802.1Qat
Multiple Talkers One Stream

Rodney Cummings
National Instruments
Rodney.Cummings@ni.com

From Gen 2 Assumptions

- Two-way reservations – decided no need to support
  - Must be handled at a higher layer
- Multiple Talkers per Stream (one streaming at a time)
  - Networked video switcher
    - Switch on a bit in a stream or switch at a specific time
    - Or do the Talkers do all the turning on or off (i.e., the MUX'ing)
      - Need the concept of a Group Reservation
- Multiple Talkers per Stream (time sliced approach)
  - Industrial control (http://www.ieee802.org/1/files/public/docs2010/at-goetz-AVB-lowlatency-part1-0510.pdf)

Searching for Common Requirements

- For two-way
  - Both paths don't exceed 75%
  - Therefore, two streams meets the requirement
- For video switcher
  - Each path can use 75% (never used simultaneously)
  - Therefore, two talkers need to share a stream (reservation)
- Goal of multi-talker: Share a bandwidth reservation

Who Guarantees the Reservation?

- All talkers for stream must never exceed bandwidth
- For industrial time-slicing
  - Higher layer in end-stations runs the schedule
- For video-switcher
  - Seems strange for SRP to decide when to switch
    - Beyond SRP's mission of reservation
  - Assume use of higher layer in talkers, or new 802.1 protocol in bridges
- Conclusion: Non-SRP protocol must guarantee
  - Egress port on bridge must meet stream's reservation

