Scheduled Traffic
- One Window Is Not Enough

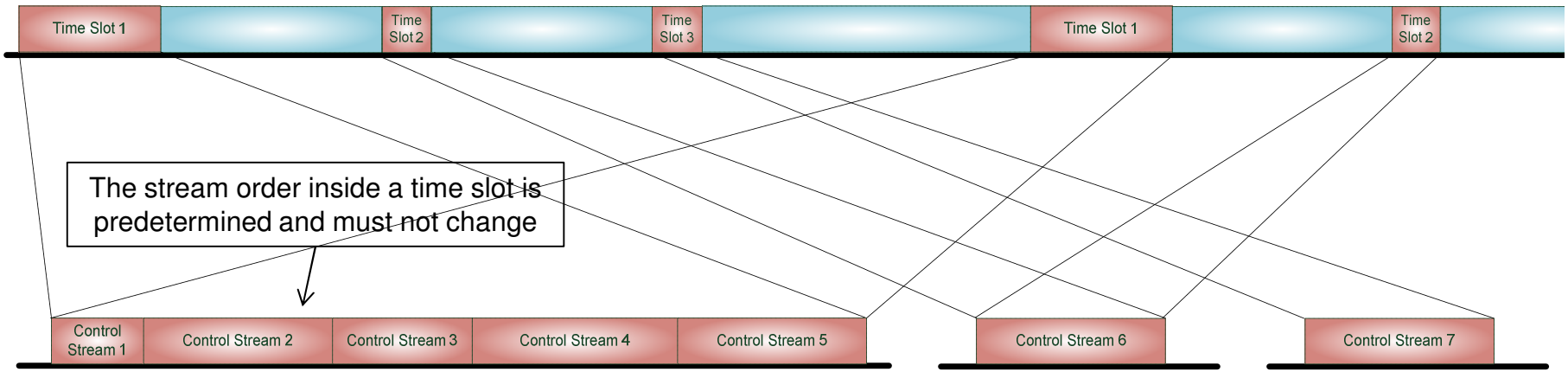
Christian Boiger
christian.boiger@hdu-deggendorf.de
IEEE 802.1 Interim
May 2012
York, UK

DEGGENDORF UNIVERSITY OF APPLIED SCIENCES

Scheduled Traffic

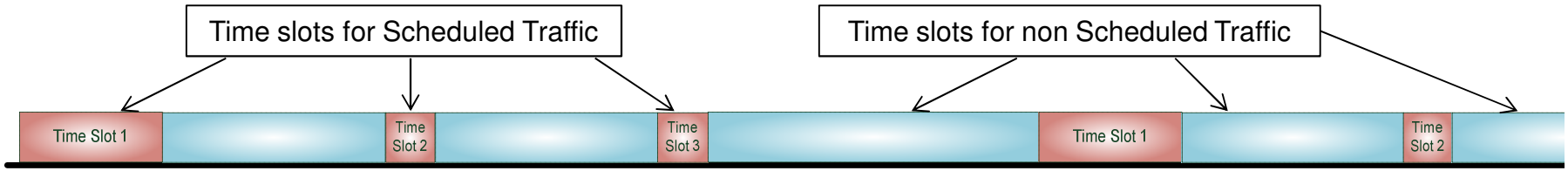
- In order to achieve really low latency and delivery variation for Scheduled Traffic two requirements have to be met:
  1. Scheduled Traffic must not interfere with non Scheduled Traffic
  2. Scheduled Traffic must not interfere with itself

- In order to fulfill this requirements it is necessary to:
  1. Separate Scheduled Traffic from the "normal traffic" (e.g. Time Sensitive Streams, Strict Priority)
  2. Schedule the streams (also inside a time slot) and keep the exact schedule



Time Aware Shaper

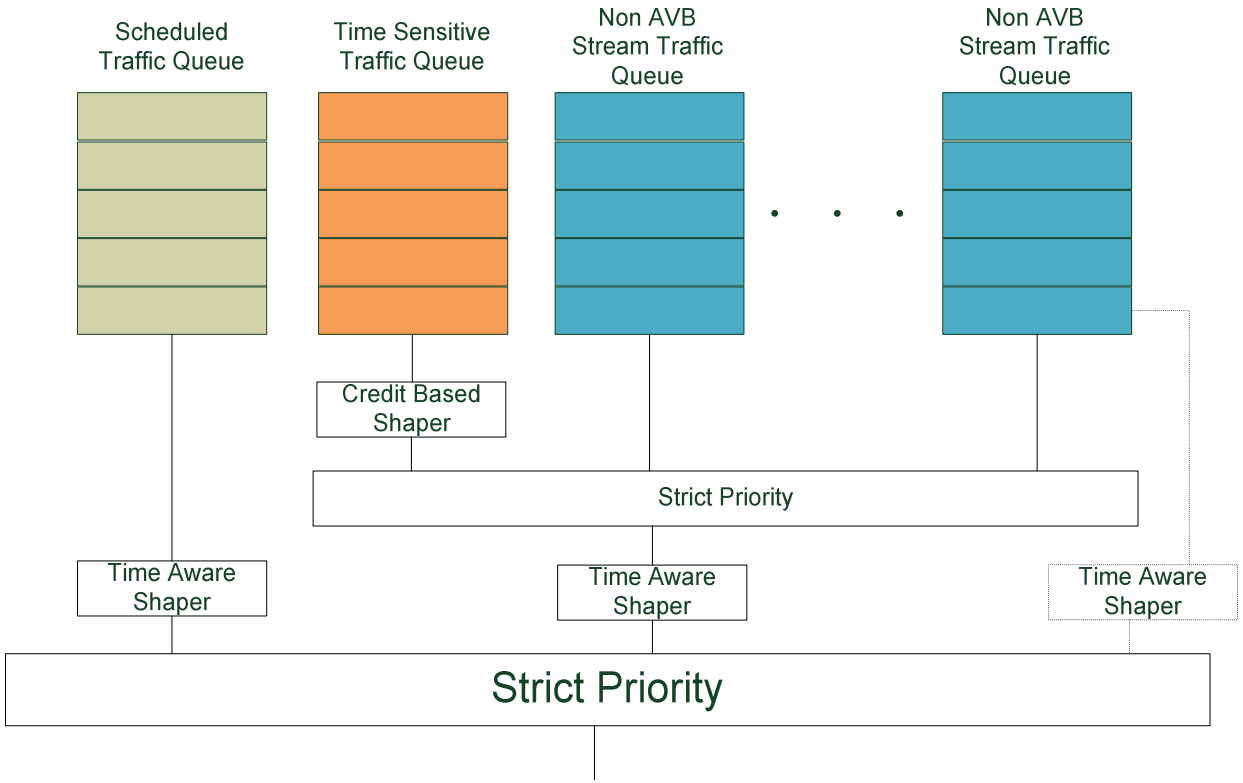
- The Time Aware Shaper (TAS) is one key element to accomplish these requirements
- The TAS blocks queues based on time
- In order to control the timeslots it is necessary that all queues are controlled by a time aware shaper (directly or indirectly)
- The TAS establish completely independent "channels" (time slots)



