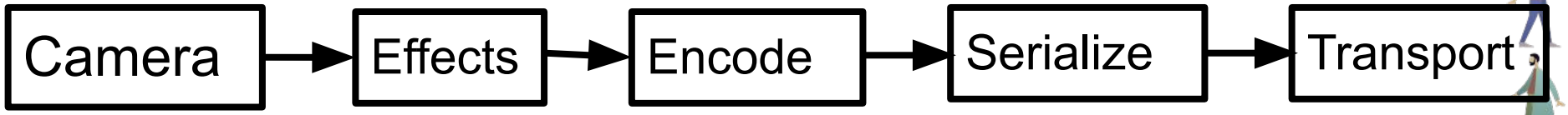
- Capture
- A/V processing (e.g. machine learning)
- Encode/Decode
- Transport
- Rendering

Next generation Web media APIs

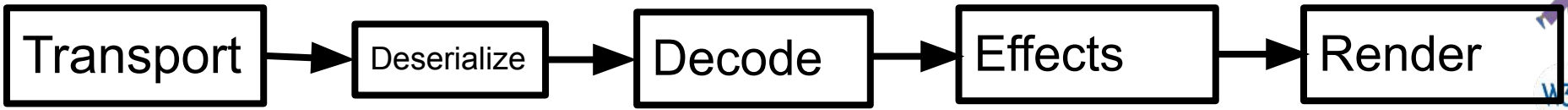
- Capture
  - Media Capture and Streams Extensions
  - Mediacapture-transform
- Encode/decode
  - WebCodecs
  - MSEv2
- Transport
  - WebTransport (HTTP/3 over QUIC)
  - WebRTC data channel in Workers (SCTP/DTLS/UDP)
- Framework
  - WHATWG Streams
  - Web Assembly

The “Pipeline” Model (WHATWG Streams)

- Send



- Receive



Challenges: Coordination

- Development of next generation protocols (QUIC, HTTP/3, WebTransport, MoQ) occurs in IETF, but there are also industry forums (DASH, AoMedia, etc.)
- Development of the next generation of Web media APIs is spread across multiple WGs in W3C and WHATWG:
  - WHATWG: WHATWG Streams
  - MEDIA WG (WebCodecs, MSEv2, EME, etc.)
  - WEBRTC WG (WebRTC-PC, RTCDataChannel, media capture, screen capture, mediacapture-transform, encoded-transform, etc.)
  - WebTransport WG (WebTransport)
  - Rendering: Canvas, WebGL, WebGPU, etc.
  - WASM
  - Implies that developers need to understand multiple APIs to create “next generation” applications
  - Sample code important (and more complex).

Challenges: Transport

- Many applications require not just encode/decode of media, but also transport.
- WebRTC's RTP transport is not directly accessible.
- RTCDataChannel (NewReno) and WebTransport (BBRv1) congestion control algorithms are not optimized for realtime communications.
  - In server -> client communications (e.g. cloud gaming), you can deploy more appropriate algorithms on the server without interoperability issues.
  - Where the client needs to send media with low-latency (to a server or another client), there is a problem.
  - Paper: https://www.in.tum.de/fileadmin/w00bws/cm/papers/epiq21-rtp-over-quic.pdf
  - Presentation (starts at Slide 14): https://datatracker.ietf.org/meeting/112/materials/slides-112-avtcore-ietf-112-avtcore-03

Challenges: “Seams”

- With API development spread across 2 SDOs and 6+ WGs, it is easy for “seams” to develop across the APIs. Examples:
  - Issues with WHATWG Streams in media use cases.
  - Conversion between VideoFrame and other formats (e.g. WebGPU external textures).
  - Leveraging WebCodecs within WebRTC (RTPTransport, encoded-transform).
  - Support for worker threads across the entire pipeline. Examples:
    - No single browser supports *both* RTCDataChannel in workers and MSE in workers.
    - Not easy to separate worker threads for send and receive pipelines.

Goal for Today

- We will not attempt to discuss (or even enumerate) ***all*** of the issues today, only a few.
- Goal is not necessarily to ***solve*** problems - but to understand where we are, and:
  - Enumerate the owners and next steps.
  - Identify topics for future discussion.