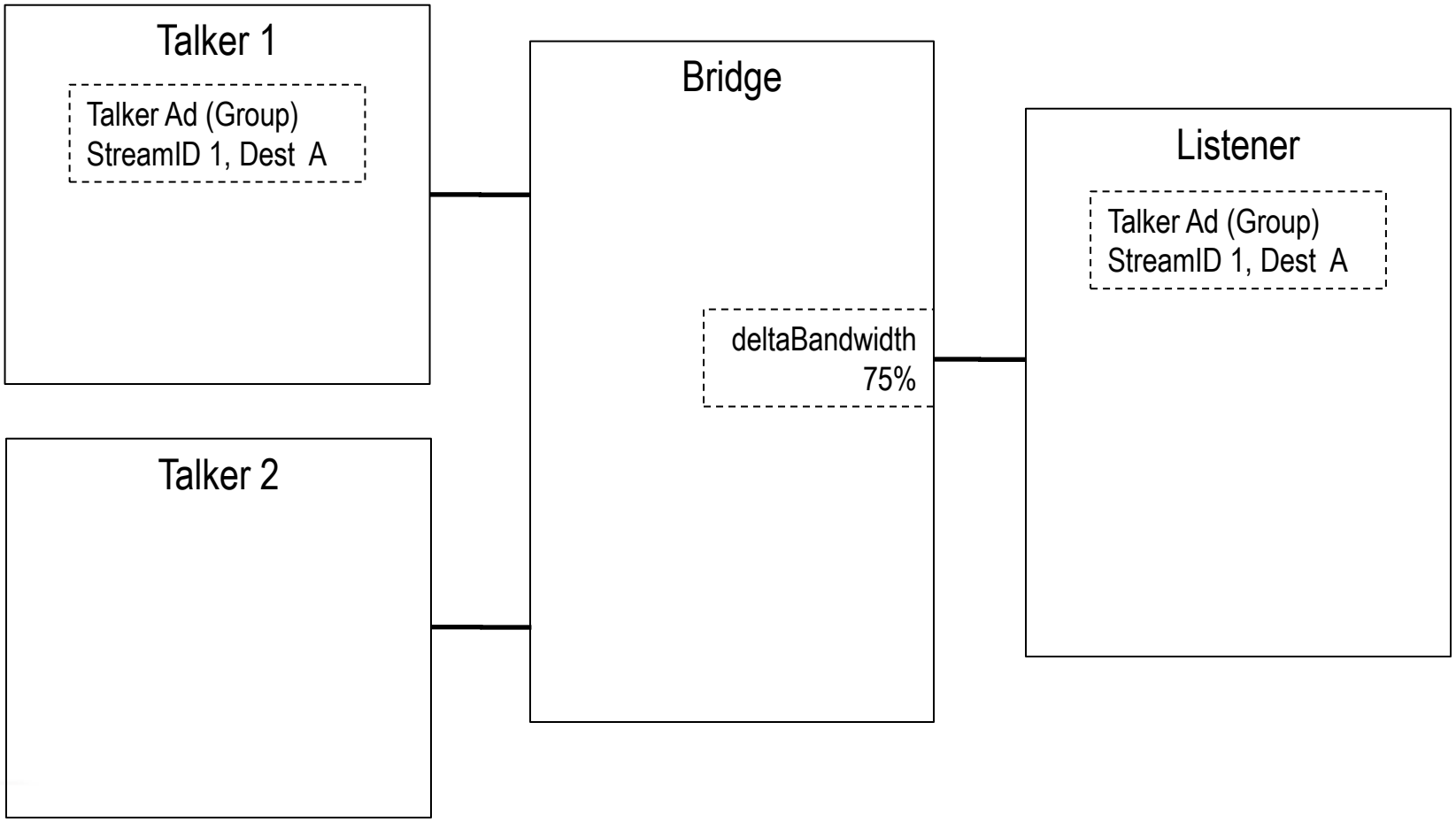
Talker Group

- Propose name of feature: Talker Group
- Concept is a group of talkers who
  - Coordinate through another protocol
  - Agree to share the bandwidth of one stream

