Journal of Systems Architecture 117 (2021) 102137

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Time-Sensitive Networking in automotive embedded systems: State of the art and research opportunities

Mohammad Ashjaei [a], Lucia Lo Bello [b], Masoud Daneshtalab [a], Gaetano Patti [b], Sergio Saponara [c], Saad Mubeen [a,*]

[a] Mälardalen University, Sweden
[b] University of Catania, Italy
[c] University of Pisa, Italy

ARTICLE INFO

Keywords:
Time-Sensitive Networking
TSN
Automotive embedded systems

ABSTRACT

The functionality advancements and novel customer features that are currently found in modern automotive systems require high-bandwidth and low-latency in-vehicle communications, which become even more compelling for autonomous vehicles. In a recent effort to meet these requirements, the IEEE Time-Sensitive Networking (TSN) task group has developed a set of standards that introduce novel features in Switched Ethernet. TSN standards offer, for example, a common notion of time through accurate and reliable clock synchronization, delay bounds for real-time traffic, time-driven transmissions, improved reliability, and much more. In order to fully utilize the potential of these novel protocols in the automotive domain, TSN should be seamlessly integrated into the state-of-the-art and state-of-practice model-based development processes for automotive embedded systems. Some of the core phases in these processes include software architecture modeling, timing predictability verification, simulation, and hardware realization and deployment. Moreover, throughout the development of automotive embedded systems, the safety and security requirements specified on these systems need to be duly taken into account. In this context, this work provides an overview of TSN in automotive applications and discusses the recent technological developments relevant to the adoption of TSN in automotive embedded systems. The work also points at the open challenges and future research directions.

1. Introduction

Embedded software has recently been the key enabler for advanced functionality and features in automotive systems, including on-road vehicles such as modern cars and off-road vehicles such as construction vehicles, mining vehicles, forest machines, recycling cranes, to mention a few [1–3]. One major consequence of the increasing demand for new software-based features in these systems is the drastic increase in size and complexity of the automotive software, which makes its development a challenging task. For example, the size of the software in a modern premium car has already reached the order of 100 million source lines of code (SLOC) that can be translated to over 1 GB of software code [1,4]. Already today, Original Equipment Manufacturers (OEMs) are estimating 1 billion SLOC in the future driver-less cars, according to Jaguar Land Rover [5].

Many automotive embedded systems are constrained by stringent timing requirements. Hence, developers of these systems have to not only manage the software complexity, but also verify *timing predictability* during their development. A system is considered timing predictable if it is possible to prove or demonstrate, at the design time, that it will meet all the specified timing requirements when executed [8–10]. Another dimension of complexity in these systems is the distribution of the software over several tens of Electronic Control Units (ECUs) that can be connected to five or more different types of in-vehicle networks. The traditional networks include Controller Area Network (CAN), CAN-FD, CAN XL, Local Interconnect Network (LIN), Media Oriented System Transport (MOST), FlexRay, Ethernet, among others [11]. Furthermore, there are several higher-level protocols in the automotive domain that are used on top of some of these protocols like CAN, including CANopen, MilCAN, HCAN, J1939 and AUTOSAR Com [12]. Modern vehicles use sophisticated sensors, cameras and Lidars to be able to support extensive and advanced functionalities. These sensors generate hundreds of megabytes of data per second that needs to be communicated among the onboard computing units

* Corresponding author.
E-mail address: saad.mubeen@mdh.se (S. Mubeen).
[1] https://1.ieee802.org/tsn

https://doi.org/10.1016/j.sysarc.2021.102137
Received 31 October 2020; Received in revised form 25 February 2021; Accepted 6 April 2021
Available online 10 April 2021

Table 1
Data-rates supported by the low-latency in-vehicle communication protocols.

In-vehicle Communication Protocol	Maximum Data-rate
Local Interconnect Network (LIN)	20 kbit/s
Controller Area Network (CAN)	1 Mbit/s
CAN-FD (Flexible Data)	5 Mbit/s (data), 1 Mbit/s (arbitration, ack)
CAN XL	10 Mbit/s (data)[a], 1 Mbit/s (arbitration, ack)
FlexRAY	10 Mbit/s
Ethernet with Time-Sensitive Networking	100 Mbit/s to 10 Gbit/s

[a]The recommended maximum data-rate for CAN XL in the data phase is 10 Mbit/s. However, the data-rate can be higher depending upon the CAN transceiver capabilities and characteristics of physical layer components [6,7].

with predictable low latencies (microseconds to milliseconds range). Although the traditional in-vehicle networks provide low-latency communications, they do not meet the high-bandwidth requirements. The maximum data-rates supported by these low-latency networks range from 10 kbit/s to 10 Mbit/s as depicted in Table 1. Due to this limitation, a high-bandwidth and low-latency network protocol is needed to support the backbone in-vehicle communications in modern vehicles. The IEEE Time Sensitive Networking (TSN) task group[1] provided a set of standards to meet the high-bandwidth and low-latency in-vehicle communications requirements. Among those advancements in the TSN standards we can mention support for clock synchronization, resource reservation for different types of traffic, various traffic shapers, scheduled traffic support, frame preemption, and network management mechanisms. This paper deals with in-vehicle communications based on the TSN standards. Consequently, here the focus is on the TSN standards only, while the integration of TSN with other technologies is not in the scope of this work. Interested readers can refer to [13] to learn about the integration of TSN with other communication technologies, e.g., 5G. Moreover, among the TSN standards, only the ones relevant to automotive communications will be considered.

Model-based Engineering (MBE) [14,15] and Component-based Software Engineering (CBSE) [16], complemented by real-time scheduling and schedulability analysis [17–19], are proving effective in dealing with the challenges of software complexity and ensuring timing predictability during the development of automotive embedded systems. MBE uses models to describe functions, structures and design artifacts throughout the software development, whereas CBSE allows to build large software systems by reusing pre-existing software components (SWCs)[2] and their architectures. Benefits of model- and component-based software development in the automotive domain include *software reuse*, early model-based timing predictability verification [20–22], refinement of software architectures based on timing verification [23–25], to mention a few. According to an estimate in the segment of heavy trucks, up to 90% of automotive software can be reused from previous releases or other projects if MBE and CBSE are used [3,26].

In order to fully utilize the potential and benefits of MBE and CBSE for the development of automotive embedded systems, the modeling of in-vehicle networks such as TSN should be supported at the abstraction level where software architectures of the systems are modeled. For example, consider the software architecture of the two-node automotive embedded system that is shown in Fig. 1. Development techniques based on MBE and CBSE facilitate the modeling of network communications in the software architecture of the system, modeling of timing properties in the network, and specification of end-to-end timing requirements and constraints (e.g., age and reaction constraints according to the AUTOSAR standard [27]). Moreover, the *end-to-end timing models* can be automatically extracted from the software architectures and fed to the timing analysis frameworks and tools. An end-to-end timing model consists of timing models of nodes, network timing model, linking model that models distributed chains consisting of SWCs and network messages (e.g., shown with red-dashed arrows in Fig. 1), and timing requirements model. We refer the reader to [20,28–30] for further details about the timing models. The analysis frameworks and tools verify the timing predictability of the automotive software architectures. The analysis results are then back-propagated to adjust or refine the software architectures to meet the specified requirements or to optimize the architectures based on other resource constraints specified on the automotive systems. The timing verified and refined software architectures can be deployed on nodes or ECUs that are connected to a TSN network. The TSN network may contain commercial-off-the-shelf (COTS) switches or fully parameterized and modular Intellectual Property (IP) TSN switches realized on Field Programmable Gate Arrays (FPGAs). Other important aspects that govern the development of automotive embedded systems are the safety and security requirements in these systems, which need to be considered throughout the development process.

1.1. Paper contribution

This paper addresses the core aspects in the model-based development of automotive embedded systems, as depicted in Fig. 1, while focusing the discussion on utilizing TSN in these systems. In this context, the paper reviews the state of the art, provides an overview of the recent technological developments relevant to the adoption of TSN in automotive embedded systems, and discusses open challenges and future research directions. The following core topics are discussed:

1. Evolution of Ethernet and TSN in automotive embedded system applications;
2. Model-based software development of automotive distributed embedded systems that utilize TSN;
3. Scheduling and schedulability analysis of TSN;
4. Simulation platforms for TSN, including OMNeT-based TSN simulation models and their evolution;
5. Hardware evolution in TSN, addressing implementations in FPGA technology and in heterogeneous platforms like Zynq;
6. Safety and security in TSN.

1.2. Paper layout

The rest of the paper is organized as follows. Section 2 addresses the role of Ethernet and TSN in the automotive domain. Section 3 discusses the support for modeling TSN communications within the model-based software development of automotive embedded systems. Section 4 presents the recent advancement in scheduling and schedulability of TSN. Section 5 provides a discussion on simulation frameworks for TSN. Section 6 discusses hardware realization for TSN. Section 7 presents the safety and security aspects of TSN. Finally, Section 8 concludes the paper.

[2] SWC is the lowest-level hierarchical entity in a software architecture. It corresponds to a schedulable entity at runtime, e.g., an operating system task.

