Introducing Support for Scheduled Traffic over IEEE Audio Video Bridging Networks

Giuliana Alderisi, Gaetano Patti, Lucia Lo Bello
Department of Electrical, Electronic and Computer Engineering
University of Catania, Italy
{giuliana.alderisi, gaetano.patti, lucia.lobello}@dieei.unict.it

Abstract

The IEEE 802.1 Audio/Video Bridging (AVB) protocol successfully supports audio/video streaming over Ethernet networks and is also a promising candidate for industrial automation. However, several industrial automation applications feature requirements that cannot be fulfilled by the current AVB standard. The next generation of IEEE 802.1 standards, which will specifically target the requirements of automotive and industrial control, is under specification within IEEE 802.1 Working Group. Inspired by such an ongoing work, this paper addresses the introduction in IEEE AVB networks of scheduled traffic, i.e., high priority traffic that is transmitted according to a time schedule so as to ensure no interference from other traffic classes. The paper describes the proposed approach, discusses the design choices and presents a performance assessment based on OMNeT++ simulations.

1. Introduction

The IEEE 802.1 Audio/Video Bridging (AVB) protocol specifications provide for time-synchronized low latency streaming services through IEEE 802 networks. Despite the success of the IEEE AVB standards for audio/video (A/V) streaming, there are applications in the industrial and vehicle control area which feature time sensitive requirements that cannot be fulfilled by the current AVB standards.

The IEEE Time Sensitive Networking task group (formerly named the IEEE Audio Video Bridging task group) of the IEEE 802.1 working group is therefore preparing the next generation of IEEE 802.1 standards, which will specifically target the requirements of automotive and industrial control applications. Two new projects are in progress within the task group. The first one, which goes under the name of project P802.1Qbu, aims to provide support for packet preemption, to allow time-sensitive packets to interrupt a non time-sensitive packet. The second project, named P802.1Qbv, refers to scheduled traffic enhancements. Scheduled traffic is a class of high priority traffic that is transmitted according to a time schedule that ensures no interference from other traffic classes. The P802.1Qbv project aims to define policies for bridges and end stations to schedule the frame transmissions based on timing derived from the IEEE 802.1AS Standard. [1][2].

Inspired by the ongoing activity within the IEEE Time Sensitive Networking task group, this work addresses the introduction of scheduled traffic in IEEE AVB networks. Differently from the current trend in [3], that investigates the inclusion of scheduled traffic in one of the traffic classes already provided by the IEEE AVB standard (i.e., the Stream Reservation Class A), here we propose to introduce a separate class for scheduled traffic, called the Scheduled Traffic (ST) class, that should be added on top of the already existing traffic classes provided by the IEEE AVB standard. Here ST traffic is assumed to be periodic, with a fixed period, and off-line scheduled. ST traffic is served in the highest priority queue according to the priority tag defined in the IEEE 802.1Q [4] and is not handled by credit-based fair queuing (CBFQ), but according to strict priority. Several reasons motivate the design choices made in our approach. First, the time-sensitive control traffic that is the intended candidate for the ST class is typically a priori known, so off-line scheduling is possible. Second, to meet the strict latency requirements of ST traffic, this traffic should be given the highest priority and should not undergo CBFQ, as shaping increases latency. Finally, we envisage a separate class for ST traffic, instead of inserting it in the SR Class A provided by the AVB standard, to avoid any mutual interference between different time-sensitive traffic flows (i.e., control and A/V) within the same priority class. To enforce temporal isolation of the ST class, a system which blocks any non-ST message transmission that would interfere with the upcoming transmission of an ST message is needed. In this paper we adopt Time-Aware Blocking Shaper (TABS), but any other method able to inhibit any transmission that would interfere with ST traffic is equally viable.

The paper is organized as follows. Sect 2 outlines Related Work. Sect. 3 summarizes the IEEE AVB features. Sect. 4 introduces our system model, while Sect. 5 addresses the performance of our approach obtained through OMNeT++ simulations. Finally, Sect. 6 concludes the paper and outlines future work.

2. Related Work

Many recent works in the literature addressed the real-time performance of IEEE AVB in multiple automation domains, namely, automotive, aeronautics, and industrial automation. The work [5] indicates AVB as one of the possible candidates for automotive communications. Encouraging simulation results obtained with the IEEE 802.1AS standard are provided in [6] and [7]. The IEEE AVB suitability for supporting the traffic flows of both Advanced Driver Assistance Systems (ADAS) and multimedia/infotainment systems was proven in [8], [9] and [10]. IEEE AVB has the potential to be used not only as a common networking technology within a single functional domain, but also as an in-car backbone network for inter domain communication [11] through suitable gateways interconnecting the heterogeneous networks used in the different functional domains [12]. The work in [13] addressed the performance of AVB in aeronautic networks and showed that, when using time synchronization, the timing requirements are met.

A number of works focused on the Ethernet AVB ability to cope with the requirements of the real-time traffic typically found in industrial automation [14]. In [15] a performance comparison between AVB and standard Ethernet is presented. The outcome of the study is that, although AVB allows for determining the worst case latency for all real-time message classes, further improvements are still needed for use in industrial automation. In fact, as the CBFQ used in AVB adopts non-preemptive scheduling, in the worst case a real-time frame might be delayed in every bridge by the ongoing transmission of a maximum sized frame not belonging to the real-time class. Approaches to mitigate this interference, such as packet preemption, fragmentation, and synchronous scheduling [16], are discussed in [17].