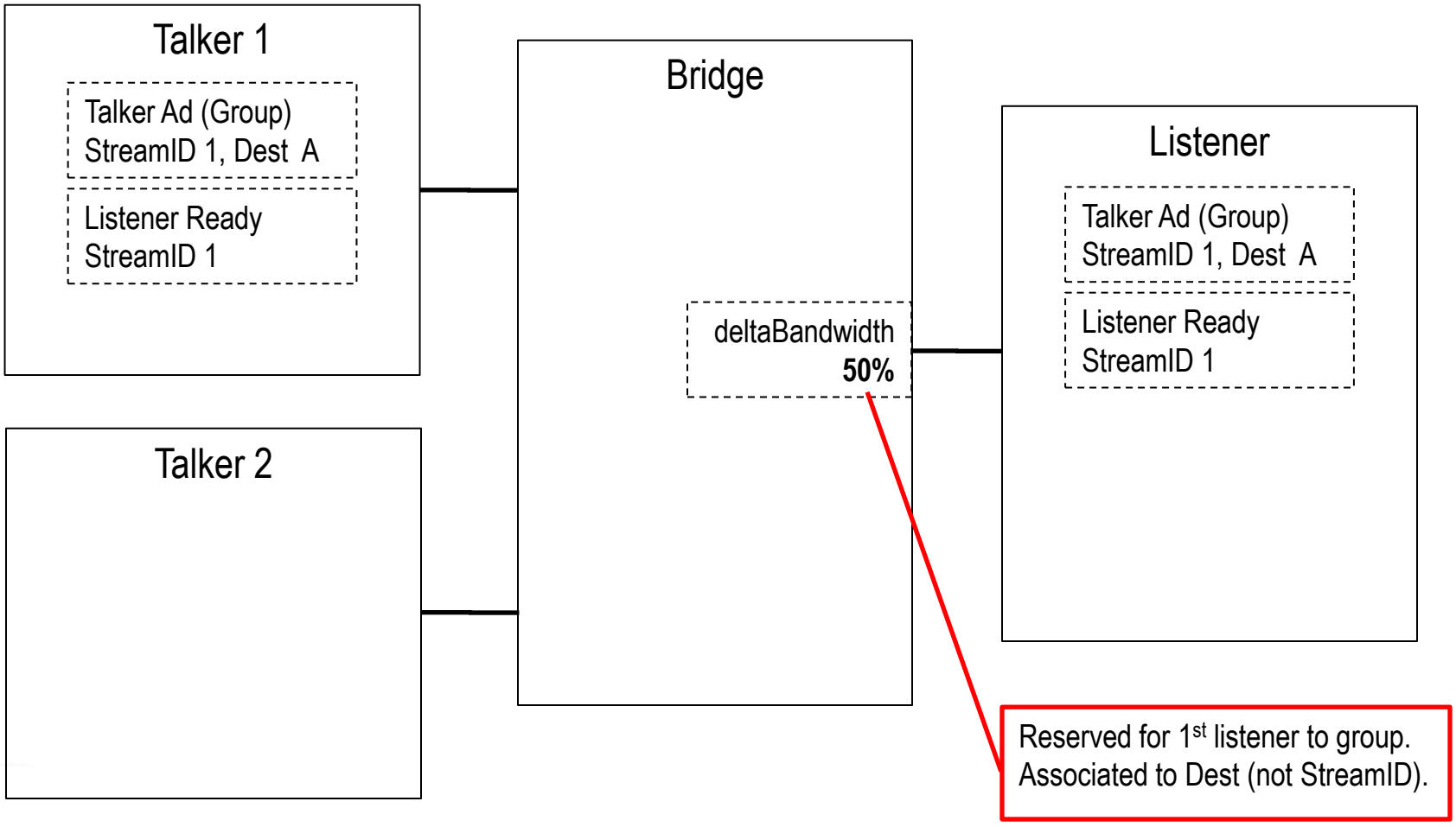
Proposal 1: Mult Stream, One Reserve

- Add Group bit to Talker MSRPDU
  - New attribute types, or new bit in *Tspec*
- Each Talker in group uses distinct StreamID
  - All in group use same *DataFrameParams* (Dest MAC)
- Bridge does not merge talkers
  - Listening end-station sees multiple Talker Advertise
- When Bridge sees Listener declarations, if Group true for that StreamID:
  - Reserve bandwidth for first Listener to group
  - Unreserve bandwidth for last Listener to group

