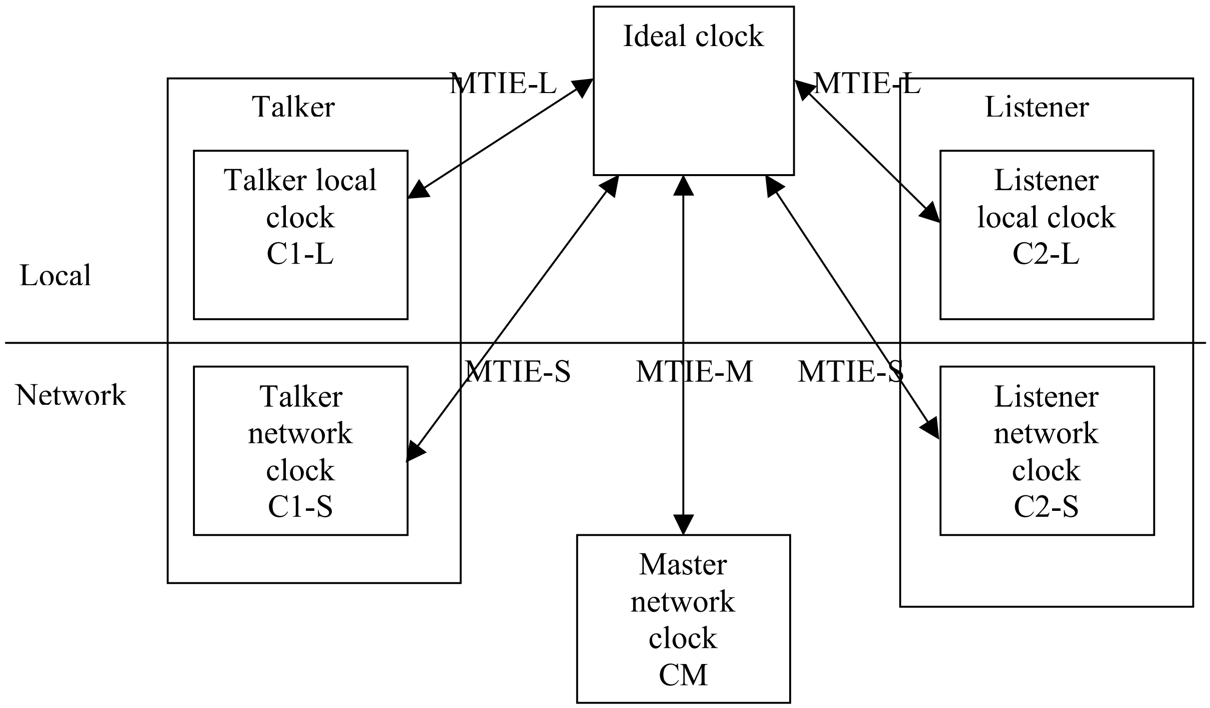
# On relation between the application MTIE and synchronization accuracy

Max Azarov, SMSC
For 802.1 AVB group




Let assume that application has specific requirements on local clocks expressed in a respective MTIE mask which we will denote as MTIE-L (L for local). In our case Talker and Listener clocks C1-L and C2-L both meet the mask MTIE-L.

We will consider now a network master clock CM with the mask MTIE-M (M for master), which distributes its clock across the network in such fashion that Talker and Listener are able to maintain their instance of the network clocks C1-S and C2-S with the MTIE mask MTIE-S (S for slave).

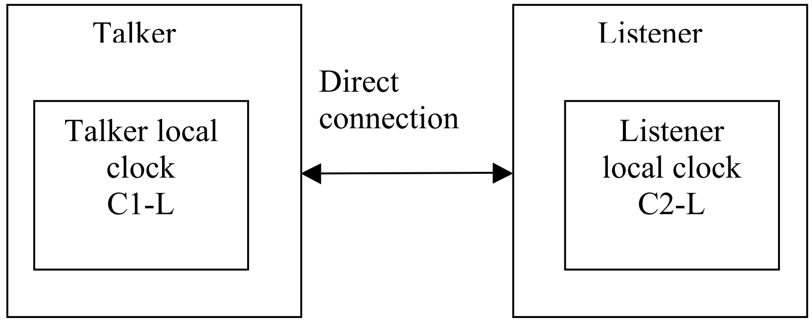



**Figure 1 - Direct connect setup**

We will assume that if we look at the configuration where Talker and Listener are connected to each other directly, they will operate correctly because their local clocks C1-L and C2-L are sufficiently well-behaved and meet the requirements for a specific application MTIE-L (vide, audio, etc.).