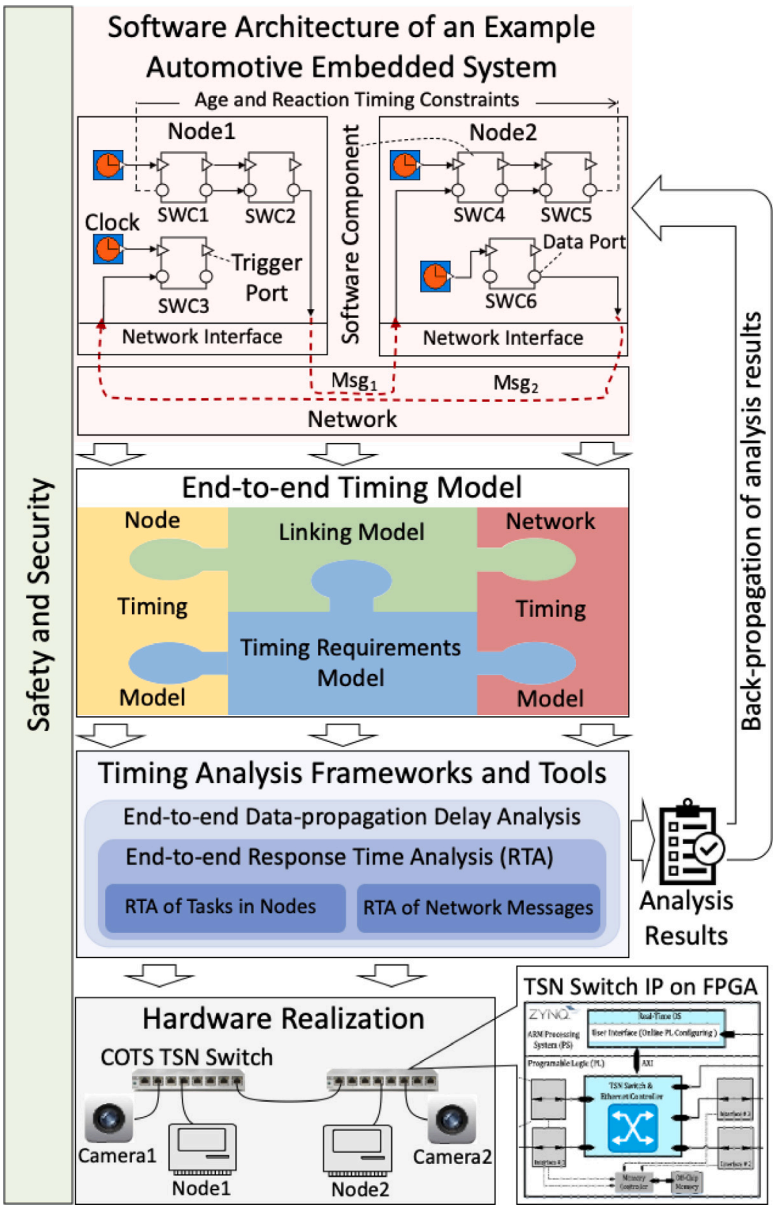

Fig. 1. Key aspects and flow in model-based development of automotive embedded systems.

2. Ethernet and time-sensitive networking in automotive applications

The TSN family of standards is a tool set that offers reliability, determinism and time synchronization to safety-critical automotive communications over Ethernet links. The TSN standards leverage on the previous work done within the IEEE 802.1 Working Group on the IEEE Audio Video Bridging (AVB).

The AVB standards offer several features. The first one is a common notion of time. This is achieved through a suite of clock synchronization protocols that allow end stations and switches (called bridges in the IEEE terminology) to synchronize their local clocks to each other (IEEE 802.1AS-2011) provided that they are time-aware nodes, i.e., systems that make explicit reference to time. The standard defines a Best Master Clock Algorithm (BMCA) to select the time reference node, and a Generalized Precision Time Protocol (gPTP), to synchronize the clock of nodes providing them the clock value of the reference node, called Grand Master (GM).

The second feature offered by AVB is bandwidth reservation for real-time classes. This is obtained through the Stream Reservation Protocol (SRP), which allows for the reservation of resources within the bridges on the path between the source and the destination (IEEE 802.1Qat).

Finally, the third feature provided by AVB standards is prioritization and traffic shaping for real-time flows to prevent traffic bursts. This is achieved by means of a Credit-Based Shaper (CBS) at the output ports of bridges and end nodes, which guarantees bounded latency to real-time classes (called Stream Reservation Classes) (IEEE 802.1Qav).

On top of the AVB protocol suite, TSN provides a flexible toolbox, consisting of protocols adding several additional features that networks designers can combine to obtain the properties required to meet the application needs. Fig. 2 shows the main TSN standards that are relevant to the context here addressed, i.e., in-vehicle communications.

Roughly speaking, TSN standards target one or more of the following four design aims, i.e., bounded low latency, timing and synchronization, high reliability, and resource management.

Bounded low latency is achieved by many standards. Among them, we find the already mentioned IEEE 802.1Qav and Qat from the AVB protocol suite, that are now enrolled in the 802.1Q-2018 standard [31], but there are many others. For example, the IEEE 802.1Qbv, that introduces the so-called enhancements for scheduled traffic, which

Name	Status	Rolled into	Relevant applications
P802.1AS-2020: Timing and synchronization for Time-Sensitive Applications.	Published	Standalone project	Audio/Video, Automotive, Industrial
802.1Qbu-2016: Frame Preemption	Published (enrolled in 802.1Q-2018)	802.1Q-2018	Audio/Video, Mobile, Automotive, Industrial
802.1Qbv-2015: Enhancements for Scheduled Traffic	Published (enrolled in 802.1Q-2018)	802.1Q-2018	Automotive, Industrial
802.1Qch: Cyclic Queuing and Forwarding	Published	802.1Q-2018	Audio/Video, Automotive, Industrial
802.1Qci: Per-Stream Filtering	Published	802.1Q-2018	Audio/Video, Automotive, Industrial
802.1CB: Frame Replication and Elimination for Reliability	Published	Standalone standard	Audio/Video, Mobile, Automotive, Industrial
802.1Qcc-2018: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements.	Published	Amends 802.1Q-2018	Audio/Video, Automotive, Industrial
P802.1Qcr-2020: Asynchronous Traffic Shaping	Published	Amends 802.1Q-2018	Automotive, Industrial
P802.1DG – TSN Profile for Automotive In-Vehicle Ethernet Communications	In progress	Ongoing project	Automotive

Fig. 2. Main TSN standards relevant to in-vehicle communications.

leverage on a transmission gate mechanism that is applied to the egress queues of a switch port to allow for transmitting traffic according to a predefined time schedule, implemented as a list of timed gate operations that cyclically repeats, called a gate control list. Another standard that is relevant to bounded low latency is the IEEE 802.1Qbu that, in combination with the IEEE 802.3br, allows for frame preemption to intersperse express (i.e. real-time) frames among best-effort ones. Moreover, the IEEE 802.1Qch, called Cyclic Queuing and Forwarding, introduces a mechanism according to which frames are received and transmitted alternately for a fixed interval of time, called a cycle time. This way, real-time frames are accumulated during a cycle and transmitted in the following one, thus minimizing the transmission jitter and guaranteeing bounded end-to-end delays. Finally, the IEEE 802.1Qcr-2020 standard defines the Asynchronous Traffic Shaping (ATS) stream selection. The ATS enables the transmission of mixed real-time traffic types, such as periodic, rate-constrained, and event-driven (sporadic). The protocol works over the port transmission queues at a stream level. The main concept of the ATS is to realize a traffic shaping through a token bucket that limits the burst size of a given traffic class. To accomplish this, an eligibility time is assigned to each received frame. When the current time is higher than or equal to the eligibility time for a given frame, the latter can be queued in the transmission queue and then transmitted.

The standards relevant to timing and synchronization are the already mentioned IEEE 802.1AS-2011 and its 2020 revision, the IEEE 802.1AS-2020, that improves the clock synchronization reliability and solves some issues of the former standard. For example, the IEEE 802.1AS-2020 standard provides replication mechanisms to limit the probability of decreasing the synchronization accuracy in case of bridge fault or frame loss.

Reliability is the focus of the IEEE 802.1CB, that deals with frame replication and elimination for reliability. By leveraging on frame identification capability, the IEEE 802.1CB standard allows for duplicating frames and sending them over multiple disjoint routes in order to increase the probability that at least one of the replicas will eventually reach the final destination. The first replica that is delivered to the destination is considered correct, so the remaining ones will be discarded. In the case the end stations are not able to implement this capability, frame duplication and elimination will be up to the closest switches, that will transparently act as proxies.

Another standard that goes in the direction of improving reliability is the IEEE 802.1Qci, which introduces support for per-stream metering and monitoring, error detection and error mitigation, by blocking a stream or a port to enforce the error containment so that it will not propagate on the network. The IEEE 802.1Qci can also apply ingress policing and filtering to improve security, blocking a traffic source when an unforeseen or not compliant traffic is detected.