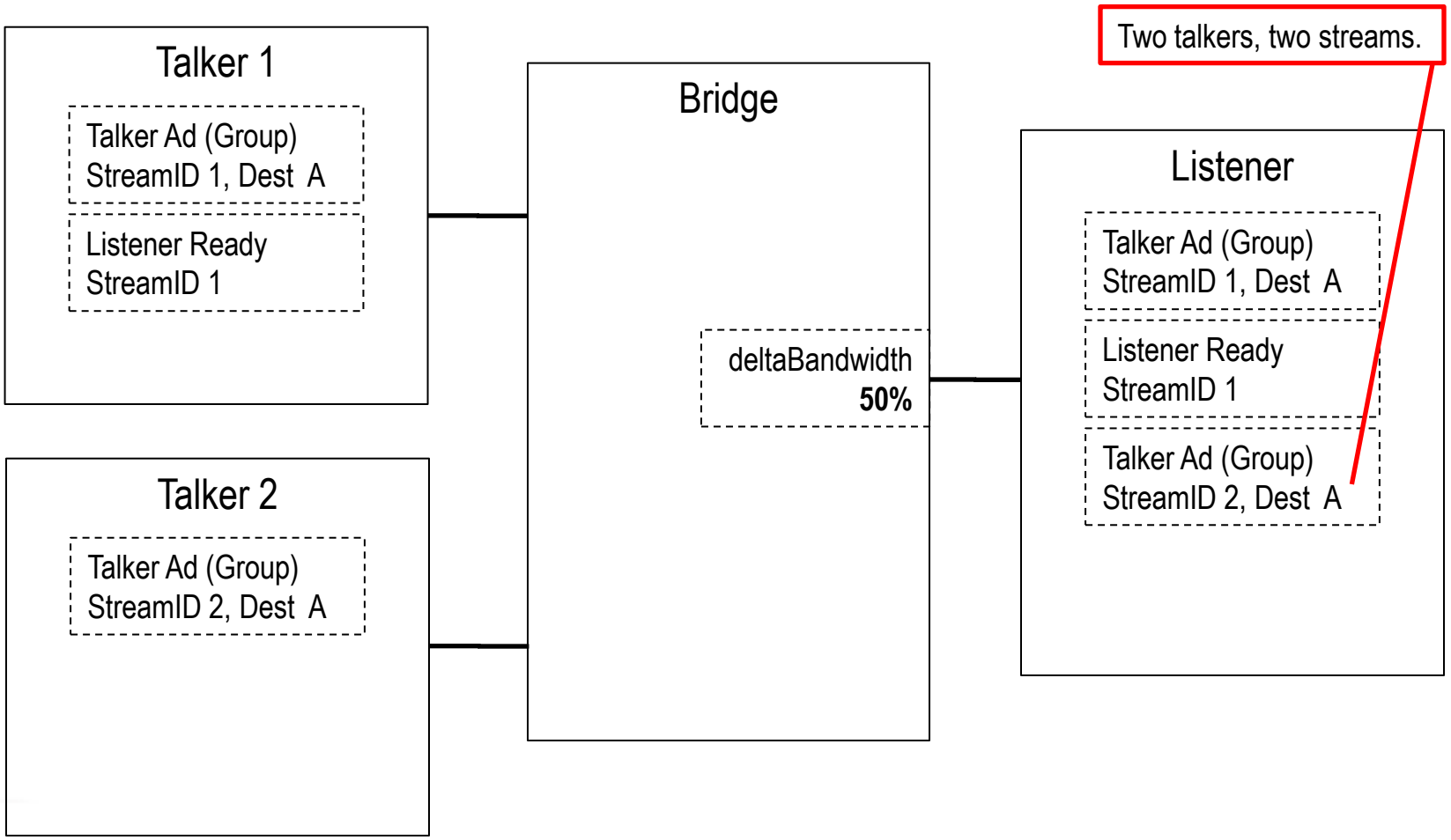
Proposal 1: Mult Stream, One Reserve



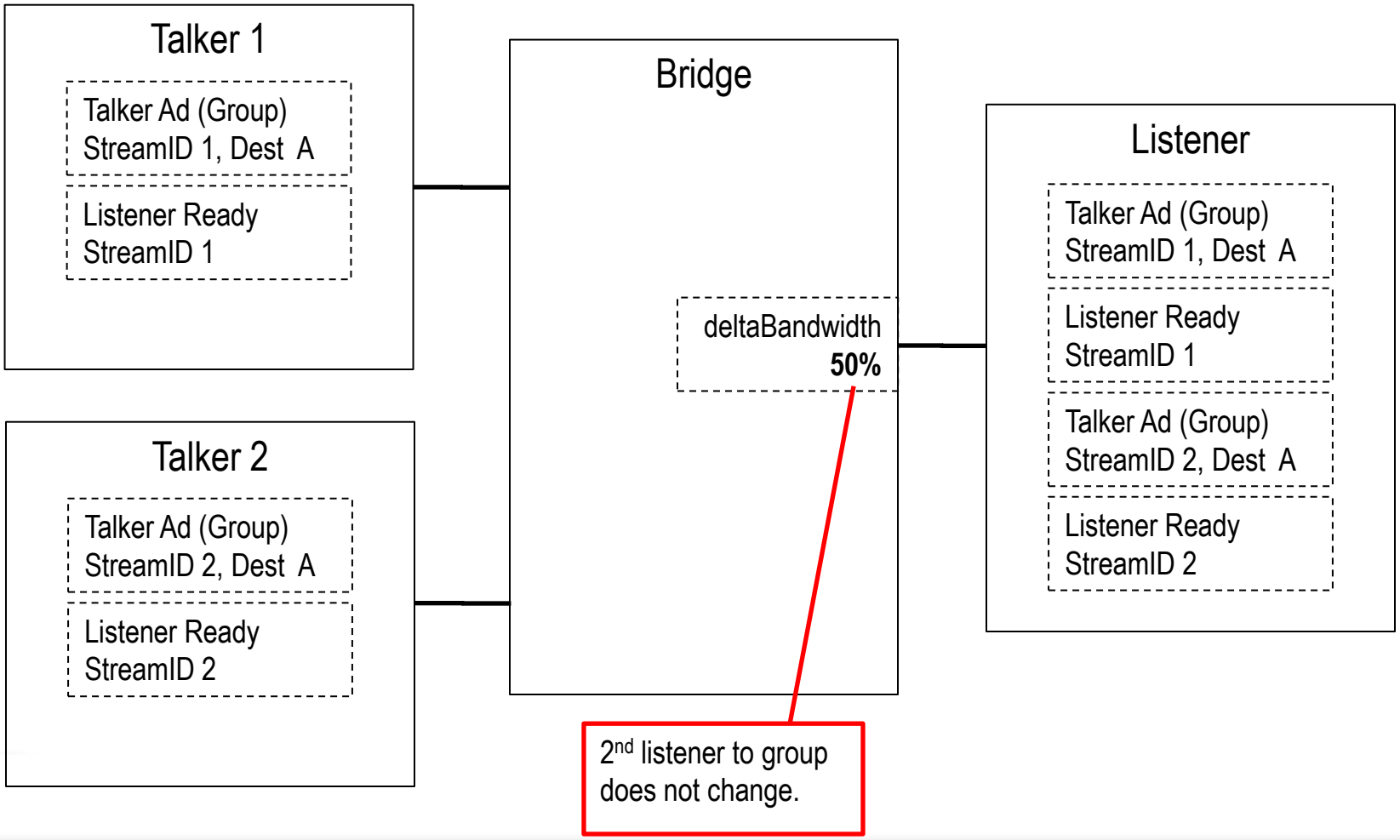
Proposal 1: Mult Stream, One Reserve



Proposal 1: Mult Stream, One Reserve



Proposal 1: Mult Stream, One Reserve



Proposal 1: Mult Stream, One Reserve

- Pros
  - Minimal change: merge reservation only
  - Listener knows which talkers are active
  - Stream ID can use source MAC of talker
- Cons
  - Listener must track multiple streams to know if group exists
  - Does not reduce attributes

Proposal 2: One Stream, One Reserve

- Background on StreamID
  - **3.174 StreamID:** A 64-bit field that uniquely identifies a stream.
    - Implies one stream uses one streamID
    - Various places in Qat assume this…

      12.22.4 SRP Stream Table

      There is one SRP Stream Table per bridge component. Each table contains a set of parameters for each StreamID that is registered on the Bridge.

  - Are we required to use Source MAC within StreamID?
    - No
    - **35.2.2.8.2 StreamID**

      NOTE 2—The MAC address component of the StreamID can, but does not necessarily, have the same value as the source_address parameter of any frame in the actual data stream.

Proposal 2: One Stream, One Reserve

- New attribute type: Talker Group
  - Use *FourPackedType* for Advertise, Failed, and Ad Failed
    - Idea is to use 1 attr type instead of 3, like Listener
- Bridge merges Talkers into one stream
  - Advertise Failed similar to Listener (e.g. 1 success, 1 fail)
  - *AccumulatedLatency* is worst of all Talkers in group

Proposal 2: One Stream, One Reserve

- Each Talker in group uses same Stream ID
  - Use Dest MAC in StreamID
    - Strange, but Source MAC returns us to Proposal 1
- Unique ID in StreamID can be used as a Group ID
  - If need multiple groups per Dest MAC