The work [18] addresses AVB worst case latency analysis, pointing out some limitations of current theoretical formulations used for AVB latency estimation in [19]. Starting from the findings in [20], the paper describes two effects that affect the latency estimation of AVB, that are not encompassed in the formulas provided by the AVB standard [19]. The first effect is the so-called "own-priority and higher-priority blocking", that occurs when several streams share the same port. In this case, bursts can accumulate over multiple hops, thus eventually interfering with other streams and increasing their latency. The second effect, called "shaper blocking", refers instead to the large blocking times that a flow may experience in certain scenarios (i.e., in daisy-chain topologies) due to traffic shaping. This effect may get worse when combined with the priority blocking described above. The outcomes of the above mentioned works are confirmed by the simulation results obtained in [10] and [9] for small-size navigation warnings traffic when this traffic is served in the same class of large-size ADAS traffic streams.

Within the IEEE 802.1 Working Group, TABSs (Time-Aware Blocking Shapers) or time windows were proposed in [3] and [21] for isolating Class A streams from the interference due to other traffic types. The feasibility of TABS is described in [3]. As AVB bridges are already time-aware thanks to the IEEE 802.1AS, the changes required to implement TABSs are mechanism able to inhibit/allow transmissions in a time interval and to configure the shapers via a management information base. In our approach, TABS are implemented both in the bridges and in the sender nodes.

The approaches proposed in [3] and in [21] map all the time-sensitive flows on the same class (Class A) irrespective of their heterogeneous sizes and time constraints. As explained before, such a choice is not beneficial to low latency, small-size traffic, which should not be handled in the same queue as large A/V frames. What we propose here is adding a separate class on top of the AVB Stream Reservation Classes A and B to introduce support for scheduled traffic, while maintaining the other traffic classes provided by the AVB standard. We expect that in this way the requirements of the different types of real-time flows typically found in automation domains can be met. Some similarity exists between the approach proposed in this paper to support scheduled traffic and the one adopted for the time-triggered class in Time-Triggered Ethernet (TTE) [22]; however, as TTE works very differently from AVB, the AVB version with the ST class that is proposed in this paper is quite different from TTE. A performance comparison between standard AVB, the AVB version proposed in this paper called AVB_ST, and TTE is addressed in Sect. 5.

3. IEEE 802.1 Audio Video Bridging

The IEEE 802.1 Audio Video Bridging (AVB) standard consists of four separate documents:

- IEEE 802.1AS [23], which provides for precise timing to support low-jitter clocks and accurate synchronization of multiple streams.
- IEEE 802.1Qat [24], which specifies mechanisms to reserve resources within bridges (buffers, queues) along the path between sender and receiver.
- IEEE 802.1Qav [25], which specifies queuing and forwarding rules.
- IEEE 802.1BA [19] Audio Video Bridging systems, which define the AVB profiles.

Two kinds of devices are found in an AVB network: end stations, that transmit (Talkers) or receive (Listeners) traffic, and bridges, that forward traffic and perform bandwidth reservation.

3.1. Stream Reservation Protocol (SRP)

A Stream Reservation (SR) Class is defined as a traffic class that is forwarded on the network according to the procedures described in the Multiple Stream Reservation Protocol (MSRP). Currently, only two SR traffic classes, namely, Class A and Class B, are supported, as the specifications in [25] define the class measurement interval parameter for these two traffic classes only. The class measurement interval is a period of time during which a station can place up to MaxIntervalFrames data frames, of a size no longer than MaxFrameSize each, into the queue associated to the stream. The MaxIntervalFrames and MaxFrameSize parameters are used by the MSRP. The first is defined as a frame rate, in a frames-per-class measurement interval, for a given stream, while the second is the maximum number of bits per every frame of the same stream. The class measurement interval value is 125 μs for Class A and 250 μs for Class B. For these two traffic classes the AVB standard guarantees a maximum end-to-end delay, for seven hops, equal to 2 ms for Class A and 50 ms for Class B. The SR classes are handled by the CBFQ.

The SRP according to [24], foresees that each traffic flow of SR Class A and SR Class B, before being sent on the network, must be registered on the bridges that will forward it on the network to the listener. The registration is possible if, and only if, every node that participates in the forwarding process of the stream, from the talker to the listener, has sufficient bandwidth and resources. If the registration is successfully completed, the amount of bandwidth required by the new flow is subtracted, at each port that is traversed by this flow, from the total amount of bandwidth available. Once the percentage of bandwidth to be reserved to the Classes A and B is determined, the remaining one is left over for best effort traffic. Three of the parameters that are involved in these procedures are:

- portTransmitRate – PTR: The transmission rate, in bits/s, that the underlying MAC service provides.
- adminIdleSlope(N): The bandwidth, in bit/s, that has been requested to be reserved for use by the queue associated with traffic class N.
- deltaBandwidth(N): A percentage of the PTR; this is the bandwidth that can be reserved for use by the queue associated with traffic class N.

IEEE 802.1Qav [25] defines eight traffic classes and foresees that at least one of them must be used as SR class. Traffic classes that are not used as SR Classes are left to best effort traffic, without any bandwidth reservation or guarantee. Each traffic class has a priority level (from 0 to 7, where 7 is the highest priority).

3.2. Synchronization and 802.1AS

Clock synchronization is performed as described in the 802.1AS standard [23]. This protocol is a variation of the IEEE 1588 [26] standard and was updated to the latest version in 2011.