- The time slots prevent the interference of scheduled traffic with non Scheduled Traffic (e.g. Time Sensitive Streams)

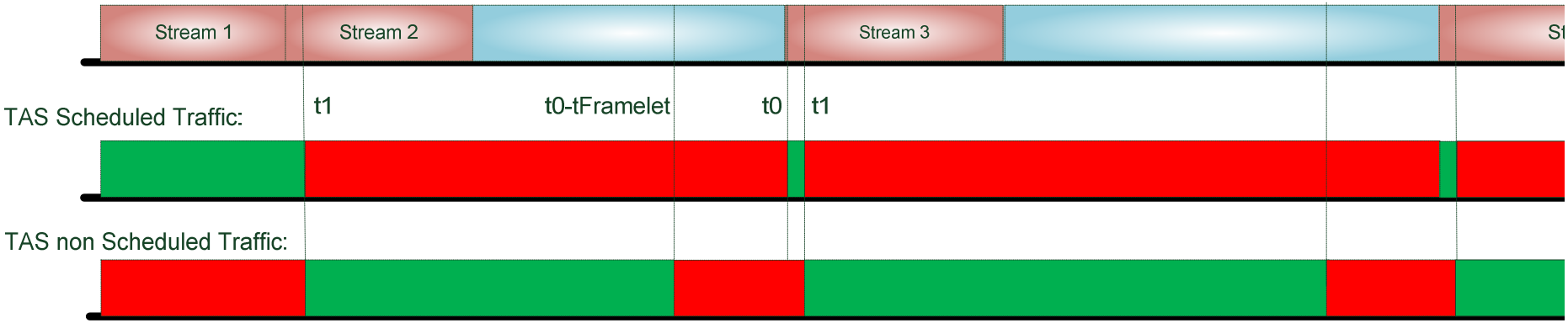
Time Aware Shaper

- Each queue is directly or indirectly controlled by a TAS
- Queues for Time Sensitive Traffic are additionally controlled by a Credit Based Shaper (CBS)
- The TAS overrules the CBS



Blocking Windows

- Blocking Mechanism



1. Scheduled Traffic is unblocked when a transmission window starts (t0)
2. Scheduled Traffic is blocked very short time after the last frame of a transmission window is scheduled for transmission (t1)
3. Non Scheduled Traffic is blocked at t0-tFramelet
4. Non Scheduled Traffic is unblocked at t1