Resource management and network configuration are accomplished thanks to the already mentioned IEEE 802.1Qat standard, which defines the SRP, and the IEEE 802.1Qcc, that extends the capabilities of SRP. In particular, the IEEE 802.1Qcc describes protocols to provide support for Configurable Stream Reservation classes and streams.

The main difference between the protocols in the IEEE 802.1Qcc compared to the SRP is that the SRP is a simple admission protocol in which a talker announces the traffic for transmission and depending on the available resources it can get granted. Conversely, the new protocols in the IEEE 802.1Qcc deal with more complex configurations with TAS and preemption mechanisms. The main idea is that a User Network Interface (UNI) specifies the requirements for sending a stream, while the network analyzes this information to check whether the switches can meet the requirements. To realize this, three different network configuration methods are presented, i.e., a fully distributed, a centralized network/distributed user model, and a fully centralized model. Another interesting effort within the standardization is to support a data modeling language that is defined by the Internet Engineering Task Force (IETF) [32], known as YANG model, to be used for the network configuration (NETCONF). Similar to the YANG model, NETCONF is also developed within the IETF [32] to create a configuration management that can easily push new configurations to the network devices using a global knowledge of the network [33]. NETCONF defines a client/server model in which the servers are the end stations, while the client is a policy manager located in the Centralized Network Configuration (CNC) unit.

Finally, it is worth mentioning an ongoing project within the Time-Sensitive Networking Task Group of IEEE 802.1, i.e., the P802.1DG Draft,[3] titled Time-Sensitive Networking Profile for Automotive In-Vehicle Ethernet Communications. This document aims at specifying

[3] https://1.ieee802.org/tsn/802-1dg

profiles for automotive in-vehicle switched Ethernet networks based on TSN and the IEEE 802.1 Security standards. The motivation for the project is to provide “profiles for designers and implementers of deterministic IEEE 802.3 Ethernet networks that support the entire range of in-vehicle applications, including those requiring security, high availability and reliability, maintainability, and bounded latency".

2.1. Novel trends for in-vehicle networks

Thanks to the novel features introduced by the TSN standards, nowadays Automotive Ethernet is unanimously considered the key solution towards a homogeneous in-vehicle network that will replace the multiple small networks connected via gateways used in today’s cars.

The Ethernet success in automotive applications stems from several factors, such as, the higher bandwidth comparing with current in-car networks, the support offered to the Internet Protocol (IP) stack, the IEEE standardization, and the availability of a broad range of datarates for automotive usage [34].

Fig. 3 shows, in chronological order, the published IEEE standards that refer to Ethernet Physical Layers (PHYs) for automotive applications. The supported datarates are:

- 100 Mbps, standardized as IEEE 802.3bw-2015, which defines the 100BASE-T1 PHY specifications.
- 1 Gbps, standardized as IEEE 802.3bp-2016, which adds point-to-point 1 Gbps PHY specifications.
- 10 Mbps, standardized as IEEE 802.3cg-2019, also known as 10Base-T1S, which specifies additions to add 10 Mbps PHY specifications. This standard is commonly referred as 10Base-T1S.
- 2.5, 5 and 10 Gbps, Multi-Gig Automotive Ethernet PHY, approved in June 2020 as IEEE 802.3ch-2020, which adds PHY specifications for 2.5 Gbps, 5 Gbps, and 10 Gbps operation for automotive applications.

Nowadays, in-vehicle networks have begun the transition from legacy domain-based electrical and electronic architectures to zonal architectures [35,36]. Domain-based architectures have many separate ECUs and different networks for each automotive subsystem (e.g., powertrain, body electronics, multimedia/infotainment, to mention a few) that require complex gateways. Conversely, zonal architectures consolidate many of these cross-domain ECU functions into a small number of supercomputer-level ECUs (called zonal controllers) that are networked through a backbone consisting of Ethernet switches working at ultra-high speeds (e.g., greater than 10 Gbps). Zonal architectures are a concept for the future in-vehicle architectures, where less, but more powerful ECUs, with High Performance Computing abilities and high data storage available, will carry on computationally-intensive control tasks. Zonal architectures are expected to facilitate service-oriented operation, in which services will be provided by application components through a communication protocol over the network. Their implementation in vehicles can be realized using local Ethernet gateways (or switches) per zone (working at speeds of 10 Mbps, 100 Mbps, 1 Gbps, 5 Gbps, up to 10 Gpbs), which connect to legacy automotive buses (e.g., LIN, CAN, CAN FD, FlexRAY), running some fast control function in the periphery of the vehicle and basic functions to convert legacy automotive networks, which are signal-based to service-based on Ethernet, and vice versa. This architectural concept provides high flexibility and scalability, as a large number of sensors and actuators can be added to the vehicle, but they do not need to be directly connected via the zonal controllers, as each zonal controller can be connected to traditional in-vehicle networks, like CAN and LIN. Communication between zones takes place via a secured ultra-high speed Ethernet backbone (from 10 to 25 Gbps). This way, Ethernet will become the dominating network technology in future vehicles.

This design change has generated the need for data rates greater than 10 Gbps in the automotive environment. However, the IEEE Standard 802.3 does not currently support rates greater than 10 Gbps for automotive usage. In this direction, work is currently in progress within the IEEE (802.3ch) on 25, 50 and 100 Gbps – with the P802.3cy project,[4] i.e., “Greater than 10 Gbps Electrical Automotive Ethernet”.

In zonal designs, legacy network technologies, e.g., CAN and LIN, will still be used to connect many electronic components. However, it is well understood that the Ethernet 10Base-T1S technology will be a competitor of CAN, CAN-FD, and CAN XL. The latter is an upcoming CAN data link layer backward compatible with CAN-FD that will provide “extra large” payloads of up 2048 bytes and a data rate of 10+ Mbps.

3. Model-based software development of automotive systems utilizing TSN

The research community has developed several domain-specific modeling languages and methodologies that are used for model- and component-based software development of automotive embedded systems. Examples of these languages and methodologies include AUTOSAR [27], EAST-ADL [37], AMALTHEA,[5] AMALTHEA4public,[6] Rubus Component Model [38,39], ProCom Model [40], CORBA [41], COMDES [42], to mention a few. However, a large majority of these languages and methodologies support modeling of traditional in-vehicle networks, such as CAN, LIN and FlexRay [28,43,44]. There are very few works, such as [45], that model high-bandwidth in-vehicle networks that are based on switched Ethernet (e.g., AVB) in the software architectures of these systems. AUTOSAR, supported by the SymTA/S approach [43] and corresponding tool,[7] also models automotive Ethernet at the software architecture abstraction, but the support for modeling TSN is still missing. Furthermore, SymTA/S tool does not expose any details about the underlying techniques, as it is a commercial tool. There are several works that model configuration of TSN, such as [46–48], but they do not support TSN modeling in the software architectures of these systems. A work in progress on developing an approach for the TSN configuration and formal verification of the modeled configuration is discussed in [49].

The first comprehensive modeling approach for TSN has been recently developed in [50], which allows integration of TSN models into a model- and component-based software development environment for automotive embedded systems. This modeling approach is expressive enough to allow extraction of end-to-end timing models from the software architectures of automotive embedded systems that use TSN for in-vehicle communications. The proof-of-concept for the approach is provided in an existing industrial component model and corresponding tool chain, namely the Rubus Component Model (RCM) [38] and Rubus-ICE,[8] respectively. The usability of the modeling approach is demonstrated by modeling and analyzing an automotive application use case from the industry. Although RCM exposes the underlying concepts and techniques to the scientific community [39], it is a commercial model and tool chain, and therefore it is not freely available.

This calls for the research community to develop open source or freely available modeling solutions for TSN that can be integrated into the existing automotive software development frameworks. In this regard, promising prospective solutions include developing a domain-specific profile of the Unified Modeling Language (UML) and extending the UML MARTE [51] profile to support TSN. Another research direction is the integration of the evolving timing analysis techniques for TSN (discussed in Section 4) in the above mentioned software development frameworks and tools for automotive embedded systems.

[4] https://standards.ieee.org/project/802_3cy.html
[5] http://www.amalthea-project.org
[6] https://itea3.org/project/amalthea4public.html
[7] SymTA/S is now acquired by Luxoft (https://www.luxoft.com).
[8] https://www.arcticus-systems.com/products/

Name	Amendment to	Scope
IEEE 802.3bw-2015	IEEE 802.3-2012	Defines the 100BASE-T1 Physical Layer (PHY) specifications and management parameters for point-to-point full duplex 100 Mb/s operation over single twisted pair balanced cabling.
IEEE 802.3bp-2016	IEEE Std 802.3-2015	Adds point-to-point 1 Gb/s Physical Layer (PHY) specifications and management parameters for operation on a single twisted-pair copper cable in an automotive application.
IEEE 802.3cg-2019	IEEE Std 802.3-2018	Specifies additions and appropriate modifications to add 10 Mb/s Physical Layer (PHY) specifications and management parameters for operation, and associated optional provision of power, over a single balanced pair of conductors. 10Base-T1S.
IEEE 802.3ch-2020	IEEE Std 802.3-2018	Adds physical layer specifications and management parameters for 2.5 Gb/s, 5 Gb/s, and 10 Gb/s operation on a single balanced pair of conductors suitable for automotive applications.