To synchronize the network according to IEEE 802.1AS, all the nodes, bridges and end-station, are required to be time-aware stations, where a time-aware station is a system that makes explicit reference to time. Among the nodes, one is selected as a reference node for synchronization, according to the Best Master Clock Algorithm (BMCA). This node is called the grandmaster node. The BMCA determines the grandmaster and constructs a time-synchronization spanning tree rooted at the grandmaster. The synchronized time is transported from the grandmaster to other time-aware systems via the time-synchronization spanning tree. In the IEEE 802.1AS standard procedures for synchronization and syntonization are described[1]. The time synchronization capability of the IEEE 802.1AS for industrial applications derives from the IEEE 1588 protocol [26]. The performance of the IEEE 802.1AS protocol was already investigated for industrial automation [27] and, more specifically, for automotive scenarios [6]. In addition, in [28] it is shown that the IEEE 802.1AS can provide the same clock quality as FlexRay.

[1]Two stations are synchronized if their clocks do not differ in time, while they are syntonized if they use, for all the time interval measurements, the same time base

4. System Model

4.1. Scheduled Traffic Class (ST Class)

Here we introduce a type of time-sensitive control traffic, called Scheduled Traffic (ST). By definition, ST traffic flows are periodic and the characteristics of this traffic (period, frame size) are fixed and a priori known. The transmission sequence for ST messages is off-line planned by a scheduler and offset scheduling techniques [29] are adopted in the planning phase to make sure, by design, that conflicts between transmissions of ST messages will not occur in the entire network, i.e. at end stations as bridges. The expected receive times of the ST messages are computed off-line, as will be explained in Subsect. 4.2. All the nodes that are either traversed by ST flows or are the intended recipient of ST flows, i.e. either bridges or listeners, are fed with the schedule of the relevant flows. As mentioned before, the TABS mechanism here adopted, that will be detailed in the following subsections, avoids any interference on ST messages from other traffic classes. Therefore also the expected receiving time of any ST message can be calculated on the receiving side, albeit with some uncertainty depending on the clock synchronization protocol. For this reason, suitable time intervals, here called ST_Windows, are defined and Subsect. 4.2 discusses the ST Windows sizing. The support of the ST Class according to our approach requires a reliable synchronization of the network nodes, such that every node knows when it is the right time to transmit its ST messages. As mentioned in Sect. 2, in AVB clock synchronization is performed by the IEEE 802.1AS protocol, whose performance for a switched Ethernet network were evaluated in [6]. In this paper some assumptions, taken from [6], are made:

- Fast Ethernet 100 Mbit/s full-duplex links.
- Switch processing time: 10 μs.
- Clock speed: 40 ns tick interval at least, or 25 MHz clock frequency.

Under these assumptions, it was shown in [6] that the synchronization process based on the IEEE 802.1AS is not affected by high network workload and that the synchronization error and its accuracy remain below the specified value of 1 µs over seven or fewer hops. The commonly suggested value for the synchronization interval is 125 ms. However, for the purposes of our approach, we opt for a shorter sync interval of 62.5 ms that, as suggested in [6], reduces the maximum synchronization error to approximately 50% and only requires 0.027 Mbit/s per link to exchange synchronization messages. In our design, we therefore rely on the support provided by the IEEE 802.1AS as far as clock synchronization and the notion of global time are concerned. Thanks to the syntonization, the time base used in the entire network is the one of the grandmaster and is the same for every node. ST messages are tagged with the highest priority TAG according to the IEEE 802.1Q standard. SR Classes A and B take the second and the third highest priority, respectively. The TAG, which is part of the IEEE 802.1Q header, has values ranging from 0 to 7. In our approach, the mapping of the priority TAG on each traffic class (i.e., ST, SR Class A, SR Class B or best effort) is known to all the devices that belong to the network (both end-station and bridges). ST traffic is handled in a separate queue and does not undergo traffic shaping. On the contrary, SR Classes are handled by CBFQ.

4.2. Sizing the ST_Windows

The communication mechanism foreseen in our approach avoids any interference on ST messages from other messages belonging either to the same class or to any other traffic class. As a result, the receiving time of any ST message can be calculated. However, the calculation of the reception instant for any ST message has to take into account the synchronization error between the nodes and the drift of each station. This drift is expressed in parts per million (ppm) and represents the clock speed. The difference between the drift of a node and the drift of the reference node can be different from node to node. According to the drift, during a synchronization interval, i.e., between a synchronization instant and the next one, the local clock of each node deviates from the reference clock. The difference between two clock values measured at the same instant of time is called skew.

On each receiving side (the bridge or the listener) for each ST message, a time window within which the message has to be received, called an ST_Window, is defined. If the ST message arrives outside this window, it is considered lost. The ST_Window is centered on the expected receiving time, called expectedReceiveTime, and is equal to twice the worst deviation (here indicated with Δ) between the local clock of the receiver and the local clock of the sender, as in (1):

$$\text{ST\_windows} = 2\times \Delta \qquad (1)$$

This is because two time-aware systems cannot differ in time by more than $2\times\Delta$, as this is the maximum time difference between two local clocks of two different nodes. Δ depends on the synchronization protocol used and on the accuracy of the clock, it is computed as in (2):