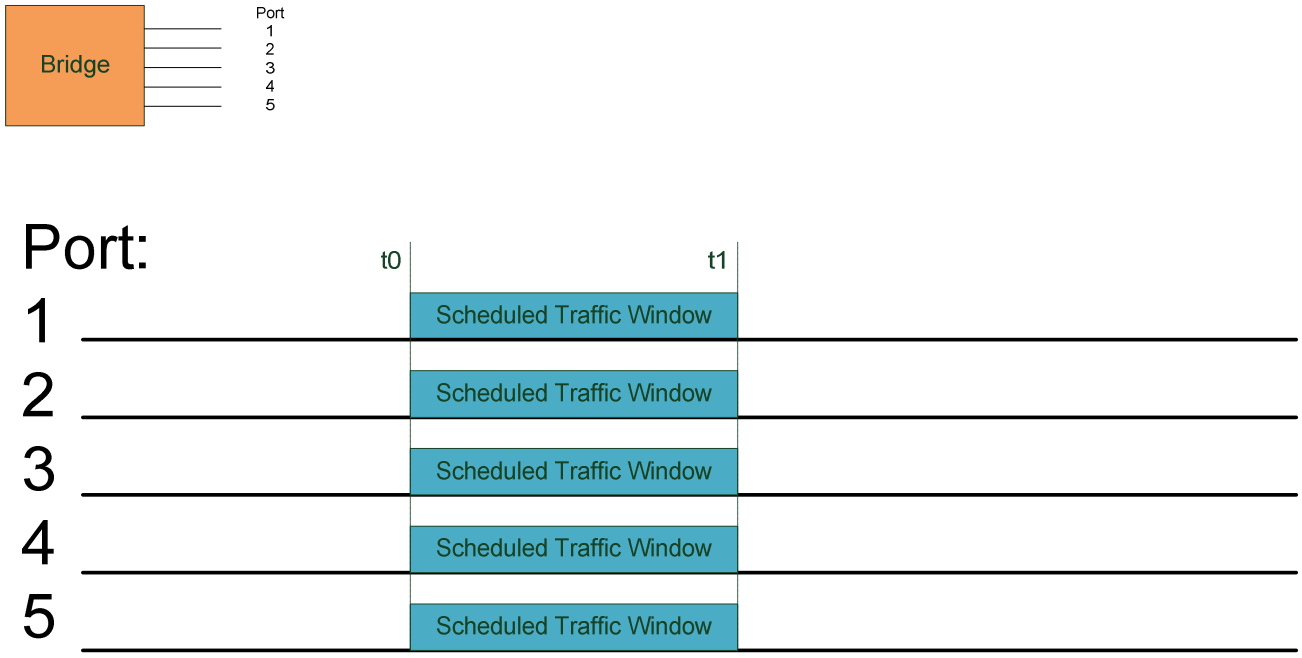
Transmission Windows

- The TAS creates transmission windows for Scheduled Traffic
- One shaper might create more than one window per cycle
- How many windows do we need?

- From the assumptions document:
  - Do we need one window per port or per bridge?
  - Do we need one window per queue per port?
  - Do we need one window interval per port or per bridge?
  - Do we need this per stream? This is currently out of scope for a bridge.

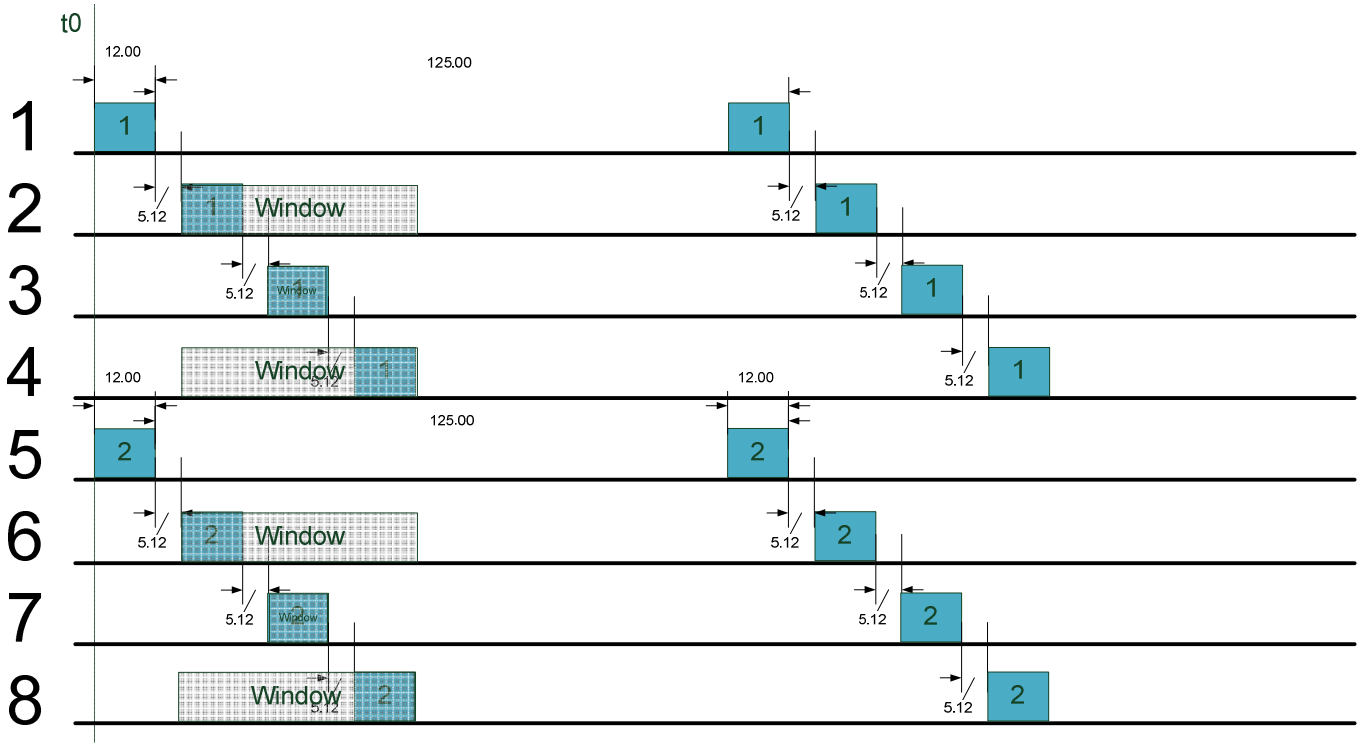
- **One window per stream shouldn't be out of scope for a bridge**