Fig. 3. The IEEE published standards relevant to Ethernet PHY for automotive applications.

4. Scheduling and schedulability analysis of TSN

This section provides a review on the works that have been done so far on two main aspects of designing distributed embedded systems using TSN networks. The first aspect has to do with creating an offline schedule for TSN scheduled traffic, which is known to be an NP-complete problem. The second aspect has to do with verifying and guaranteeing the traffic timeliness in TSN networks. The majority of the considered works proposed solutions and methods for generic applications and only few of them motivated and evaluated their solutions for automotive use cases.

4.1. Scheduling algorithms

Time-triggered communication [52] is an established paradigm in industrial communications that is very suitable for systems that have strict timing requirements on delays and jitter. In time-triggered communications, the time is divided into time-slots and each frame is transmitted within a dedicated time-slot, thus leading to a contention-free communication. The transmission plan of frames within the predefined time-slots, that is called a time-triggered schedule, has to be computed offline for a system. The time-triggered schedule is a bin-packing problem that is known as NP-complete, in particular for multi-hop networks that require path-independent schedule for frames. One of the earliest solutions to extract a time-triggered schedule for a multi-hop Ethernet network was presented in [53]. The work uses the Satisfiability Modulo Theories (SMT) solver to find a solution based on a set of scheduling and timing constraints. The extension of the work to consider co-scheduling of tasks and frames in a network was presented in [54]. However, the above-mentioned works are in the context of the Time-Triggered Ethernet (TTE) protocol [55]. It is worth mentioning that TTE was developed for time-critical and mission-critical applications, including the automotive domain. As there are similarities between TTE and the enhancements for scheduled traffic in TSN networks defined in the IEEE 802.1Qbv amendment, several recent works provided schedule for TSN traffic using the same model used for TTE, i.e., the same timing and scheduling constraints. In order to provide scheduling for TSN networks, several algorithms and approaches have been taken by different research groups so far. A taxonomy of the solutions is presented in Fig. 4.

In the area of the TSN enhancements for scheduled traffic, the optimization technique based on an SMT solver and the concept of OMT (Optimization Modulo Theories) is developed in [56]. The SMT solver can find multiple solutions to fulfill the scheduling and timing constraints. The OMT module, however, can define an optimization problem, e.g., minimizing the end-to-end delays of frames. The proposal in [56] uses the OMT module to minimize the number of queues per port for the scheduled traffic. In the presented work the main requirements are on frame latency and jitter, whereas tailoring of the schedules for device-specific properties is missing. The main step that would naturally come after finding a schedule is to map it to the gate control list per network port. The work in [57] uses the first-order theory of arrays to formalize the scheduling and timing constraints to optimize the solution. The solution can be applied directly to the gate control list. The overview of the challenges and scheduling mechanisms is presented in [58]. The work in [59] proposed an optimization method to schedule the gate control list as windows for transmission leading to a better schedule for networks. There are also few works with different optimization goals than minimizing the end-to-end delays, e.g., making larger porosity on the network links [60], [61]. Moreover, a Tabu search algorithm is proposed in [62] to provide a no-wait packet schedule for TSN where the number of guard bands are minimized in the given schedule.

We can also find few solutions that do not follow the common optimization tools, such as the OMT and Integer Linear Programming (ILP). For instance, the solution in [63] is based on genetic algorithms to find a schedule for the scheduled traffic with lower time-complexity compared to the SMT-based solutions. In addition, the work in [64] proposed a heuristic, called a heuristic list scheduler (HLS), to generate valid schedules considering both routing and scheduling constraints. The work in [65] addressed the problem of time-triggered scheduling considering the minimum queue utilization where the solution is based on a multi-objective combinatorial optimization problem with several strategies, such as ILP, heuristic, and metaheuristic strategies. In addition, the work in [66] proposed a method to co-schedule the traffic and routing in a TSN network.

The above-mentioned solutions mostly focus on the schedule synthesis of the scheduled traffic in the TSN networks. There are a few works that consider other constraints besides the timing requirements. For instance, the work in [67] not only considers the timing constraints, but also provides redundant paths for traffic to obtain a fault-tolerant network. Moreover, the work in [68] proposes a solution to schedule the scheduled traffic in a TSN network considering the schedulability of the lower priority classes, i.e., classes A, B and best effort. The solution is extended in [69] for routing and scheduling optimization considering the AVB traffic. The scheduler that is proposed in [70] considers a quality of service level for best-effort traffic, while scheduling the ST frames. The work in [46] studied the co-routing and co-scheduling of

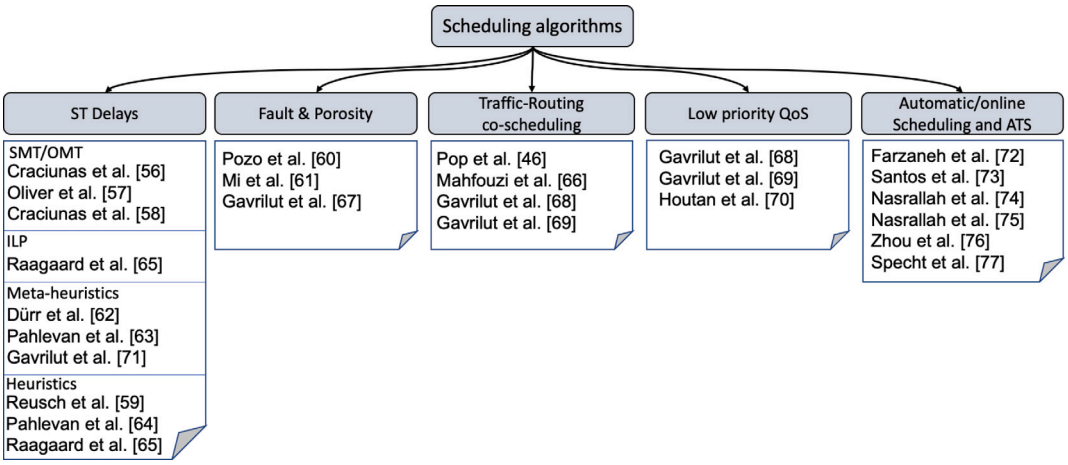

Fig. 4. A taxonomy of TSN scheduling algorithms.

hard real-time flows and best-effort audio video-bridging flows. In the proposed method, the co-scheduling problem is solved by using ILP and the co-routing problem is solved by proposing an adaptive heuristic search. Finally, the work in [71] proposed a solution based on a tabu-search metaheuristic to find a feasible schedule for a mixed-critical system in which both hard and soft real-time traffic co-exist.

In addition to the proposed solutions for extracting the best (and possibly the optimum) schedule for the scheduled traffic in a TSN network, there are several works in the literature to automate parts of the TSN configuration. For instance, the work in [72] uses a graph-based solution to generate the constraints for the schedule. Moreover, a tool named TSNSCHED is developed in [73] to automatically generate a schedule based on the SMT solver to fulfill the timing constraints. An interesting work in [74] provided a method to dynamically reconfigure the time-aware shaper configuration depending on the requirements. The method is supported by the IEEE 802.1Qcc standard based on centralized and distributed network control within TSN networks.

Previously in this work we reviewed the literature relevant to the time-aware shaper in TSN standards. In addition, there are few works aimed at evaluating the performance of the asynchronous traffic shaper (ATS). For instance, the work in [75] compares the TAS and ATS shapers to evaluate whether the ATS can obtain the same level of performance as the TAS in industrial networks. Although the scope of the work is not in the automotive industry, the evaluation results can give insights about the TAS usage in automotive use cases. In addition, the work in [76] presents a performance evaluation of the ATS shaper. The work shows that there is a trade-off in using the TAS shaper, while it can achieve an effective traffic shaping and switching. The main advantage of the ATS shaper is that it does not need clock synchronization in the network, thus reducing the network complexity. The paper [77] proposes the urgency-based scheduler (UBS) and an optimum scheduling algorithm (based on SMT) to solve the synthesis problem for it.

The review above presented shows that so far the challenge of finding a feasible schedule in a TSN network has been deeply addressed in the literature mostly using SMT and OMT solvers. Although such solvers proved to work efficiently even for industrial-size TSN networks, one of their main drawbacks is their time-complexity, in particular, when they are applied to larger networks with routing planning together with schedule synthesis. In addition, the recent advancements in the TSN configuration management allow dynamic configuration or even self-configuration of a TSN network. In this type of networks, a schedule or route can be dynamically updated according to any change in the system or environment during the system run-time. In this context, solutions to find an optimum schedule with lower time-complexity might be required. Consequently, deep investigation and comparative study of search-based, meta-heuristics and heuristic solutions are essential to achieve a solution with low time-complexity, such as a sub-optimal schedule suitable for online use. Another line of work that has received little attention so far is considering the quality of service provided to the AVB traffic classes when scheduling and routing the TSN traffic. One interesting observation is that the majority of the above mentioned solutions is evaluated on example or synthetic network topology, e.g., in [63] with grid and mesh topology or in [64] with ring and mesh topology. The work in [67] evaluated the proposed solution using five case studies provided by General Motors. Nevertheless, the mentioned works lack of real use cases, especially within the automotive domain, to address the performance of the solutions.