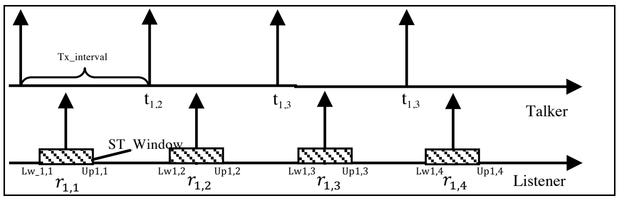

Fig.1 -ST_Windows

$$\Delta = \text{max\_sync\_error} + \text{max\_skew} \qquad (2)$$

where max_sync_error is the synchronization error (1µs for IEEE 802.1AS) and max_skew is the maximum possible skew between two consecutive nodes, that depends on the difference between their drifts and is computed after an entire synchronization interval. Fig. 1 shows the message transmission sequence for an ST traffic stream. In the figure, $t_{i,j}$ represents the send time of the j-th message of the i-th stream from the talker. Transmission occurs every TxInterval and $r_{i,j}$ is the expected receive time of the j-th message of the i-th stream (sent at $t_{i,j}$), calculated as:

$$\text{expectedReceiveTime}_{i,j} = r_{i,j} = \text{TxTime} + t_{i,j} \qquad (3)$$

where TxTime is the time needed to transmit the message from the sender to the receiver[2]. TxTime depends on the number of hops for the stream and is calculated similary to [3], as (4):

$$\text{TxTime} = \text{TxTime}_{\text{Talker}} + \sum_{n=0}^{\text{BridgeHops}} \text{TxTime}_{\text{Bridge\_n}} \qquad (4)$$

Where $\text{TxTime}_{\text{Talker}}$ is the time needed to transmit one frame out of the Talker, given by:

$$\text{TxTime}_{\text{Talker}} = (\text{Payload}_i + \text{Overhead} + \text{IFG}) \times 8 \times 10 \qquad (5)$$

where: Payload is the payload of each frame of the i-th stream, expressed in bytes; Overhead represents the MAC layer overhead added to each IEEE 802.1Q frame (22 bytes); IFG is the inter-frame gap (according to the Ethernet standard recommendation, 12 bytes); 10 is the transmission time for one bit on Fast Ethernet, expressed in nanoseconds.

In Eq.(4), BridgeHops is the number of bridges that a frame has to cross to reach the Listener. $\text{TxTime}_{\text{Bridge\_n}}$ is the time a frame takes to traverse the n-th bridge, i.e. (6)

$$\text{TxTime}_{\text{Bridge\_n}} = t_{\text{BRIDGE}} + t_{\text{TX}} \qquad (6)$$

where:

- t_{BRIDGE} is the time to process the frame within the bridge. We recall that here, by design, ST traffic does not wait in the bridge queue. This is because ST traffic is properly staggered through off-line scheduling, so as to not pile up in the queues. Moreover, ST traffic does not wait for other traffic classes thanks to TABS. As a result, t_{BRIDGE} is a constant that depends on the bridge specifications. Here we assume Store and Forward bridges and a t_{BRIDGE} value of 10 µs.
- t_{TX} is the time needed to transmit the frame of j-th stream out of the bridge, computed as in (7)

$$t_{\text{TX}} = (\text{Payload}_i + \text{Overhead} + \text{IFG}) \times 8 \times 10 \qquad (7)$$

To assess whether the j-th message of the i-th stream has been entirely received within its ST_Window, the following values are observed for each j-th message of the i-th stream.

- UpperSTWindows$_{i,j}$ (Up$_{i,j}$ in Fig. 2), i.e., the upper value of the ST_Window
- lowerSTWindows$_{i,j}$, (Lw$_{i,j}$ in Fig. 2), i.e., the lower value of the ST_Window.

These two values are calculated as:

$$Up_{i,j} = expectedReceivedTime + ST\_Windows/2 \quad (8)$$

$$Lw_{i,j} = expectedReceivedTime - ST\_Windows/2 \quad (9)$$

If the message is delivered out of its ST_Window, it is considered invalid and is discarded.

4.3. AVB with scheduled traffic support: Outline

To enforce temporal isolation for the ST Class, thus preventing any non-ST message transmission that could delay the transmission of the next ST message, all the non-ST traffic goes through a TABS both in the bridges and in the talkers. The idea of TABS was proposed in [3], to preserve the QoS of SR Class A traffic in AVB networks. In [3] the TABS blocks any lower priority transmission (i.e., from SR Class B or non-AVB traffic) that would interfere with the upcoming transmission of SR Class A traffic. For instance, if a given non-SR Class A queue has a frame ready, but the transmission of such a frame, if allowed, would delay the start time of the next transmission of SR Class A traffic, such a transmission is not allowed.

In our approach, we instead propose to adopt TABSs to provide temporal isolation between the ST traffic and all the other traffic types. Thanks to the TABS mechanism, any non-ST message that is enqueued, ready for transmission, has to wait not only for the duration of an ST message transmission, but also for an additional time if the time needed for its transmission is longer than the difference between the send time of the next ST message and the current time.

This time difference, called a minimal distance, is enforced to avoid the transmission of non-ST traffic that would delay the next ST message. This way, an ST message ready for transmission will never experience blocking due to the ongoing transmission of a non-ST message, as the transmission of a non-ST message that could delay the ST message will not be allowed. In the AVB_ST approach, therefore, there is no need for preempting the ongoing transmission of a non-ST message, as if such a message is being transmitted, this means that it will not interfere with any ST message, otherwise it would have been blocked in advance.

To avoid bandwidth waste, in [3] it is suggested to make the TABS inspect the next priority queue and see whether such a queue has a smaller frame whose transmission would finish before the start time of the next Class A transmission. If this is the case, this frame is allowed to start. In our AVB_ST approach, therefore, in the time interval between two consecutive ST message send times, the queues of non-ST traffic, starting from the nonempty highest priority one, are looked for a ready message whose transmission would finish before the send time of the next ST message. If the ready message in the highest nonempty priority queue (e.g., SR Class A) requires a transmission time greater than the time interval up to the next send time for ST traffic, the message is not selected for transmission, and the next highest priority queue is inspected (e.g. SR Class B). In the end, either a suitable message is found or no transmission is allowed. The maximum time taken by the search is the time needed to perform a comparison between two different time values multiplied by the queues number. Such a number is negligible, as the time difference computation is made "on the fly", and concerns the propagation time in logic gates. The TABS mechanism represents a tradeoff between the need to temporally isolate ST traffic, avoid any blocking due to lower priority traffic, and bandwidth utilization.