One Window Per Bridge



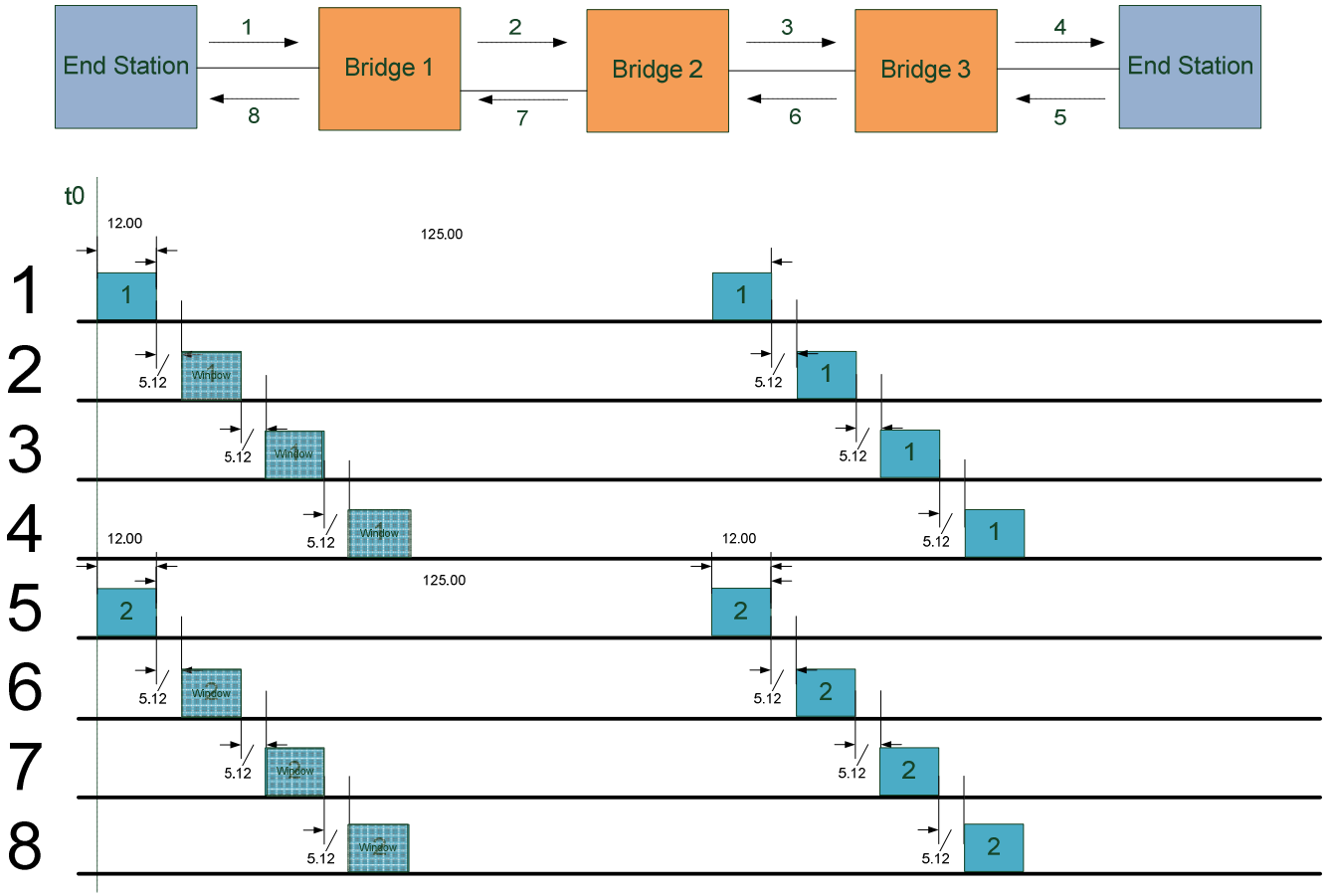
- There is one transmission window per bridge which starts at t0 and ends at t1
- Scheduled Traffic is transmitted inside this window
- A port which transmits Scheduled Traffic has to use this window

One Window Per Bridge – Issues



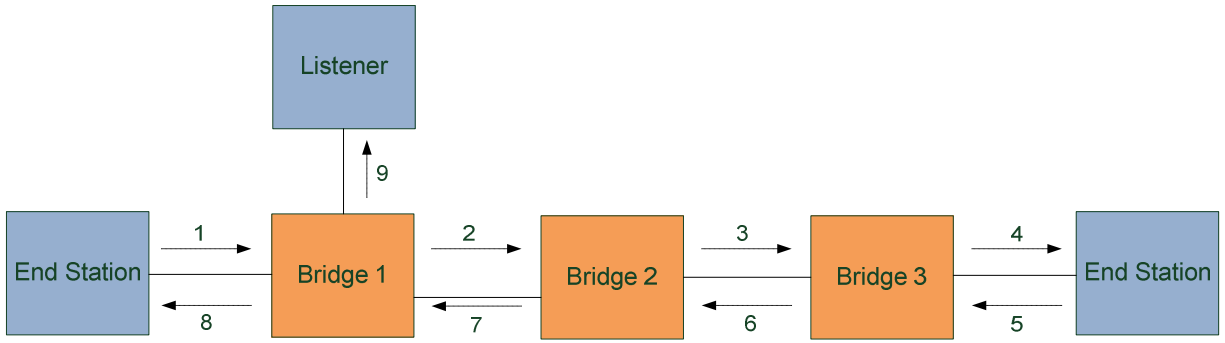
- Only unidirectional communication possible (in the whole network!)
- Not feasible for industrial networks (especially control loops)