Now we consider configuration where Talker and Listener are connected via the network. If we assume that network has a bound latency and inter-packet delay variation, we can always establish an isochronous channel using one of two methods:

1. In-bound:

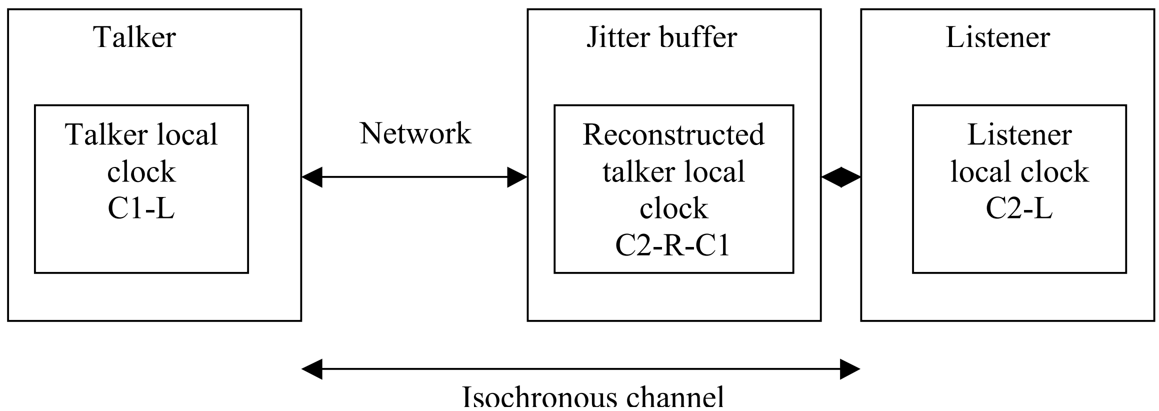



**Figure 2 - In-band clock reconstruction setup**

   - Time-stamp data on the taker side using talker local clock C1-L
   - Use these time-stamps to reconstruct the talker clock on the listener C2-R-C1
   - Use *reconstructed talker clock* C2-R-C1 to operate a jitter-removal buffer on the listener
   - Accuracy of synchronization of *reconstructed talker clock* C2-R-C1 to the talker clock C1-L will be one of parameters dictating the size of the jitter-removal buffer

2. Out-of-bound:

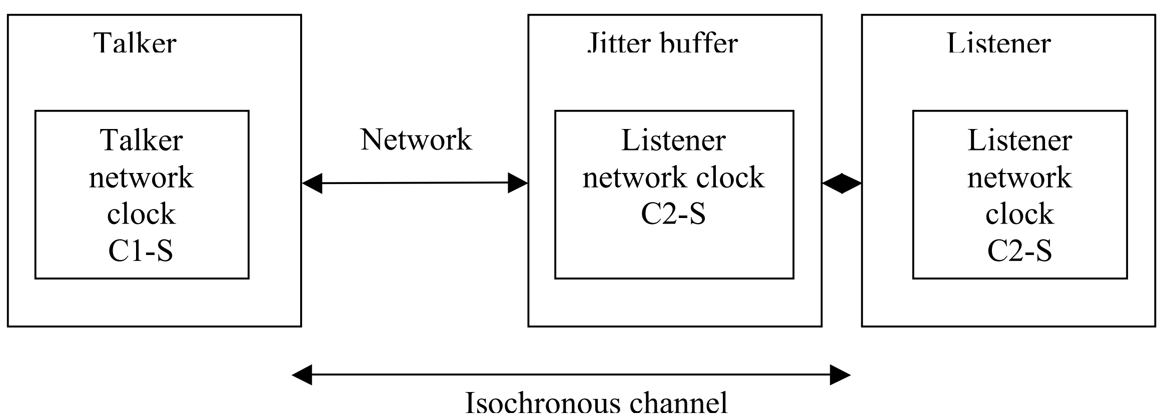



**Figure 3 - Out-of-band network clock setup**

   - Time-stamp data on the talker using talker network clock C1-S
   - Use *listener network clock* C2-S to operate a jitter-removal buffer on the listener
   - Accuracy of synchronization of *listener network clock* C2-S to the *talker network clock* C1-S will be one of parameters dictating the size of the jitter-removal buffer