4.2. Schedulability analysis of traffic in TSN networks

The TSN standards are becoming accepted solutions in complex time-critical systems, such as in the automation and automotive industries. In a time-critical system, it is essential to verify the timing predictability of the system during the design phase, i.e., verify the timing requirements that are specified on the system. Schedulability analysis techniques are used to verify whether the system is schedulable under given scheduling algorithms and that the specified timing requirements are satisfied [78,79].

Various schedulability analysis techniques have been already proposed in the TSN research community to calculate the worst-case delay of traffic crossing through a TSN network. Many of these techniques focus on the analysis of worst-case delays for classes A and B solely under the CBS. We can mention the analysis presented in [80,81] and [82], where the first work adopts delay computation and the latter two works exploit Network Calculus. The above cited works compute the worst-case delays per traffic class without providing upper bounds on the delays of individual frames. However, in a time-critical system it is essential to verify the timing requirement on each individual frame. Consequently, several works started to compute the worst-case delays of the classes A and B frames, e.g., the technique in [83] and the improved technique given in [84]. Interestingly, different approaches have been taken by different research groups to verify the timing requirements in TSN networks. Among these approaches, the work in [85] proposes a technique based on the trajectory approach to compute the delays of classes A and B. The technique obtains tighter bounds on the frame delays compared to the previous techniques [83]. Later, the work in [86] proposed the notion of eligible interval, which could provide a bound for per-frame delays, thus leading to tighter analysis. Note that providing a tight analysis is important for such systems that are limited in resources. As an intuitive fact we know that considering more resources (e.g. network bandwidth) allows to allocate

more schedulable frames in the network. However, as an optimum design on one hand we would like to minimize the resources being used in the system, and on the other hand to use more schedulable frames. Thus, a tight analysis for the schedulability test allows us to prevent pessimistic design and achieve better utilization of the available resources. Two works by the same authors in [87] and [88] showed that it is possible to use machine learning techniques to partially evaluate the feasibility of TSN networks. The techniques can be used when a large state space of TSN configuration should be checked, thus conventional feasibility checks are slower, while machine learning techniques may result in less accurate predication.

The above-mentioned works solely consider the CBS. There are a very few works that provide schedulability analysis for TSN considering other traffic shapers. For instance, the proposal in [89] presents an analysis technique based on the network calculus considering a TSN shaper, called the Burst Limiting Shaper (BLS). The work is extended in [90] to evaluate the proposal for realistic avionics case studies.

As it was mentioned before, the TSN standards define the gate mechanism to provide support for scheduled traffic, and therefore several works modified previous schedulability analysis techniques so as to consider the gate mechanism. For instance, the technique that is proposed in [91] computes the worst-case delay of classes A and B in a TSN network considering both the CBS and the gate mechanism. However, the analysis technique is proposed for a single-switch network, which could be a limitation as many industrial networks consist of multiple switches. Consequently, later works attempt to provide schedulability analysis techniques for larger networks. The work in [92] provides an analysis technique based on calculating accumulating delays, whereas the works in [93] and [94] use network calculus to check the schedulability of the system. In addition, the technique in [95] presents a response time analysis for classes A and B traffic considering the CBS and the support for scheduled traffic that is proposed in [96]. The traffic forwarding and shaping model in the latter work is different from the TSN standard models, as it was proposed before the finalization of the first TSN draft. A comparative evaluation of the two models is performed in [95]. In addition to the evaluation, the work in [95] presents an efficient solution for network bandwidth allocation to frame sets. The proposal is further modified in [97], where an efficient per-link per-class bandwidth allocation is extracted using an algorithm to optimize the network resource utilization.

Another research direction in the context schedulability analysis for TSN is to consider the preemption support in TSN networks along with the CBS and the gate mechanism. The progress in this direction is limited to few recent works, which require improvement to reduce pessimism in the analysis. It should be noted that few works have addressed the preemption model and its improvements, e.g., [98] and [99]. However, there is currently no analysis technique considering the proposed novel preemption models. The work in [100] proposes a technique to perform an analysis considering the preemption support under the IEEE802.3br standard. Further, the work in [101] presents an analysis considering multiple classes of traffic instead of only classes A and B. In addition, the latter work considers the CBS, gate mechanism, and preemption. A recent work in [102] proposes a response time analysis for classes A and B, while taking the CBS, gate mechanism and preemption in the TSN networks into account. The work considers various preemption modes, including the hold and release mechanism and its effects on the jitter of the scheduled traffic in the network.

In light of the above discussion, it can be concluded that the schedulability analysis techniques for TSN, that solely consider the CBS, have gained relatively high level of maturity. Furthermore, some of these techniques can compute tight upper bounds on the frame delays, e.g., the technique that uses the eligible interval approach [86]. However, these technique are not extended to other TSN mechanisms, e.g., the gate mechanism and preemption. When it comes to the analysis techniques that consider the CBS along with the gate mechanism and preemption, we observe that there are very few works and they are pessimistic. An interesting research direction is to perform a comparative evaluation of the existing works and reduce the pessimism in their calculations. Another point is that the existing works consider several restrictions in their system models, e.g., they consider one scheduled traffic queue of the express type (i.e. non-preemptable), while the other classes (A, B and BE) are low-priority and preemptable. For example, the approach in [102] considers two preemption modes (i.e., with and without hold and release mechanism), but the model could be extended to have multiple scheduled traffic flows that are not necessarily mapped on the highest priority queues. The above analysis techniques are mainly evaluated using synthetic network examples, not fully verified for automotive use cases. The works that evaluated analysis techniques using some realistic automotive case studies are the ones in [85,95,101] and [102]. The lack of evaluations performed in realistic case studies represents one of the major limitations of the current literature, in particular for automotive industries.

Another limitation of the existing works is the lack of support for other traffic shapers, such as the IEEE 802.1Qch Cyclic Queuing and Forwarding (now rolled into the IEEE 802.1Q-2018 standard) and the IEEE 802.1Qcr-2020 Asynchronous Traffic Shaping. Consequently, this research direction is also worth of investigation.

4.3. Integration of analysis techniques into software development frameworks

In practice, the proposed timing analysis techniques should be integrated into model-based software development frameworks and tools to assist the automotive software developers in automatically verifying the timing behavior of the modeled systems. Some notable industrial tools that support automatic timing analysis of automotive software architectures include Rubus-ICE and SymTA/S. Within the context of the above discussion, three core steps in the model-based software development of automotive embedded systems are shown in Fig. 5. The software architecture of a system consisting of software components and their inter-connections is developed in step a. Afterwards, the timing properties and requirements of the system are extracted from the software architecture in step b. There is a step in this process to validate the timing requirements and constraints specified on the system, which is based on the schedulability analysis techniques. The analysis techniques are implemented in the analysis engine, which is integrated in the software architecture modeling tool (step c). The analysis results do not only verify the system' timing requirements, but they also provide a feedback to the system developers regarding the bottlenecks, e.g., which traffic is not schedulable on which port in the network (step d).

There are very few works that have addressed the challenges involved in the integration of schedulability analysis techniques with the model-based software development frameworks and tools for automotive systems. The work in [45] proposed an integration process for TSN networks considering only the credit-based shaper, while the work in [50] complemented the integration with other TSN features, e.g., gate mechanism and frame preemption. The integration methodology in both cases was realized in an industrial tool, Rubus-ICE. Note that the other commercial software development tools, such as SymTA/S, do not expose any detail about the integration techniques for the timing analysis. The methodology for integrating these analysis techniques with industrial or academic software development tools did not get any attention in the TSN research community. This opens a new research opportunity that can gain a momentum as an essential requirement for the commercialization of the proposed theoretical analyses. In addition to this line of research, the TSN network configuration and scheduling algorithms are not integrated into the existing software development tools. This integration requires new software modeling techniques as well as modifications in existing system models to allow the designers to configure TSN networks efficiently.

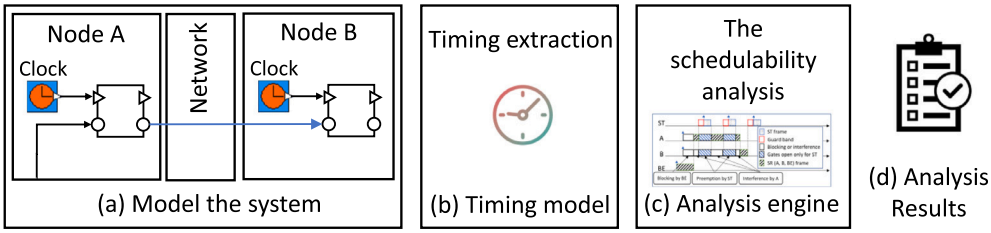

Fig. 5. Software development process.