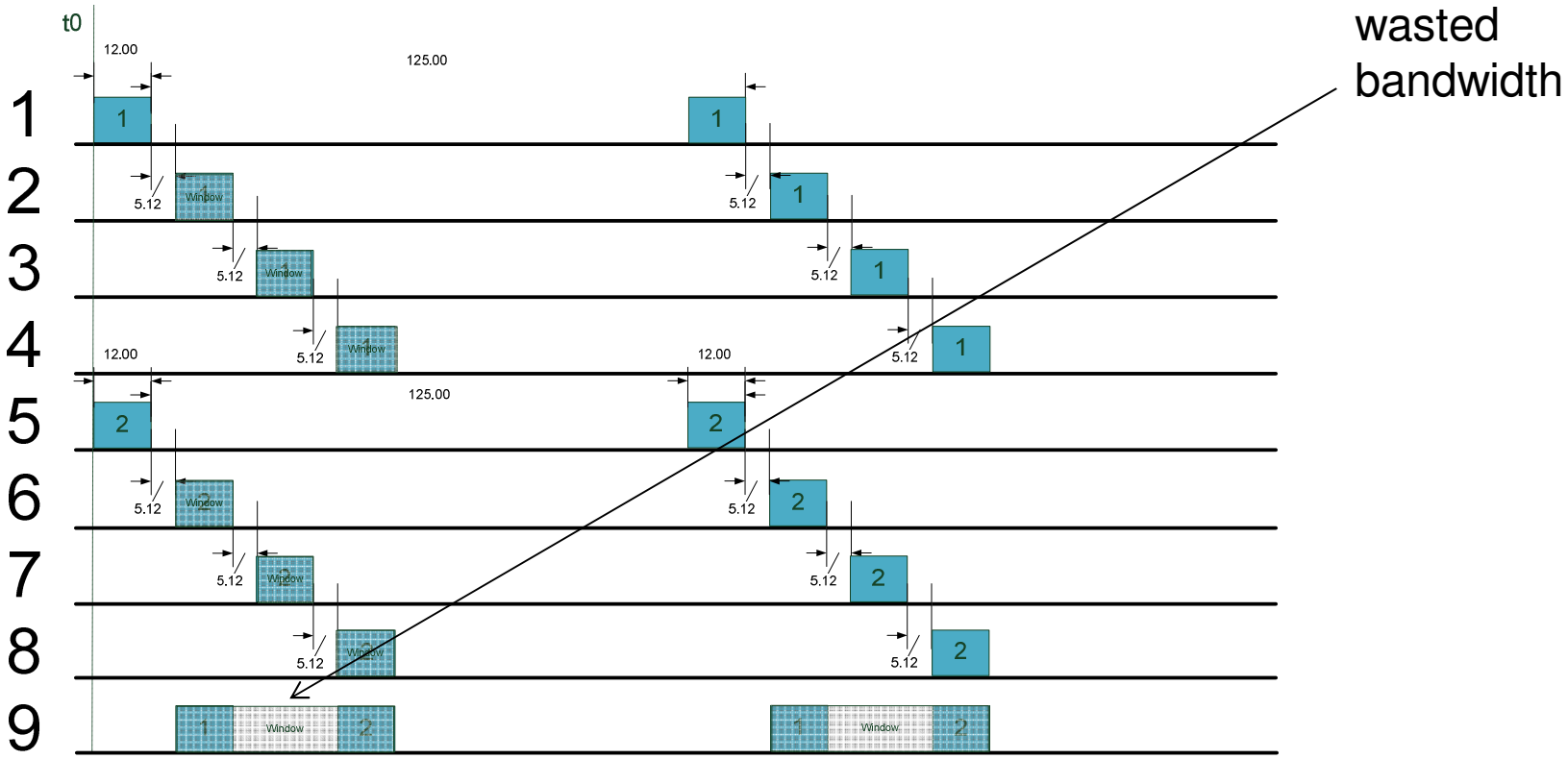
One Window Per Port



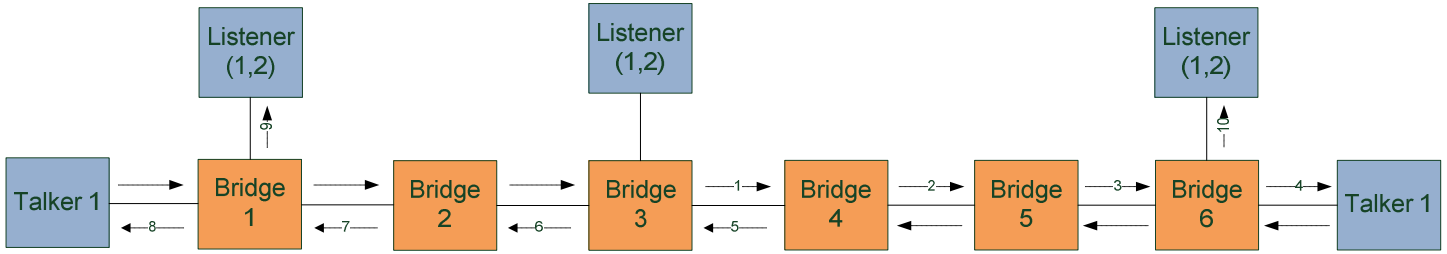
One Window per port would support bidirectional communication