In the AVB_ST approach, therefore, the messages in the queues associated to the SR Classes undergo both the TABS and credit shaping, while best effort messages go through the TABS only, as no shaping is foreseen for best effort traffic in the standard [25]. In our design, shaping for SR Classes is performed after the TABS, so that credits are consumed only when the frame transmission is allowed by the TABS.

Summarizing, in our design offset based scheduling prevents ST traffic from experiencing queuing delays due to other ST traffic, TABSs avoid interference from the other traffic classes and strict priority handling for ST traffic avoids the additional delays due to CBS. This design makes the ST traffic latency both low and predictable.

4.4. SRP with Scheduled Traffic

In AVB_ST the bandwidth reservation has to be accomplished taking into account the bandwidth reserved to ST streams. In addition, the computation of the correct adminIdleSlope for each traffic class(as seen in Subsect. 3.1) has also to consider the guard band, i.e., the bandwidth not utilized when the enforcement of the minimal distance (as seen in Subsect. 4.3) does not allow any transmission between two consecutive ST send times. For this reason, a new parameter for the SRP, here called realPortTransmitRate (RPTR), has to be defined, as follows:

$$RPTR = PTR - adminIdleSlope(ST) \quad (10)$$

where PTR is defined as in Subsect. 3.1 and adminIdleSlope(ST) is the bandwidth reserved to ST traffic, which is calculated off-line. The RPTR is therefore the bandwidth that is left to SR Classes and best effort traffic. The bandwidth reserved to SR Classes is defined by the adminIdleSlope(N) parameter. The adminIdleSlope for a specific SR stream is obtained as in the standard [25], with the difference being that the new RPTR parameter is used instead of the RPTR. However, this new definition of the transmission rate does not take

[2]Eq. (3) does not include cable delay, as it is negligible over distances shorter than 100 meters.

$node_1$	$node_2$	$node_3$	$node_4$	$node_5$	$node_6$	$Bridge_1$	$Bridge_2$	$Bridge_3$	$Bridge_4$
-30	20	10	-15	-50	0	30	-15	40	50

Table I – Drift difference between time-aware systems and the reference node[ppm]

into account the guard band mechanism.

The bandwidth not utilized due to the guard band depends on the number of ST transmissions and on the size of the maximum non-ST frame that may interfere with these ST transmissions. A simple way to calculate the guard band is to make a worst case assumption, i.e., considering that the interfering frame has always the maximum Ethernet frame length, but this assumption is overly pessimistic. For this reason, here a more realistic method for dynamically computing the guard band is proposed, that uses the largest frame between all the SR Class flows that are registered at the node. The mechanism works as follows. At the time of registration attempt of an SR Class A or SR Class B stream at a node, the bandwidth availability for the specific traffic class is checked. If there is bandwidth available then the frame size for the specific flow is examined. If this size exceeds the maximum frame size of all the other SR flows already registered at the relevant node, then the ratios between such a size and the period of each ST stream are computed, as for each ST message a guard band to prevent the largest SR Class frame from interfering is needed. The sum of the values thus obtained represents the maximum guard band needed to isolate the ST traffic of that specific node. This estimate is still pessimistic, but safe. If the guard band amount exceeds the given threshold, then the SR stream cannot be registered. We heuristically set a threshold equal to 95% of the bandwidth left over for best effort traffic to leave room for network traffic such as, synchronization messages, that, as calculated in [6], require 3% of the network bandwidth to realize a synchronization period of 62.5 The same admission control mechanism is repeated every time a new AVB SR flow enters the registration procedure.

Summarizing, in AVB_ST, for each node, the PTR of each port is initially reduced by the bandwidth reserved to ST traffic (that is known a priori, thanks to the off-line scheduling). The remaining bandwidth is subsequently distributed between the SR Classes, according to the deltaBandwidth(N) (seen in Sect. 3.1) parameter. What remains is left for both best effort traffic and the guard band. If the guard band reaches 95% of the remaining bandwidth, then it will not be possible to register any SR traffic flow with a frame size larger than the one used to compute the current guard band.

5. Performance Evaluation

This Section presents a proof-of-concept assessment of the AVB version proposed in this paper, also in comparison with standard AVB and TTEthernet (TTE). Simulations were run using the OMNeT++ [30] simulation tool and the INET-Framework. The simulated protocol stacks feature the Application level on top of Ethernet, therefore the overhead is 22 bytes for both the AVB versions and 18 bytes for TTE, respectively. Three simulations, called AVB_ST, AVB and TTE were performed. Each simulation was 600 s long so as to collect a large number of samples, and all the flows started at the same time to assess a critical scenario from the interference point of view. Our performance metrics

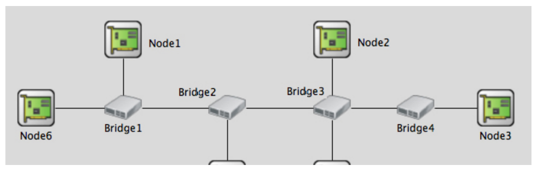

Fig. 2 –Network Topology

definitions are:

- Latency: the one-way end-to-end frame delay, i.e., the time from the source sending a packet to the destination receiving it.
- Absolute Jitter: the difference between the maximum and the minimum latency value for a given flow.