In case 1, a listener will receive data timed by a reconstructed talker clock, while in case 2, a listener will receive data timed by a listener network clock. This means that in order for system to operate correctly, following conditions have to be met:

1. In-bound: *Reconstructed talker clock* C2-R-C1 has to meet the MTIE-L
2. Out-of-bound: *Listener network clock* C2-S has to meet the MTIE-L, i.e. MTIE-S should be equal or more stringent that MTIE-L

Case 1 has no relevance with the network clock, but we should mention that this method cannot provide operation of multiple synchronized talkers/listeners. Method is inherently peer-to-peer. It is true that we can have multiple listeners listening on multicast and each reconstructing single talker's clock and correctly rendering the stream, but this would work only if we don't need listeners to render simultaneously (i.e. multiple speakers). Same problem exists for multiple synchronized talkers (multiple microphones).
From network time perspective it is more interesting to look into the case 2. If we assume that listener network clock is reconstructed based on the local MTIE-L clock, MTIE-S is naturally a function of a time synchronization algorithm, MTIE-L and MTIE-M. But how exactly MTIE-S will look like? If we assume that synchronization algorithm provides certain synchronization accuracy T, that we can conclude that long-term MTIE-S values (right part of the MTIE) will approach the respective values of the MTIE-M. In other words, if master clock is 100 ppm, then the slave clock will be around 100 ppm as well because synchronization will ensure that both clocks have nearly identical drift.

In this light we can review two cases:

1. If long-term values of MTIE-M exceed than MTIE-L,
   than MTIE-S cannot meet the MTIE-L. In other words, if master network clock does not meet the requirements of MTIE-L, than slave network clocks will not meet this requirement as well and system of talker-listener will not operate with out-of-band method.
2. If MTIE-M meets or better than the MTIE-L
   Synchronization algorithm may either
   a. enable MTIE-S to meet the MTIE-L requirements, and talker-listener system will operate, or
   b. not provide enough accuracy resulting in MTIE-S failing to meet the MTIE-L and talker-listener system not operational.

As we can see, only case 2a allows for operational system and imposes some requirements to the effectiveness on the synchronization algorithm.

**Conclusions:**

1. If we have 100 ppm master clock (CM), then slave network clock certainly cannot be used on talkers and listeners for application requiring better clocking of the data. According to Geoff's requirements all application will require better than 100ppm clock except for Digital Audio Level 2.
2. Applications that require better clock still have an option of using in-band clock reconstruction, which isolates them from the imprecise network clock, but in this case multiple synchronized talkers/listeners scenario cannot be implemented.
3. Network clock synchronization accuracy is not directly related with the MTIE requirement for the applications! In case when we have a "good-enough" master clock, insufficient synchronization accuracy can be alleviated by increasing the jitter buffer. Of course this will increase the latency of the reconstructed isochronous channel as well. But as long as this additional increase is small when compared to the inter-packet arrival variation of the network, synchronization inaccuracy can likely be absorbed.

4. More importantly though, synchronization may be dictated by multiple talkers or listeners scenarios, when signal from multiple talkers should be precisely synchronized. For example, when we have video encoder and audio encoder, we would want resulting audio and video streams to be in sync. Depending on how much sync precision we want we will arrive at the requirements on network clock synchronization. Same is possible with multiple microphones recording or multiple speakers rendering (5+1 surround, etc.).

**Suggestions:**

1. Consider multiple clock domains with possibility of better clocks for better applications?
2. Alternatively, make such master selection algorithm that it can ensure that the best clock on the network is the master. This has a down-side, of disrupting whole network operation when new master has to be selected.
3. Review network clock synchronization accuracy requirements from the point of view of application with multiple synchronized talkers/listeners.