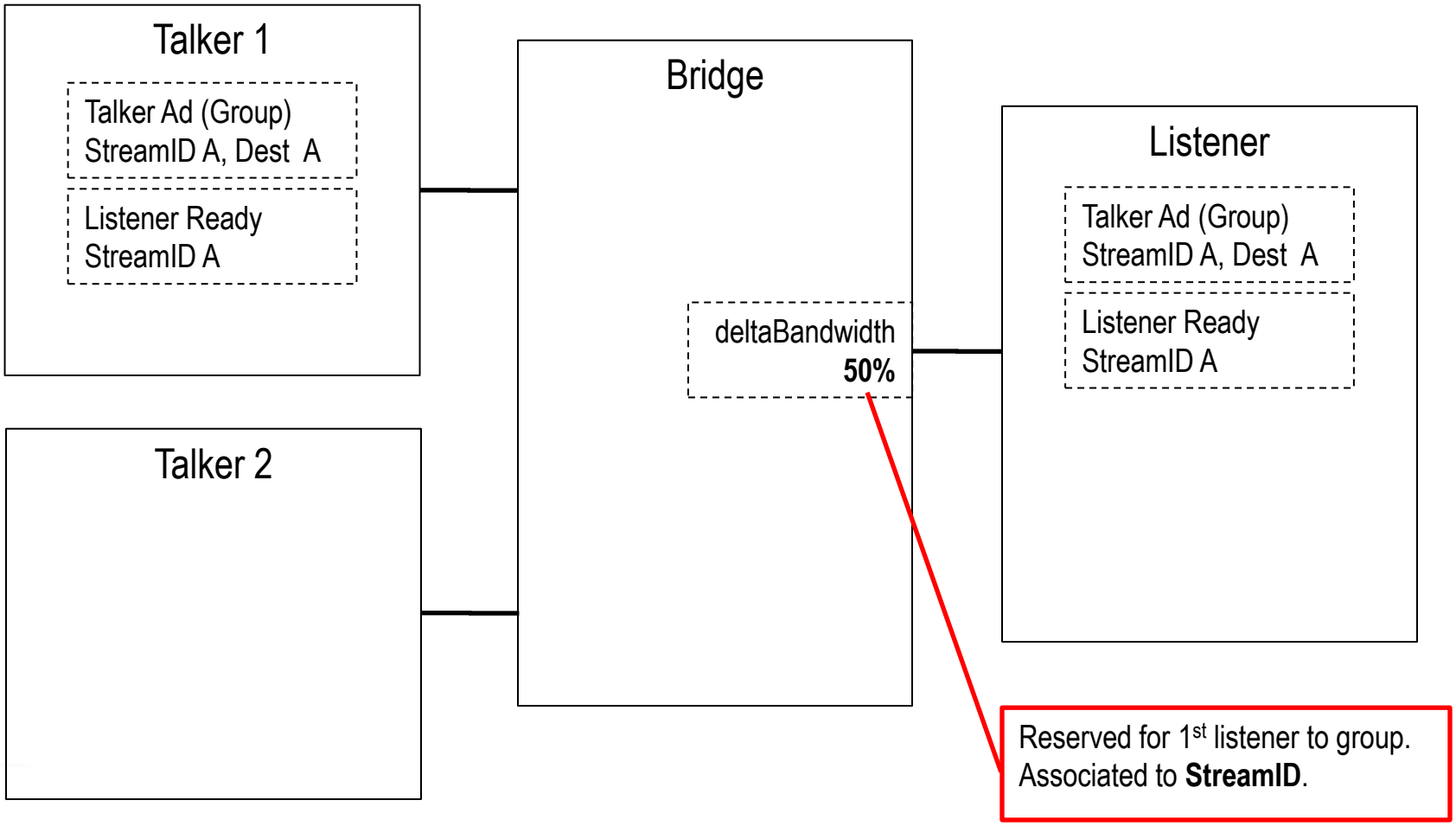
Proposal 2: One Stream, One Reserve

- Propagate latency changes as speced in 35.2.6?
  - If *AccumulatedLatency* changes, !rLv then !JoinIn/Mt
  - May have other benefits: recalculate on topology change?