5.1. Topology and Traffic Model

Our simulation scenario consists of 6 nodes and 4 bridges B_j, with j=1..4. The network topology is shown in Fig.2. There are 5 talkers, called $node_k$ with k=1..5,and one listener, i.e., $node_6$. Every talker sends frames to $node_6$, this because the purpose of this simulations is to load as much as possible the queues of the Bridge1 and generate a sort of bottleneck to evaluate the interfering problems. Table II shows the traffic model and the mapping of the traffic flows on the available traffic classes for each simulation. All the flows are time-sensitive, but with some differences, as the ones generated by $node_k$, with k=1..4, are assumed to be more time-critical than the flow generated by node 5. In particular, the flows generated by $node_3$ and $node_4$ and labeled as ST are scheduled traffic, while the non-ST flows are indicated as NST in Table II. The second column of Table II gives the Application level payload. All the flows in Table II are periodic, with period equal to the service rate (shown in the third column), and have constant payload. The switch processing time is equal to 10 μs.

5.2. AVB_ST Simulation

Stations			Max_skew(μs)	Δ(µs)
$Node_4$	↔	$Bridge_3$	3.43	$\Delta_{Bridge3}$= (1+3.43)
$Node_3$	↔	$Bridge_4$	4.062	$\Delta_{Bridge4}$= (1+4.06)
$Bridge_4$	↔	$Bridge_3$	0.625	$\Delta_{Bridge4}$= (1+0.63)
$Bridge_3$	↔	$Bridge_2$	3.437	$\Delta_{Bridge4}$= (1+3.44)
$Bridge_2$	↔	$Bridge_1$	2.812	$\Delta_{Bridge4}$= (1+2.81)
$Bridge_1$	↔	$Node_6$	1.875	$\Delta_{Bridge4}$= (1+1.87)

Table III - Maximum skew and Δ values

In this simulation, which refers to the AVB version with scheduled traffic, the priority value assigned to the ST class is 7, as this flow is given the highest priority, while the values for the SR Class A and B are 6 and 5, respectively. Following the approach proposed in this work, the bandwidth that is left to the SR Classes and best effort traffic is the difference between the total available bandwidth and the sum of the bandwidth needs of the ST traffic, i.e., 4.096 Mbit/s in our case, plus the bandwidth required to support the 62.5 ms synchronization interval, i.e., about 3 Mbit/s according to [6]. The 75% of the remaining bandwidth, similar to what is foreseen in the IEEE 802.1Qat standard [24], is reserved to SR Class A and B, while the other 25% is left for best effort traffic and the guard band. As a result, the overall bandwidth that SR Classes can be reserved is 69.7 Mbit/s, while 23 Mbit/s are left for best effort traffic and guard band. As much as 40% of the bandwidth available for the SR Class is reserved to Class A, while the remaining 35% is reserved to Class B. ST messages are sent at regular time intervals and the cycle time, i.e., the least common multiple of their periods, is equal to 500 μs in our case. The ST_window is sized as 2×Δ. This value is off-line calculated for all the nodes and bridges traversed by ST frames (i.e., B_4, B_3, B_2, B_1, and $node_6$) using equation (2), where the maximum synchronization error (max_sync_error), according to the findings in [6], is 1μs and the maximum skew (max_skew) is obtained from the drift values in Table I applying the formula (11):

$$\text{max\_skew} = \text{drift\_difference}/\text{sync\_interval} \quad (11)$$

Table III shows the maximum skews between consecutive nodes and the Δ values for each station crossed by an ST flow, obtained from (2).

The drift is considered a fixed value because, as show in [31], after each synchronization the skew starts growing again but, due to the very small drift variations, it grows in a quasi-linear way.

Each ST_Window is centered on the expectedReceiveTime instant obtained from equation (3). Here the t_{BRIDGE} is assumed equal to 10 µs, as in [6], and the time to transmit one frame out of the bridge or of the Talker node is calculated according to the frame size, also considering the 22 bytes overhead per frame. Time checks on the minimal distance are used to implement TABS. From the stream model in Table II, it is possible to calculate the amount of bandwidth reserved for the guard band, that depends on both the transmission period of the scheduled traffic (500 μs), and the maximum frame size among all the registered streams (256 bytes for AVB SR Class flows). We calculate the guard band, as in (12)

$$\text{guard band} = 2 * \frac{(256+22)*8}{500*10^{-6}} = 8.192\ \text{Mbit/s} \quad (12)$$

As there are two different nodes that send ST traffic, offset scheduling is applied to ensure that the ST_Windows of the two ST flows at the common bridge they traverse, i.e. B_3, are one after the other, to avoid bandwidth waste. We accomplish this adding a suitable offset (22 μs in our case) to the ST flow generated by $node_4$.

Talker/ Type	Latency(µs)						Absolute Jitter(µs)		
	AVBST		AVB		TTE		AVBST	AVB	TTE
	Mean	*Max*	*Mean*	*Max*	*Mean*	*Max*			
$Node_1$	55	55	55	55	51	53	0	0	6
$Node_2$	121	121	132	145	110	115	0	24	10
$Node_3$NST	154	187	159	174	154	157	32	14	8
$Node_3$ST	103	103	141	165	122	122	0	30	0
$Node_4$NST	145	145	121	121	143	150	0	0	10
$Node_4$ST	89	89	149	149	100	100	0	0	0
$Node_5$	90	95	94	94	97	100	7	0	6

Table IV - Mean and Maximum latency and absolute jitter, for each simulation

5.3. AVB Simulation

This simulation refers to the AVB standard. Here, the time-sensitive traffic is mapped on the Stream Reservation classes, as shown in Table II. SR Classes A and B are given priority 7 and 6, respectively. Following the SRP, 75% of the total bandwidth was reserved for SR traffic (40% for Class A and 35% for Class B). All the SR traffic is handled by CBFQ.

