Introducing “Peeps”

An Extended MoQT Object Model

“Get them before they’re stale”

What's a group?

- Also a term that exists in applications, leading to baggage in discussions
- Provides a convenient join point
- *Sometimes* contains dependent objects
- *Sometimes* contains multiple independent groups of dependent objects
- *Sometimes* in-order delivery is important
- Can skip objects in the middle of the group

These ambiguities make it hard to map to QUIC streams and other transport concepts

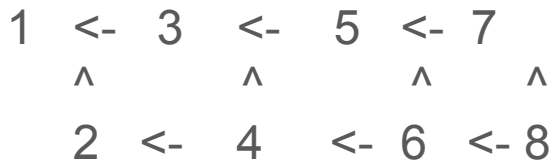
Forwarding Preference

Track, Group, Object, Datagram

- Helps to support cases where dependencies fall in one of those four scopes
- If there are separate dependencies in a group, this doesn't really help
  - Can build using object-per-stream by using complex priorities and/or separate tracks
- Applies to tracks, so can't be adjusted for subscriber conditions

Stream Resets and end-of-foo markers are hard to reason about with four different stream mappings

A Use Case Example



In the current spec:

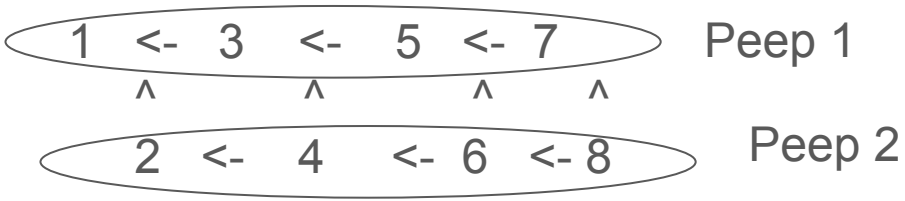
- Each object on its own stream OR
- Separate tracks (ugh) OR
- These objects will be delivered in ID order, resetting before 6 will cause you to lose (higher priority) 7

So what's a Peep?

Group					Group		
Peep		Peep		Peep	Peep		
Object	Object	Object	Object	Object	Object	Object	Object

- They SHOULD contain objects with a dependency
- The application MAY put dependent objects in different peeps (e.g. to fit peeps in a datagram, or because the ordering relationship is inappropriate for a stream)
- They MUST be a group or subset of a group
- They MUST be delivered in increasing Object ID order
- They MUST map to QUIC/WebTransport streams
- They have a single priority
- Does not preclude apps putting internal structure in objects
- Subscribers do not need to know about peeps, except as relays
- Do we want to declare dependencies between peeps? Priority is probably enough
- MoQT would work fine without group IDs, but if apps want to keep it for the API that's fine

A Use Case Example



Wire Image Changes

- Abolish forwarding preferences - one stream per Peep.
- Add a peep ID in case subscribing relays/caches receive a peep over two sessions
- One Peep ID is reserved to indicate an object should be sent via Datagram

API Changes

- Application declare objects to belong to exactly one peep
- Smallest unit of priority is peeps
- Peeps are the unit of expiration/cancellation

Questions: Do you want to see PR(s) that:

(These are independent questions)

- allow an application to define how to use multiple streams inside a group?
- eliminate stream-per-track forwarding preference?