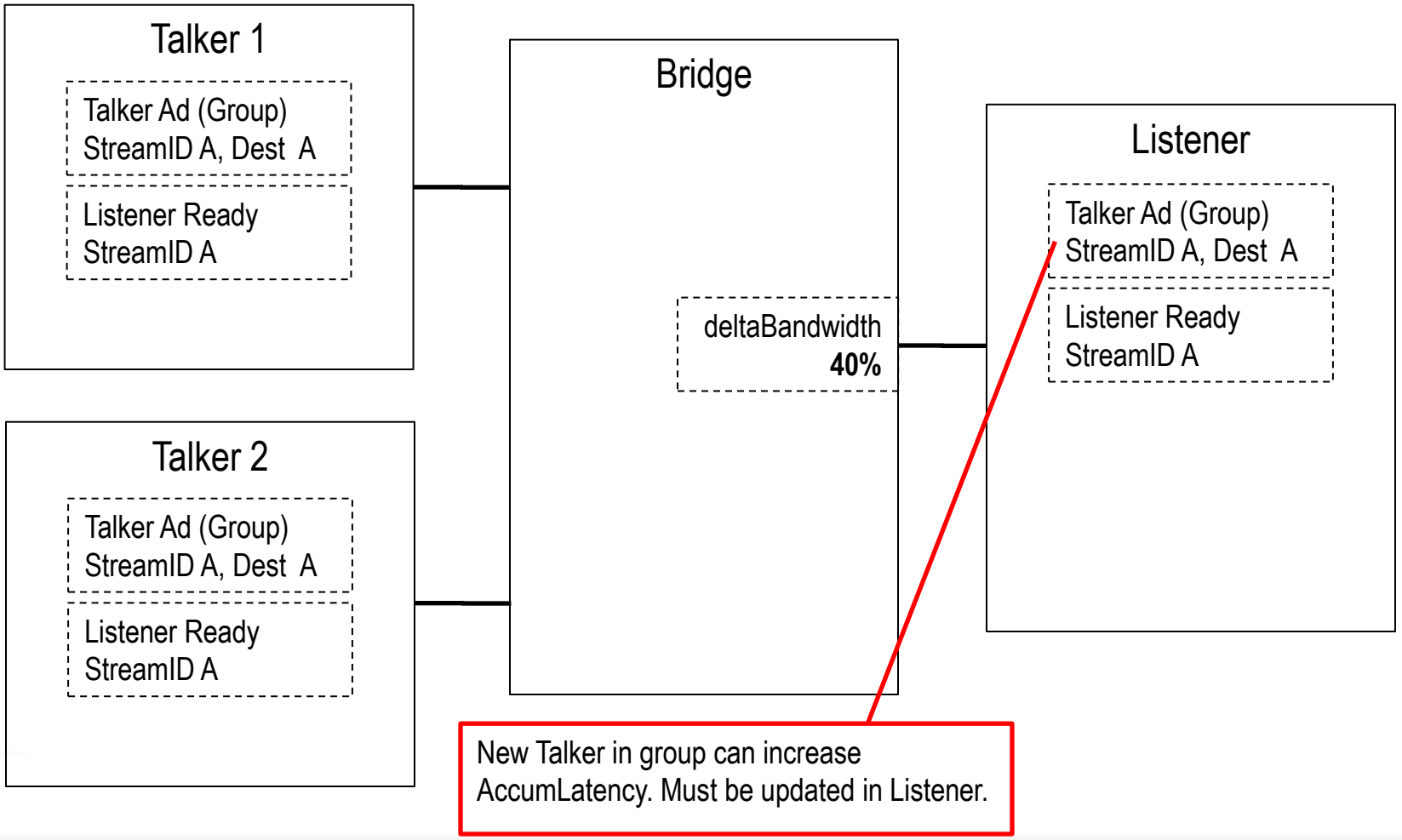
35.2.6 Encoding

If an MSRP message is received from a Port with an event value (35.2.6) specifying the JoinIn or JoinMt message, and if the StreamID (35.2.2.8.2), and Direction (35.2.1.2) all match those of an attribute already registered on that Port, and the Attribute Type (35.2.2.4) or FourPackedEvent (35.2.2.7.2) has changed, then the Bridge should behave as though an **rLv!** event (with immediate leavetimer expiration in the Registrar state table) was generated for the MAD in the Received MSRP Attribute Declarations before the **rJoinIn!** or **rJoinMt!** event for the attribute in the received message is processed. This allows an Applicant to indicate a change in a stream reservation, e.g., a change from a Talker Failed to a Talker Advertise registration, without having to issue both a withdrawal of the old attribute, and a declaration of the new. A Listener attribute is also updated this way, for example, when changing from a Listener Ready to a Listener Ready Failed.

