Temporally Redundant Audio Format work item questions

Ashley Butterworth
Apple Inc

The Questions

- Does it fit in the AAF clause
  - Would it be backwards compatible??
  - Do we need it to be?
  - One version is temporally redundant; one version is not (ie: regular). Perhaps by adding a “codepoint”.

- Could we simply add a flag bit to all audio formats?
  - Or all media streaming formats?
  - What about command and control formats?
  - Is there even a way to do this?

Does this fit in AAF clause

Backwards compatibility - flags

- What happens if a 1722-2016 AAF receiver receives a TRAF packet
  - Depends on how implementation handles validation of packet fields
    - Is the stream_data_length is used to calculate the number of samples (which is expected of the implementation) what does it do when it doesn't match what it expects. i.e. if its supposed to be 6 samples in a packet and the calculaion comes out as 12 does it try to use it all as 12 samples, does it truncate after the first 6, etc.
    - Does the implementation drop packets that fail validation or does it just use what could be correct, i.e. in the above does it drop that packet or truncate

Does this fit in AAF clause

Backwards compatibility - formats

- There are plenty of format codes (full 8 bits (256) and we only have 6 defined)
  - stream couldn't be sent into an existing AAF listener (assuming proper validation of streaming packets)
- Would apply only to AAF/TRAF (i.e. not a general mechanism for adding this to all streaming formats)

Does this fit in AAF clause

Conclusion

- Adding a flag indicating TRAF vs normal is risky given unknown implementations of already existing as we don't know how they treat the existing well defined fields and they wouldn't validate the value of a reserved field.
- Adding additonal formats is a possibility.
  - Would add a TRAF PCM and TRAF AES3 section in addition to the existing PCM and AES3 sections which could be heavily referenced.

Can we add a flag to all formats

- As discussed in previous section, flags in reserved areas would not be validated by existing implementations which could mean that it incorrectly processes the packet
- We don't know what packet validation existing implementations do and how they react to failing validation to be able to safely adjust them.

- I don't think it's a good idea to try this.