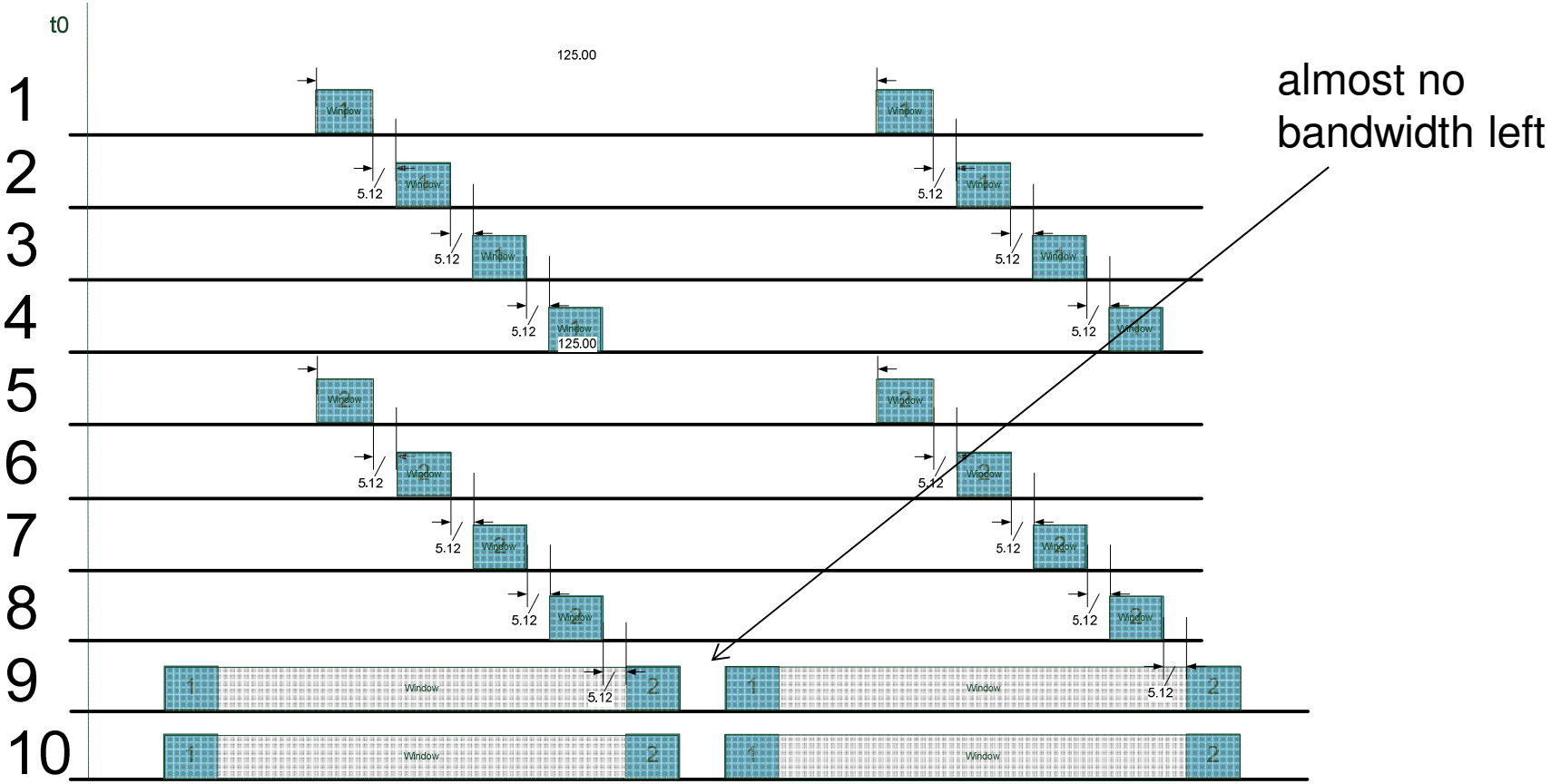
One Window Per Port – Issues (1)