5. Simulation platforms for TSN

Nowadays, network simulators are widely adopted in industry and academia. Simulation helps designers in decision-making of network configuration problems and in the quantitative performance assessment of a network design. Due to the complexity of TSN networks design, performance evaluation at design time is challenging, as the network behavior is affected by multiple parameters, e.g., the maximum frame size, the network topology, traffic scheduling and routing, etc. For these reasons, the adoption of network simulators is a good option to assess the performance of a TSN network design. In fact, network simulators provide fast network modeling and the ability to test the network with several configuration parameters in a simple way. Network simulation frameworks typically adopt a discrete event simulation model in which events are organized on a time basis. In such a type of simulation frameworks, events are collected in a queue and ordered based on their execution times. The simulation engine cyclically extracts the event with the lowest execution time and processes it. Most of the network simulation frameworks (e.g., OMNeT++ [103,104], NS3 [105], etc.) adopt this approach, as it allows for fast simulation runs.

OMNeT++ is one of the most widely used simulation frameworks for TSN networks in academic settings. Thanks to its open-source license, several TSN simulation models based on OMNeT++ are found in the literature, for both industrial [95,106] and automotive [107,108] scenarios.

The following subsections present an overview of simulation models for TSN. Table 2 summarizes and compares the TSN features implemented by each framework.

5.1. TSN simulation models based on OMNeT++

TSN simulation models are mainly developed using the OMNeT++ platform, as it provides a modular framework. Modules implement specific component behaviors and can be assembled into larger components and models.

The NeSTiNg [109] TSN simulation model, which is based on the INET framework for OMNeT++, is the model suggested by the IEEE TSN Task Group. NeSTiNg models both the enhancements for scheduled traffic mechanisms (as defined in the IEEE 802.1Q-2018) and the Ethernet frame preemption (as defined in the IEEE 802.1Q-2018 and IEEE 802.3br standards). As far as the main TSN standards suitable for automotive applications are concerned, the NeSTiNg model currently lacks of the implementation of (i) the IEEE 802.1AS-2011 or IEEE 802.1AS-2020 standards for clock synchronization, (ii) the per-stream filtering and policing mechanism defined in the IEEE 802.1Q-2018 standard, (iii) the IEEE 802.1CB standard, and (iv) complete support for the CBS. NeSTiNg is based on the INET library, which is the main model library implemented for OMNeT++. Thanks to the INET support, NeSTiNg has a low learning curve for OMNeT++ users. Finally, NeSTiNg design offers ease for extensions supporting clock synchronization models.

Another popular OMNeT++ TSN simulation model is CoRE4INET [110], which is also based on the INET libraries. It models real-time Ethernet protocols like TTEthernet (AS6802 standard) and the main TSN standards. Core4INET was mainly created to support TTEthernet simulation models [111] and later on it was extended to support the IEEE 802.1Q protocols, such as AVB and TSN. The current version of CoRE4INET models the CBS, the Stream Reservation Protocol for network configuration, the enhancements for scheduled traffic, and the per-stream filtering and policing defined in the IEEE 802.1Q-2018 standard. However, CoRE4INET provides no simulation models for Ethernet frame preemption, the IEEE 802.1AS-2011/802.1AS-2020 clock synchronization protocols and the IEEE 802.1CB standard for reliability.

An extension of CoRE4INET is presented in [112], which introduces the clock synchronization model (based on a draft version of the IEEE 802.1AS-2020) and provides an implementation of the enhancements for scheduled traffic.

It is also worth mentioning TSimNet [113], which is an OMNeT-based TSN simulation model that provides per-stream filtering and policing mechanisms, the IEEE 802.1CB mechanisms for frame replication and elimination, and frame preemption according to the IEEE 802.1Qbu and IEEE 802.3br amendments. However, TSimNet does not offer models to simulate the CBS and the enhancements for scheduled traffic.

5.2. Other TSN simulation platforms

Alhough the OMNeT++ framework is widely adopted for simulating TSN networks, there are several TSN simulation models that are based on other frameworks and legacy simulation platforms. In particular, in [114] a simulation model implemented with the OPNET [115] framework (now called Riverbed Modeler) is presented. The model supports the enhancements for scheduled traffic mechanisms, the per-stream filtering and policing mechanisms, the IEEE 802.1CB standard for reliability, and the IEEE 802.1AS clock synchronization protocol. Moreover, it offers reconfiguration support [116]. However, the code of this simulation platform is not publicly available.

TCN TimeAnalysis [117] is a software platform devised for network designers to assess the performance of TSN network configurations. This platform supports the simulation of the CBS and the enhancements for scheduled traffic mechanisms. However, TCN TimeAnalysis is not a network simulation framework, and therefore it does not allow implementing extensions with new features or new protocols.

Thanks to the wide use of open-source TSN simulation models, a large number of TSN standards is currently supported by different simulation models. However, to the best of our knowledge, there is no a single model supporting all of them. An open challenge is therefore integrating all the developed models in a single open-source model (e.g., in the INET framework developed on top of OMNeT++), so as to provide the network designers and researchers with a versatile and common simulation model.

6. Hardware evolution in TSN

A field-programmable gate array (FPGA) is a promising programmable hardware platform for many industrial applications, thanks to low energy consumption, re-programmability, and reconfigurability. Besides, FPGAs offer a better time-to-market compared

Table 2
TSN features implemented in the addressed simulation platforms.

Feature	OMNeT++ models			Other models	
	NeSTiNg	CoRE4INET	TSimNet	OPNET Model	TCN
Credit-Based Shaper		✓			✓
Scheduled Traffic	✓	✓		✓	✓
Clock Sync.				✓	
Frame Preemption	✓		✓		N.A.
Per-stream Filtering and Policing		✓	✓	✓	N.A.
Frame Replication and Elimination			✓	✓	N.A.
Configuration Protocols		✓		✓	N.A.

to Application-Specific-Integrated-Circuit (ASIC) solutions [118]. TSN switches are implemented with FPGA fabrics for the following features:

- Reconfigurability: The design can be easily upgraded and/or replaced with the new TSN protocols. This feature highly reduces the cost of design for any upgrades.
- Parallel designs: FPGAs can be used to implement multiple data processing pipelines in parallel to increase the throughput. This can facilitate developing time-predictable designs.
- Security: FPGA provides hardware-level security through integrated security features, such as differential power analysis protection, cryptography IP-cores, and advanced encryption standards [119].
- Energy efficiency: FPGAs are an energy-efficient solution compared to other computing fabrics, such as Graphics Processing Units (GPUs) and CPUs [120].

It is expected that, when TSN will reach a large diffusion, the ASIC solutions will be needed to address the adoption at low-cost in the automotive large-volume market. To this aim, it is worth noting that FPGAs today are integrable in ASIC as embedded FPGA tiles and direct FPGA to ASIC conversion is made available by FPGA technology providers.

FPGAs are widely used in Ethernet networks [121–123]. However, there are only a few works that have addressed the hardware realization of TSN. A TSN architecture is designed in [124], where the computationally-intensive and time-sensitive functions are implemented in dedicated hardware modules to mitigate the CPU load. The proposed solution allows the network functions to be implemented entirely in hardware or software, based on the dynamic load. The experimental performance results show a significant reduction in the CPU load compared to a fully software implementation. TrustNode [125] is another proposed hardware architecture for low-latency switching and routing. The architecture is the integration of a standard x86-64 processor and an FPGA for Ethernet and TSN switches. It does not only facilitate frequency synchronization across networks, but it is also easily extendable. OpenTSN [126] is an open-source project which enables design-time rapid customization of application-specific TSN system on FPGAs. It is composed of hardware and software components in which the hardware components are designed for switches and network interfaces, while the software components control the underlying devices in a centralized manner. TSN-Builder [127], is another application-specific TSN design framework for customizing resource-efficient switches at design time. However, none of these works have addressed the run-time adaptivity and scalability of the TSN switch fabric.

One open challenge in this regard is to design the TSN switch fabric as a parameterized and modular hardware IP core with hardware description languages (e.g., VHDL or Verilog). The modular and parameterized design can help to update the TSN IP core to match the evolving TSN standards quickly, while enabling engineers to meet the design requirements of the evolving automotive industry. The switch fabric IP core can be equipped with optimized memory modules supporting the present and up-coming types of memory. Using a portable memory controller for data storing might improve performance. The TSN scheduler is implemented as part of the IP core in hardware with interfaces for communication with switch ports and the processor. The switch fabric's run-time reconfigurability is a vital feature that enables the switch to adjust to the application requirements. Such automatic adjustment avoids the customization overhead needed at design time (addressed in [126,127]) and efficiently utilizes the fabric resources with a reasonable energy footprint.

The scalability is another challenging open issue of switch fabrics in general when the number of input and output ports increases. In particular, in automotive systems, each platform requires a different configuration (e.g., topology, number of ports) depending on the design and bandwidth requirements. Conventional crossbar switches are not scalable to support applications with different bandwidth requirements. Such a bottleneck would open the door to studying new switch architectures for TSN systems.