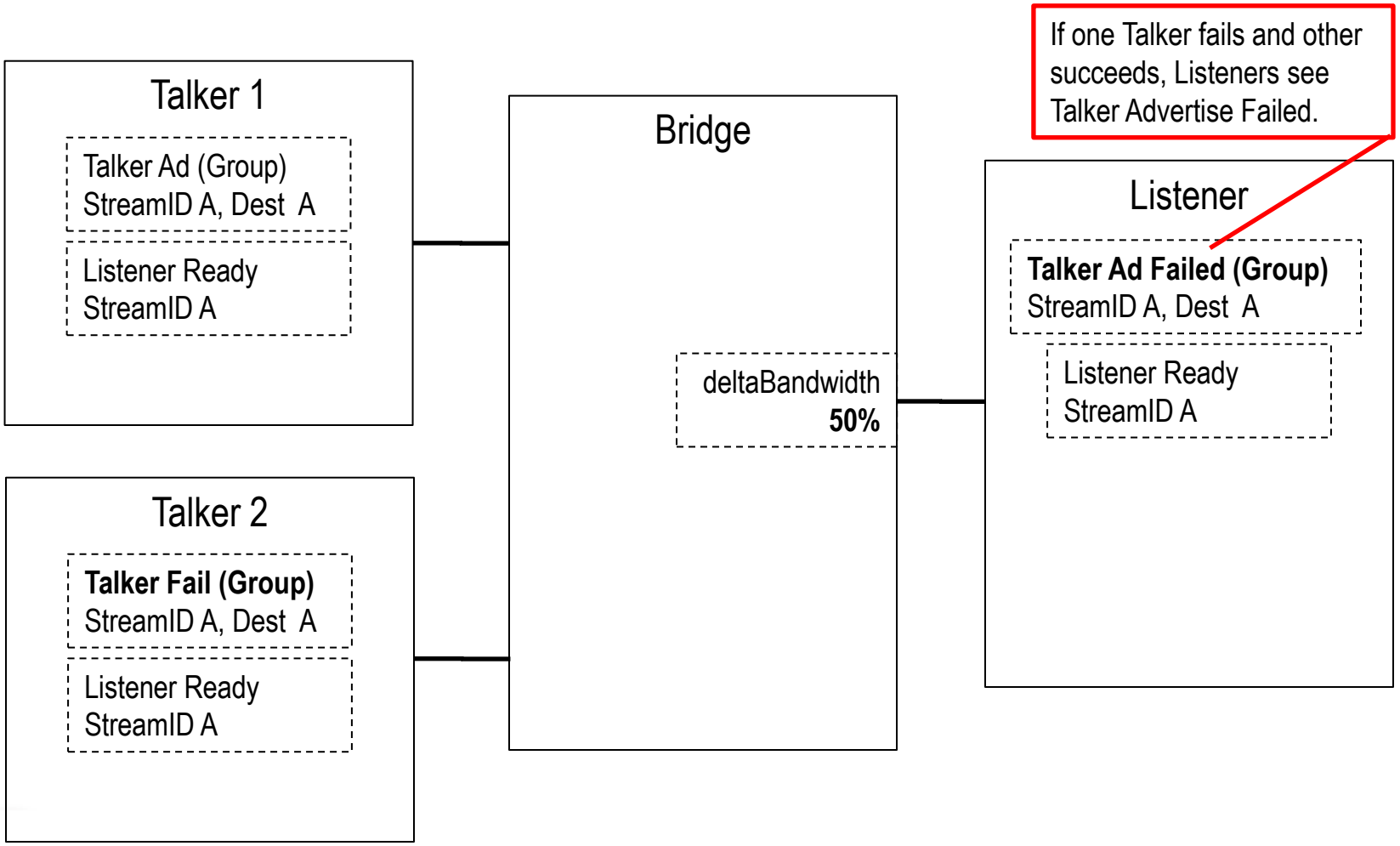
Proposal 2: One Stream, One Reserve



Proposal 2: One Stream, One Reserve



Proposal 2: One Stream, One Reserve



Proposal 2: One Stream, One Reserve

- Pros
  - Complete solution: Truly multi-talker one stream
    - Listener sees a single stream
  - Merge is similar to Gen 1 Listeners
- Cons
  - Listener doesn't know which Talkers are active
    - Best left to higher layer anyway
  - More work for bridges (merge)
- Concern
  - Can a talker in group also listen to that group?