5.4. TTE Simulation

TTE [22], SAE standard (AS6802), supports three different traffic types: time-triggered (TT), rate constrained (RC) and best effort (BE). Table II shows the mapping used in our TTE simulations. Scheduled traffic was mapped on the TT class and was given the highest priority, i.e. level 1. The other streams were mapped on the RC class with level 2 and 3, respectively. Strict Priority scheduling is used between the switch queues, FIFO within the same queue.

Talker Type	Payload (byte)	Service rate (μs)	Traffic Classes AVB_ST	Traffic Classes AVB	Traffic Classes TTE
$Node_1$	256	250	SR A (prio 6)	SR A (prio 7)	RC (prio2)
$Node_2$	256	250	SR A (prio 6)	SR A (prio 7)	RC (prio2)
$Node_3$NST	256	300	SR A (prio 6)	SR A (prio 7)	RC (prio2)
$Node_3$ST	128	500	ST (prio 7)	SR A (prio 7)	TT (prio1)
$Node_4$NST	256	500	SR A (prio 6)	SR A (prio 7)	RC (prio2)
$Node_4$ST	128	500	ST (prio 7)	SR A (prio 7)	TT (prio1)
$Node_5$	256	500	SR B (prio 5)	SR B (prio 6)	RC (prio3)

Table II - Traffic model and configured traffic classes

5.5. Results

Table IV shows that the simulation results for the ST traffic flows obtained by AVB_ST outperform the ones obtained for the same traffic by standard AVB. In particular, for ST traffic flows we not only observe a lower latency with AVB_ST than with standard AVB, but we also see that the mean and maximum latency with our approach are equal, thus the jitter is zero. This is because, in our approach, ST traffic does not experience queuing delays, thanks to offset scheduling and TABS, is provided with preferential service while crossing the network and does not undergo shaping, so the latency is low and predictable. Conversely, with standard AVB, the ST originated from $node_3$ experiences higher and non-constant latency. This is due to the interference from other traffic flows in the Class A, (i.e., those originating from the same node, from $node_2$ and $node_4$) and to the additional delay due to CBFQ. The AVB_ST results are comparable with TTE results for both the ST flows from $node_3$ and $node_4$, and in both cases a null jitter is found. The ST flow from $node_4$ shows a lower latency than the $node_3$ ST flow, as the latter has one more bridge to cross (i.e., B_4) to reach the destination. In all these simulations, $node_1$ flow experienced the lowest latency, as this flow is the only one generated from $node_1$ and has to cross only one bridge to reach the listener. In AVB simulations, the latency for this flow is constant as, according to our settings, the frames sent from $node_1$ reach the listener without waiting in the bridge queue (as the Class A queue of B1 is empty when $node_1$ frames go through it). The second lower latency value is found for the AVB stream coming from $node_5$, although such a flow is mapped on the lowest priority in all the simulations, as this flow is the only one generated at $node_5$ and has to cross only two bridges, thus experiencing a limited interference from other flows. The maximum latency for the non-ST flow generated by $node_3$ is lower with standard AVB than with AVB_ST. This is natural, as the latter downgrades the flow, while standard AVB serves it in the highest priority class, i.e. SR Class A. However, the latency increase, for this non-ST flow, in AVB_ST is not dramatic, and as far as the mean latency is concerned, the three protocols offer comparable values. The maximum latency for the $node_3$ non-ST flow is lower with TTE, that serves this flow in the RC class with the second highest priority, and also the jitter obtained by TTE is the lowest one. This is because TTE does not apply CBFQ. We underline that the non-ST flow from $node_3$ is the one that experiences the highest latency with AVB_ST, as this flow not only suffers from the interference due to the ST traffic class and the TABS mechanism, but is also affected by all the other SR-Class A flows, as this flow has to cross five hops (and so, all the bridges) to reach the listener.

6. Conclusion and future work

The AVB_ST approach proposed in this paper to introduce support for scheduled traffic in AVB networks proved to be very beneficial. Thanks to the offset-based scheduling, the temporal isolation provided to the ST traffic class through the TABS mechanism, and strict priority handling, ST traffic obtained low and predictable latency values, without significantly affecting SR traffic. Future work will deal with further refinements of the approach. Moreover, the AVB_ST performance will be investigated through a comprehensive evaluation in different scenarios and under varying workloads.

7. References

[1] 802.1Qbv - Enhancements for Scheduled Traffic. Website: http://www.ieee802.org/1/pages/802.1bv.html.

[2] IEEE P802.1Qbv/D0.2 Draft Standard for Local and Metropolitan Area Networks—Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks—Amendment: Enhancements for Scheduled Traffic, 2013.

[3] D. Pannell, "AVB - Generation 2 Latency Improvement Options", 802.1 AVB group, March, 2011.[Online]:http://www.ieee802.org/1/files/public/docs 2011/new-avb-pannell-latency-options-1111-v2.pdf.

[4] IEEE Standard for Local and metropolitan area networks-Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks, 2011.

[5] L. Lo Bello, "The case for Ethernet in automotive communications," SIGBED Review, vol. 8, n.4, pp. 7–15, Dec. 2011.

[6] H.-T. Lim, D. Herrscher, L. Volker, and M. Waltl, "IEEE 802.1AS Time Synchronization in a Switched Ethernet based In-car Network," in Vehicular Networking Conference (VNC), 2011 IEEE, Nov. 2011, pp. 147 –154.

[7] G. M. Garner, A. Gelter, and M. D. J.Teener, "New Simulation and Test Results for iee 802.1AS Timing Performance, ISPCS '09, Brescia, Italy.