Agenda (1 of 2)

- WebCodecs + WebRTC
  - Issue 90: Add feedback streams in RTCRtpScriptTransformer
  - Issue 121: Need APIs for using WebCodecs with PeerConnection
- RTP transport issues
  - Issue 131: Should we expose packetization level API to RTCRtpScriptTransform?
- Interactions between WebRTC-PC and WebCodecs
  - Issue 141: Relationship to WebCodecs
  - Issue 70: FaceDetection metadata & VideoFrame
  - Issue 99: Encoded frame IDLs share definitions with WebCodecs

Issue 90/Issue 121: Pluggable codecs

- This involves using WebCodecs, EME or other processing mechanisms instead of WebRTC encoding/decoding to generate and parse data
- When integrating with WebRTC, it is important to ensure that SDP negotiation tells the truth of what is being sent across the wire
- This means that codec descriptions must be under application control.

Suggestion: We should allow codec descriptions to be injected into the SDP negotiation machinery, invoking known packetization/depacketization functions, and allow the app to figure out if the injected codec is acceptable to the remote party.

Transforms should operate in terms of codec descriptions, not PT.

Issue 131: Packetization API

- Use cases where WebRTC encoded transform is missing support
  - RED transform that hits the MTU limit
    - Expose MTU value (plus event) in transformer
  - Use SFrame at the packet level (SPacket) instead of frame level
    - Extension to RTCEncodedVideoFrame on sender side
      - Provide RTP packet payloads instead of encoded frame data
  - Read/Write access to RTP headers like voice activity
    - Extend RTCEncodedAudioFrame & RTCEncodedVideoFrame
      - Dedicated getters/setters of useful supported RTP headers

- Do we have enough of a use case to do any of this?

Issue 99 & Issue 141: WebCodecs & WebRTC

- Similarities between some structures
  - RTCEncodedAudioFrame vs. EncodedAudioChunk
  - RTCEncodedVideoFrame vs. EncodedVideoChunk

- Some differences
  - Data is mutable in WebRTC, not WebCodecs
  - WebRTC specific metadata needed

- Proposal: do not use WebCodec main types
  - Reuse WebCodec subtypes when feasible
    - RTCEncodedVideoFrameType ↔ EncodedVideoChunkType
    - Metadata

Issue 70: WebCodecs & MediaStream transform

- Desire for WebRTC pipeline to manipulate VideoFrames + metadata
  - Face detection metadata from camera up to encoder, via transforms
    - OS based Object detection?
  - AR/VR metadata generated by WebRTC encoded transform
    - Attached to VideoFrame by WebRTC decoders

- Requirements
  - Metadata readable/writable by WebRTC encoders/decoders
  - Metadata persisting VideoFrame cloning/transfer/serialization

- VideoFrame does not have (yet) a mechanism for generic metadata
  - What should we do?

Issue 70: WebCodecs & MediaStream transform

- Proposal 1: wrap VideoFrame around our own WebRTC object

- Proposal 2: design a generic mechanism API with WebCodecs WG
  - Add specific face detection metadata API in VideoFrame
    - In WebCodecs or in WebRTC specs
  - Metadata is mutable
    - Cloned when VideoFrame is cloned or when sent to encoder
  - Metadata is any JS that can be structure cloned

- Proposal 3: add specific metadata definition in WebRTC spec
  - Start with face detection
  - 2a: Add generic VideoFrame metadata mechanism in WebCodec spec
    - Do not expose it to the Web
  - 2b: Add specific VideoFrame metadata mechanism in WebRTC spec

Agenda (2/2)

- WebCodecs and MediaCapture-Transform (20 minutes)
  - w3c/webcodecs #189 WebCodec as pass through to application metadata extension
- WebCodecs and VideoFrame associated metadata (20 minutes)
  - Face Detection API metadata - see explainer
  - w3c/webcodecs #198 Emit metadata (SPS,VUI,SEI,...) during decoding
- WebCodecs (20 minutes)
  - w3c/webcodecs #371 AudioEncoderConfig.latencyMode (or similar) extension
  - w3c/webcodecs #405 Fixed audio chunk size support extension
  - w3c/webcodecs #270 Support per-frame QP configuration by VideoEncoder extension
  - We have 30 minutes allocated to WebCodecs for any issues not covered today