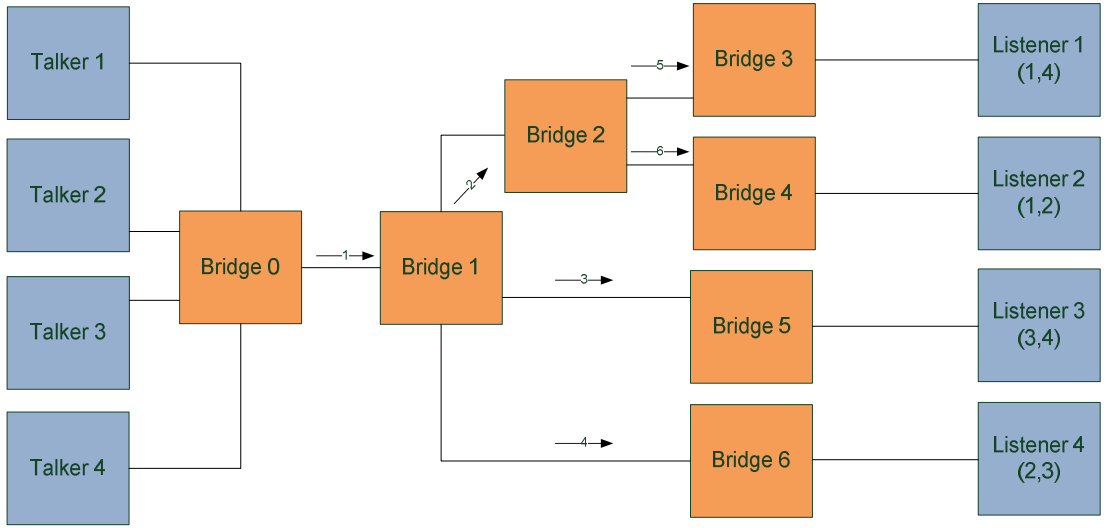


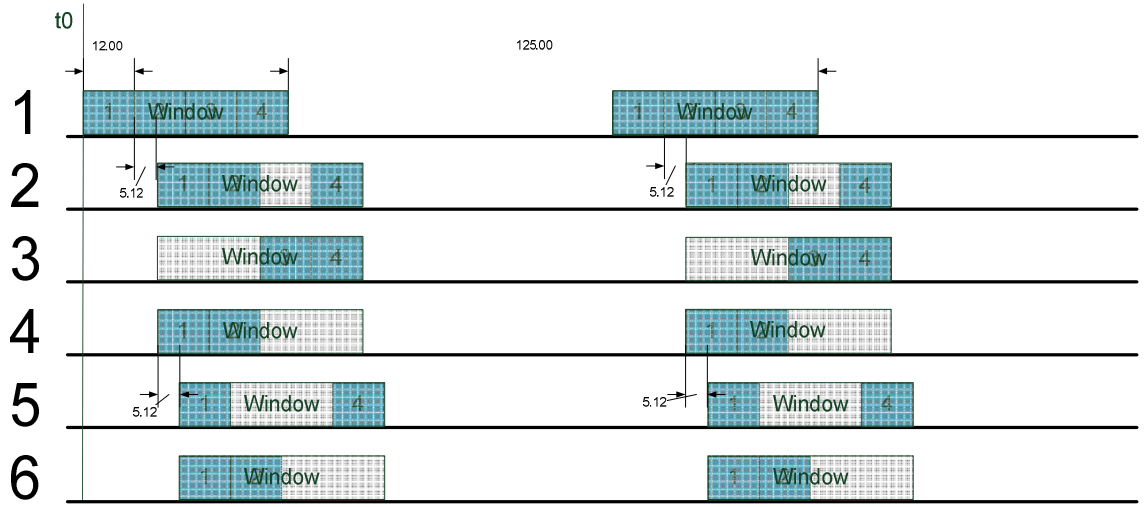
One Window Per Port – Issues (2)





One Window Per Port – Issues (3)





One Window Per Port

- One window per port solves the bidirectional communication issues

- But one window per port causes problems for:
- One-to-many communication relations

  -> Waste of bandwidth
- More complex networks
- Keeping the order inside a time slot

  -> Jitter problem

One Window Per Port Per Class

- The one window per class concept supports more than one class for Scheduled Traffic

  → More than one queue for the Scheduled Traffic frames

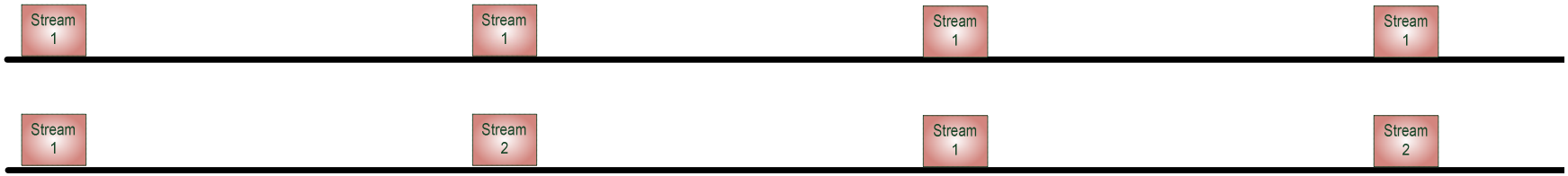
  → More than one TAS for the Scheduled Traffic frames

- This concept doesn't solve the problems of the one window per port concept, but it increases the flexibility, as it is possible to put frames which don't fit in one window in the second class

- As the two windows are completely separate the two classes would deliver the same performance in terms of latency and delivery variation

One Window Per Port Per Stream

- One window per port per stream delivers the highest flexibility and bandwidth efficiency
- And where is the difference between the one window per port and one window per stream concept?



- Where is the difference between one stream with a transmission period of 125µs and two streams with a transmission period of 250µs?

- One would have to make sure that each stream is transmitted in its own slot

Stream Order

- But is this really a difference from in the one window per port concept?

If two streams are in one window

and the same happens at another ingress port

The stream order matters if streams from both ingress ports have to be transmitted on the same egress port

Even with one window per cycle it is necessary to have "small windows" within the big window