7. Safety and security in TSN

7.1. Safety in TSN

The development of TSN networking technology discussed in previous sections is a key enabling technology to improve safety in high performance computing systems in the automotive domain. Modern automotive applications require high computational performance with strong real-time constraints, e.g., autonomous cars and Advanced Driver Assistance Systems (ADAS). Indeed, automotive ECUs are evolving from simple microcontrollers to complex MPSoCs [128]. However, with the adoption of massively parallel platforms, it is challenging to predict task execution times, because of different processing elements competing for shared on-chip and off-chip resources. Moreover, in-vehicle networks are in a transition from legacy domain-based electronic architectures to zonal architectures. As discussed in Section 2 and in [36] this architecture uses Ethernet (100Base-T1, 1000Base-T1) as the backbone network to integrate zone-level subsystems, such as driver, chassis, body, multimedia, and powertrain subsystems. In these architectures, TSN offers a promising solution for the backbone in-vehicle network.

In the above mentioned networking scenario, domain-based architectures with many simple and separate ECUs will be used for commodity automotive subsystems, while for high-performance tasks (e.g. sensor fusion, navigation and planning) a small number of supercomputers is needed. This evolution poses several challenges in terms of safety, since general-purpose and high-performance computing units suffer from poor timing predictability and high latency for contention when accessing to shared resources (memories, I/O, bus). Moreover, in the case of ADAS, large amounts of data have to be acquired and processed from high-bandwidth sensors (e.g., radars, Lidars, ultrasonics, videocameras, among others) within strict timing constraints. Lastly, high-performance automotive computing units using hypervisor technology [129] will concurrently sustain both safety-critical functions (e.g., for ADAS) and non-critical functions (e.g., infotainment).

While the timing predictability issue for on-chip execution of multiple tasks in multi-core architectures is addressed by frameworks like PRedictable Execution Models (PREM) [130] and by designing corresponding schedulers, the current TSN developments try to overcome the

limits in the state of the art for what off-chip connectivity is concerned. To improve the safety of the platform, the features addressed by the current TSN development efforts, and already discussed in previous Sections, are the following:

- Sustaining data rates of at least 1 Gbps, already standardized in IEEE 802.3bp-2016, and moving to 2.5, 5 and 10 Gbps, in Multi-Gig Automotive Ethernet PHY, approved in June 2020 as IEEE 802.3ch-2020;
- Providing a suite of clock-synchronization protocols that allow end stations and switches to synchronize their local clocks to each other in order to have a common notion of time;
- Reserving bandwidth for real-time functions thanks to the development of the SRP, which allows for the reservation of resources within the bridges on the path between the source and the destination;
- Prioritization and traffic shaping for real-time flows to prevent traffic bursts, by means of the CBS at the output ports of bridges and end nodes, which guarantees bounded latency to real-time classes;
- Schedulability analysis techniques to verify whether the system is schedulable, under given scheduling algorithms such those reviewed in Section 4.1, and that the specified timing requirements are satisfied

7.2. Security in TSN

In automotive embedded systems, the increased complexity of in-vehicle networking and the increased connectivity of vehicles with infrastructures, pedestrians, vehicles (the so-called V2X-vehicle to everything scenario) is increasing the surface exposed to cyber-attacks [131,132]. Hence, in new car generations with distributed control systems, the likelihood and severity of security attacks will increase. A further source of security issues for embedded vehicle systems is due to the trend of updating the firmware/software of ECUs through over-the-air (OTA) remote techniques. For this purpose, new Attribute-Based Encryption (ABE) techniques are proposed in the literature [133] to increase the strength of public key cryptographic schemes for OTA firmware/software updates in automotive embedded systems.

This manuscript focuses on in-vehicle networked embedded systems, hence, the security analysis made in this Section refers to intra-vehicle networking. Unfortunately, classic automotive intra-vehicle networks, like CAN and FlexRay, were not designed taking security in mind, hence, they are characterized by several vulnerabilities and weaknesses, as largely discussed in the literature [3,132,134]. For example, the net-spanning exchange of data through several gateways gives potential access to any vehicular bus from every other bus system. In principle, each LIN, CAN, FlexRay, or MOST controller on-board a vehicle can send messages to any other existing controller, and therefore, without adding extra preventive measures, a single attacked bus system could compromise all in-vehicle communications. As CAN is typically adopted as a backbone bus in traditional vehicles, successful attacks on CAN network may lead to malfunctioning of safety-relevant driving assistance functions.

Current networks have limits in terms of confidentiality, authentication, and availability. The broadcast nature of CAN/CAN-FD and of FlexRay is a risk in terms of confidentiality, since an attacked ECU can be used to monitor all the data passing on the bus. In a CAN network, new ECUs can be added in a plug-and-play fashion, just assigning them a new ID, without modifying the already installed ECUs. Since the data link layer does not provide any signature mechanism, CAN suffers from high risks of authentication vulnerability. The multi-master feature of CAN, with an arbitration based on identifier priority, poses risks in terms of availability. In fact, a hacker can attack a bus and behave as a new ECU, reading all data on the bus and generating false packets. Using a high priority identifier, the malicious ECU can win the arbitration and then continuously send invalid messages, thus making a jamming attack. Even though these invalid frames will be discarded by the receiving controllers, the attack makes the bus unavailable to the other ECUs connected to the bus. The malicious ECU, after reading a message from the bus, can also impersonate another ECU for replay attacks, with a potential for harmful consequences for the vehicle occupants. In a CAN network, due to the lack of signature mechanisms for authentication and transmission encryption, it is easy for an attacker to emulate a protocol-compliant behavior. As a consequence, controllers are not able to verify whether an incoming message comes from an authorized or unauthorized and/or a malicious sender. Controllers just check rules, e.g. bit stuffing, CRC, and data length code consistency, which are not relevant to cybersecurity. Moreover, utilizing the CAN mechanisms for automatic fault localization, malicious CAN frames can cause the disconnection of every single controller by posting several well-directed error flags.

Similar to the CAN automatic fault localization, the bus guardian in FlexRay can be utilized for the well-directed deactivation of any controller by appropriate fake error messages. Attacks on the common time base, which would make the FlexRay network completely inoperative, are also feasible by posting proper malicious SYNC messages on the bus. Moreover, the introduction of well-directed sleep frames deactivates the corresponding power-saving capable FlexRay controllers.

To face all the above issues of the classic in-vehicle networks, some strategies should be implemented in emerging TSN networks. First of all, given the increased available data-rates in TSN (e.g. the Multi-Gigabit automotive Ethernet), critical data in terms of security may be transmitted encrypted. However, complete data encryption to prevent eavesdropping and guarantee message authenticity would cause heavy use of resources and lead to large latency values, possibly clashing with timing requirements. Hence the data to be processed should be divided in security classes with different priorities, where only the critical ones have to be encrypted. Applying integrity and authentication mechanisms, such as Message Authentication Codes (MAC), could be a more efficient approach than complete encryption of all data, for such use cases. This approach is also connected to the separation of the networking domain in multiple zones with different trust levels. This way, non-critical applications (e.g. infotainment) and safety-critical ones (e.g. ADAS) can be concurrently implemented on the same high-performance computing platform by assigning them to domains with different trust levels. Another technique to be adopted in TSN to face security issues is the implementation of intrusion detection algorithms. For example, by exploiting some security limitations of the CAN network (e.g. non-encrypted data, priority declared by the transmitting ECUs, all ECUs being able to always read the traffic data), a malicious software can take the control of an ECU and easily implement some attacks by impersonating another ECU (e.g. man in the middle, jamming, ...). To avoid this issue, the TSN adoption should be combined with the implementation of anomaly and intrusion detection techniques where, in an initial calibration phase (or training phase if AI techniques are adopted), the features of the incoming traffic are extracted and then classified to assess the normal behavior of the network. To this aim, features of both the physical level (e.g. clock skew, voltage levels) and the cyber-level (e.g. data content, message IDs) are analyzed. Once the training phase of the network has been completed, then the incoming traffic can be analyzed, its feature extracted and, if their classification does not match the "normal" one, a warning for anomalies caused by an intrusion can be raised. Considering the architecture of in-vehicle network, the most suitable node to host the intrusion detection system is the gateway. If AI techniques are adopted to detect intrusion by classifying normal vs anomaly behaviors the training phase (needing high computational capability and a huge amount of data) is typically done off-line, while the inference phase (e.g. applying the intrusion detection technique to the incoming traffic) can be done in real time. A proposal from industry (Bosch) for a TSN/AVB compliant intrusion detection system has been recently presented in [135]. Moreover, with the promotion of TSN in the design of autonomous vehicles, security-aware routing and scheduling of real-time automotive applications have recently become novel new research trends, see [136].

8. Conclusions

This work surveyed recent efforts, challenges and opportunities in the field of Time-Sensitive Networking to address the requirements of high-bandwidth and low-latency in-vehicle communications. Among the standards developed by the IEEE TSN task group, the paper pointed at those that are relevant to in-vehicle communications, with the aim of providing the readers with an understanding of the mechanisms and the properties that make them suitable candidates for that scenario. As practical recipes to improve the sustained data-rate and the reliability of in-vehicle infrastructures this paper suggested that new features should be introduced in the in-vehicle networking. One of them is the support for accurate and reliable clock synchronization, which enables a common notion of time in the network. Moreover, other useful properties include the definition of delay bounds for real-time traffic, the ability to reserve resources to the different traffic types and to apply traffic prioritization and traffic shaping. To achieve temporal isolation for jitter-sensitive flows, scheduled traffic support is needed. At lower data rates, frame preemption is also beneficial. Moreover, network management mechanisms are sought for the sake of effectiveness and efficiency. All these add-ons are crucial for the new generations of vehicles, and especially for those characterized by the increasing use of advanced assisted driving or even fully autonomous driving. In addition, to fully exploit their potential in the automotive domain, TSN protocols should be integrated into the state-of-the-art of model-based development processes for automotive embedded systems. As future research direction, one is relevant to the IEEE 802.1Qcr-2020 standard, which is very promising, as it offers bounded latency asynchronous shaping with robustness properties (e.g., integrated policing) and permits compositional timing analysis.