[8] T. Steinbach, H.-T. Lim, F. Korf, T. C. Schmidt, D. Herrscher, and A. Wolisz, "Tomorrow's In-Car Interconnect? A Competitive Evaluation of IEEE 802.1 AVB and Time-Triggered Ethernet (AS6802)," in IEEE Vehicular Technology Conf., pp.1-5, Sept. 2012.

[9] G. Alderisi, G. Iannizzotto, and L. Lo Bello, "Towards 802.1 ethernet avb for advanced driver assistance systems: a preliminary assessment," in 17th IEEE Conference on Emerging Technologies and Factory Automation (ETFA 2012), Kracow, Poland, Sept. 2012.

[10] G. Alderisi, A. Caltabiano, G. Vasta, G. Iannizzotto, T. Steinbach, L. Lo Bello, "Simulative Assessments of IEEE 802.1 Ethernet AVB and Time-Triggered Ethernet for Advanced Driver Assistance Systems and In-Car Infotainment" ," in Proc. of the IEEE Vehicular Networking Conference (VNC), pp. 187-194, Nov. 2012, Seoul, South Korea.

[11] H.-T. Lim, B. Krebs, L. Volker, and P. Zahrer, "Performance evaluation of the inter-domain communication in a switched ethernet based in-car network," in IEEE 36th Conf. on Local Computer Networks (LCN), Oct. 2011.

[12] Nolte, T., Hansson, H., Lo Bello, L.. "Automotive Communications - Past, Current and Future", in Proc. of the 10th IEEE Int. Conference on Emerging Technologies and Factory Automation (ETFA'05), 985-992, Catania, Italy, Sept. 2005.

[13] E. Heidinge, F. Geyer, S. Schneele, M. Paulitsch, "A Performance Study of Audio Video Bridging in Aeronautic Ethernet Networks", 7th IEEE International Symposium on Industrial Embedded Systems (SIES'12), June 2012, pp.67-75.

[14] J. Imtiaz, J. Jasperneite, and S. Schriegel. "A proposal to integrate process data communication to IEEE802.1 Audio Video Bridging (AVB)".In proc. of the 16th IEEE International Conf. on Emerging Techonologies and Factory Automation (ETFA), Toulouse, France, Sep 2011.

[15] J. Imtiaz, J. Jasperneite, and L. Han. "A performance

study of Ethernet Audio Video Bridging (AVB) for Industrial real-time communication", in 14th IEEE Conf. on Emerging Technologies & Factory Automation (ETFA), pp. 1–8, Palma de Mallorca, Spain.

[16] J. Jasperneite, J. Imtiaz, M. Schumacher, K. Weber, "A Proposal for a Generic Real-Time Ethernet System", IEEE Trans. on Industrial Informatics, 5, 2, May 2009.

[17] J. Imtiaz, J. Jasperneite, K. Weber, "Approaches to reduce the Latency for High Priority Traffic in IEEE 802.1 AVB Networks" on 9th IEEE International Workshop on Factory Communication Systems (WFCS), Lemgo, Germany, May 2012.

[18] R. Cummings, K. Richter, R. Ernst, J. Diemer, and A. Ghosal, "Exploring Use of Ethernet for In-Vehicle Control Applications: AFDX,TTEthernet, EtherCAT, and AVB," in SAE World Congress and Exhibition, Apr. 2012, SAE Technical Paper. [Online]: http://papers.sae.org/2012-01-0196.

[19] IEEE Standard for Local and metropolitan area networks– Audio Video Bridging (AVB) Systems," IEEE Std 802.1BA-2011, pp. 1–45.

[20] C. Boiger, "Class A Latency Issues," in IEEE 802.1 Interim Meeting, Kaua'i, HI, USA, Jan. 2011. [Online]: http://www.ieee802.org/1/files/public/docs2011/baboiger-class-a-latency-issues-0111.pdf.

[21] R. Cummings, "802.1Qbv Scheduled Traffic: Window Options", 802.1 Interim Meeting, May 2012, York UK

[22] TTA-Group, "TTEthernet-Spezification,". [Online] http://www.ttagroup.org/ttethernet/specification.html.

[23] IEEE Standard for Local and metropolitan area networks—Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks - IEEE Std 802.1AS-2011.

[24] IEEE Std 802.1Qat, IEEE Standard for Local and metropolitan area networks, Virtual Bridged Local Area Networks, Amendment 14: Stream Reservation Protocol.

[25] IEEE Std. 802.1Qav, IEEE Standard for Local and metropolitan area networks, Virtual Bridged Local AreaNetworks, Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams, 2009.

[26] IEEE Std 1588-2008, Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems.

[27] R. L. Scheiterer, D. Obradovic "Synchronization Performance of the Precision Time Protocol in Industrial Automation Networks", IEEE Trans. on Instrumentation and Measurement,. 58, 2, Jun 2009.

[28] H. Zinner, J. Noebauer, J. Seitz, T. Waas, "A Comparison of Time Synchronization in AVB and FlexRay in-vehicle networks", in proc of the 9th Workshop on Intelligent Solutions in Embedded Systems (WISES), 2011

[29] K. Tindell, "Adding Time-Offsets to Schedulability Analysis", Rep. YCS221, Dept. of Computer Science,York University 1994.

[30] OMNeT++ Community, "OMNeT++ 4.0." [Online]: http://www.omnetpp.org.

[31] L. Lo Bello, A. Raucea, G. Patti, O. Mirabella, "L-PTP: a Novel Clock Synchronization Protocol for Powerline Networks" in 17th IEEE Conference on Emerging Technologies and Factory Automation (ETFA 2012), Kracow, Poland, Sept. 2012