Transmission Windows Discussion

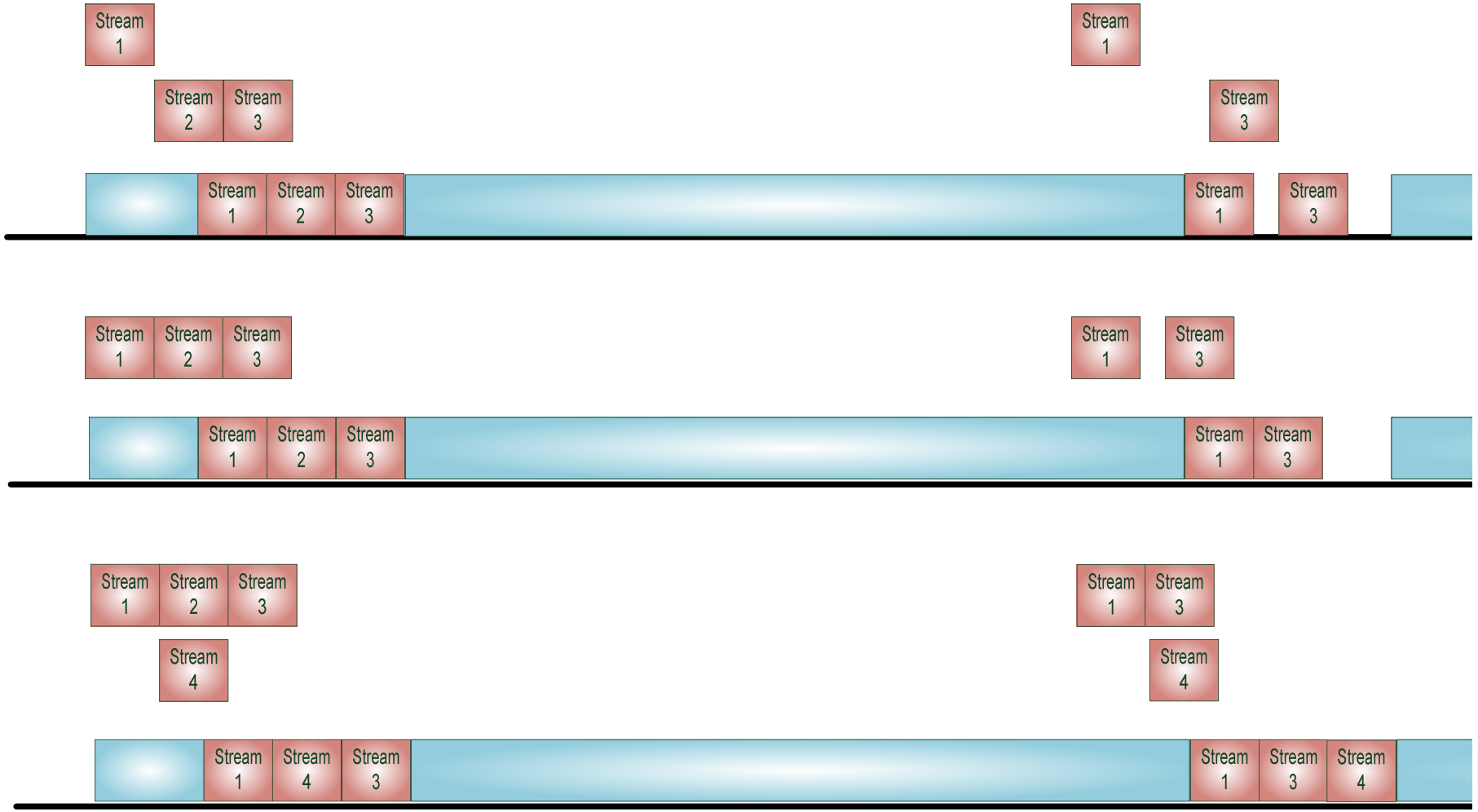
Two results from the discussion yesterday (15.05.2012) seems to be (my interpretation, correct me if I'm wrong):

- The TAS mechanism for Scheduled Traffic should support several windows per cycle

  → One window per stream could be a possible result

- The Scheduled Traffic queues have a TAS in end stations and bridges

Stream Order



Stream Order

- If the streams inside a window don't keep their timeslot, Scheduled Traffic frames can interfere with each other
- Interfering Scheduled Traffic frames cause a higher packet delivery variation
- A higher packet delivery variation breaks the schedule and therefore also the deterministic low latency

It is a major requirement to have a deterministic very low packet delivery variation in order to get a deterministic ultra low latency end to end packet delivery.

This can be only accomplished if the stream order inside a window is fix.

So the problems of a one window per port solution and a one window per stream solution are identical.

How Can We Control The Stream Order?

- All concepts (excluding the one window per stream concept) have the stream order problem
- They only solve the first part of the requirement to achieve really low latency

  → They separate "normal" traffic from Scheduled Traffic
- But they don't prevent the interference of Scheduled Traffic with itself
- The one window per stream concept would theoretically solve this issue but practically the same problem occurs

  → How does the shaper know that the packet in the queue is the right one?

➔ We need a mechanism to protect (enforce) the schedule

One Window Per Port Per Stream

- Assuming we have a mechanism to guarantee the stream order, there is no reason to disallow a one window per stream solution

- This solution would provide:
  - The flexibility to work in small and complex networks
  - Scalability (depending on the engineering efforts one can achieve low or ultra low latency)
  - Support for one-to-one and one-to-many communication relations

Ingress Policing

- Ingress ports discard Scheduled Traffic frames:
  - when no frame is scheduled to arrive at that port (+- delivery variation)
  - which arrive in the wrong time slot (according to SA, DA, ?)

Ingress Policing

But 70 byte @GE = 0.56µs!

➔ It would be not possible to keep the order of small packets from different ports

Egress Policing

- Make the shaper aware of the ingress port associated with the stream (or even the stream DA, SA)

  → The shaper checks the ingress port of a packet (or the stream DA, SA) before the transmission

  → If the ingress port (or stream DA, SA) is not correct the frame is discarded

- A ingress port check could be improved with a frame length check

Egress Policing

- Reordering of packets
- If one packet arrives early and another one late **but in time**, so that the earlier scheduled frame would be last inside the queue, the late frame is inserted in front of the early one

Policing

- Ingress blocking when no Scheduled Traffic is scheduled
- Length check at ingress (less efficient than on egress)
- Stream, SA, DA? or ingress port aware shaper
- Length check (does the frame fit into the time slot)
- Reordering of packets inside the queue

Thank You