Declaration of competing interest

As this paper is submitted as a state-of-the-art paper for the Special Issue on Parallel, Distributed, and Network-Based Processing in Next-generation Embedded Systems (SI:PDP20), the authors have a conflict of interest with the guest editors of the special issue.

Acknowledgments

The work in this paper is supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA) via the DESTINE, PROVIDENT and INTERCONNECT projects, the Swedish Knowledge Foundation via the FIESTA, HERO and DPAC projects, the Ministry of Education, University and Research (MIUR) Italy via the project Crosslab-Dipartimenti di Eccellenza, and by the University of Catania via the CHANCE project.

References

[1] J. Schroeder, C. Berger, A. Knauss, H. Preenja, M. Ali, M. Staron, T. Herpel, Predicting and evaluating software model growth in the automotive industry, in: IEEE International Conference on Software Maintenance and Evolution, 2017, http://dx.doi.org/10.1109/ICSME.2017.41.

[2] P. Pelliccione, E. Knauss, R. Heldal, S.M. Ågren, P. Mallozzi, A. Alminger, D. Borgentun, Automotive architecture framework: The experience of volvo cars, J. Syst. Archit. 77 (2017) 83–100.

[3] L. Lo Bello, R. Mariani, S. Mubeen, S. Saponara, Recent advances and trends in on-board embedded and networked automotive systems, IEEE Trans. Ind. Inf. 15 (2) (2019).

[4] I. Baas, A glimpse into the future of travel and its impact on marketing, The Drum (2018) URL http://www.thedrum.com/opinion/2016/01/11/glimpse-future-travel-and-its-impact-marketing.

[5] T. Shale-Hester, Driverless cars will require one billion lines of code, says Jaguar Land Rover, AutoExpress (2019) URL https://www.autoexpress.co.uk/car-news/106617/driverlesscars-will-require-one-billion-lines-of-code-says-jlr.

[6] F. Hartwich, Introducing CAN XL into CAN networks, in: 17th CAN in Automation Conference, ICC, 2020.

[7] C. Schanze, Future of CAN from the prospective of an OEM, in: 17th CAN in Automation Conference, ICC, 2020.

[8] J.A. Stankovic, K. Ramamritham, What is predictability for real-time systems? Real-Time Sys. 2 (4) (1990) 247–254.

[9] D. Grund, J. Reineke, R. Wilhelm, A template for predictability definitions with supporting evidence, in: Bringing Theory To Practice: Predictability and Performance in Embedded Systems, in: OpenAccess Series in Informatics, vol. 18, Dagstuhl, Germany, 2011, pp. 22–31.

[10] S. Mubeen, E. Lisova, A.V. Feljan, Timing predictability and security in safety-critical industrial cyber-physical systems: A position paper, in: Applied Sciences–Special Issue "Emerging Paradigms and Architectures for Industry 4.0 Applications", Vol. 10, (3125) 2020, pp. 1–17.

[11] N. Navet, F. Simonot-Lion, In-Vehicle Communication Networks - a Historical Perspective and Review, Technical Report, University of Luxembourg, 2013.

[12] S. Mubeen, J. Mäki-Turja, M. Sjödin, Integrating mixed transmission and practical limitations with the worst-case response-time analysis for Controller Area Network, J. Syst. Softw. 99 (2015) 66–84.

[13] A. Nasrallah, A.S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, H. ElBakoury, Ultra-low latency (ULL) networks: The IEEE TSN and IETF detnet standards and related 5G ULL research, IEEE Commun. Surv. Tutor. 21 (1) (2019) 88–145.

[14] T.A. Henzinger, J. Sifakis, The embedded systems design challenge, in: 14th International Symposium on Formal Methods, 2006.

[15] J. Bezivin, O. Gerbe, Towards a precise definition of the OMG/MDA framework, in: 16th Annual International Conference on Automated Software Engineering, ASE 2001, 2001, pp. 273–280.

[16] T. Vale, I. Crnkovic, E.S. de Almeida, P.A. da Mota Silveira Neto, Y.C. Cavalcanti, S.R. de Lemos Meira, Twenty-eight years of component-based software engineering, J. Syst. Softw. 111 (2016) 128–148.

[17] L. Sha, T. Abdelzaher, K.-E. Årzén, A. Cervin, T.P. Baker, A. Burns, G. Buttazzo, M. Caccamo, J.P. Lehoczky, A.K. Mok, Real time scheduling theory: A historical perspective, Real-Time Syst. 28 (2/3) (2004) 101–155.

[18] N. Feiertag, K. Richter, J. Nordlander, J. Jonsson, A compositional framework for end-to-end path delay calculation of automotive systems under different path semantics, in: CRTS Workshop, 2008.

[19] M. Becker, D. Dasari, S. Mubeen, M. Behnam, T. Nolte, End-to-end timing analysis of cause-effect chains in automotive embedded systems, J. Syst. Archit. 80 (2017) 104–113.

[20] S. Mubeen, J. Mäki-Turja, M. Sjödin, Support for end-to-end response-time and delay analysis in the industrial tool suite: Issues, experiences and a case study, Comput. Sci. Inf. Syst. 10 (2013) 453–482.

[21] R.T. Kolagari, D. Chen, A. Lanusse, R. Librino, H. Lönn, N. Mahmud, C. Mraidha, M.-O. Reiser, S. Torchiaro, S. Tucci-Piergiovanni, T. Wägemann, N. Yakymets, Model-based analysis and engineering of automotive architectures with EAST-ADL: Revisited, Int. J. Concept. Struct. Smart Appl. 3 (2) (2015) 25–70, http://dx.doi.org/10.4018/IJCSSA.2015070103.

[22] M. Becker, S. Mubeen, D. Dasari, M. Behnam, T. Nolte, A generic framework facilitating early analysis of data propagation delays in multi-rate systems (Invited paper), in: 2017 IEEE 23rd International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA, 2017, pp. 1–11.

[23] S. Mubeen, T. Nolte, J. Lundbäck, M. Gålnander, K.-L. Lundbäck, Refining timing requirements in extended models of legacy vehicular embedded systems using early end-to-end timing analysis, in: S. Latifi (Ed.), Information Technology: New Generations, Springer International Publishing, Cham, 2016, pp. 497–508.

[24] S. Mubeen, T. Nolte, M. Sjödin, J. Lundbäck, K.-L. Lundbäck, Supporting timing analysis of vehicular embedded systems through the refinement of timing constraints, Softw. Syst. Model. (2019) 39–69, http://dx.doi.org/10.1007/s10270-017-0579-8.

[25] A. Bucaioni, L. Addazi, A. Cicchetti, F. Ciccozzi, R. Eramo, S. Mubeen, M. Sjödin, MoVES: A model-driven methodology for vehicular embedded systems, IEEE Access 6 (2018) 6424–6445.

[26] P. Thorngren, Keynote talk: Experiences from EAST-ADL use, in: EAST-ADL Open Workshop, Gothenberg, 2013.

[27] AUTOSAR techincal overview, release 4.1, rev. 2, ver. 1.1.0., the AUTOSAR consortium, 2013, http://autosar.org.

[28] S. Mubeen, J. Mäki-Turja, M. Sjödin, Communications-oriented development of component- based vehicular distributed real-time embedded systems, J. Syst. Archit. 60 (2) (2014) 207–220.

[29] S. Mubeen, J. Mäki-Turja, M. Sjödin, Translating timing constraints during vehicular distributed embedded systems development, in: 1st International Workshop on Model-Driven Engineering for Component-Based Software Systems, 2014.

[30] S. Mubeen, M. Gålnander, J. Lundbäck, K.-L. Lundbäck, Extracting timing models from component-based multi-criticality vehicular embedded systems, in: 15th International Conference on Information Technology : New Generations, 2018.

[31] IEEE Std. 802.1Q-2018: IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks, IEEE, 2018.

[32] R. Enns, M. Björklund, A. Bierman, J. Schönwälder, Network Configuration Protocol (NETCONF), in: Request for Comments, (6241) RFC Editor, 2011, RFC 6241.

[33] J. Schönwälder, M. Björklund, P. Shafer, Network configuration management using NETCONF and YANG, IEEE Commun. Mag. 48 (9) (2010) 166–173.

[34] L. Lo Bello, Novel trends in automotive networks: A perspective on Ethernet and the IEEE audio video bridging, in: Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA, 2014, pp. 1–8.

[35] J. Klaus-Wagenbrenner, Zonal EE architecture: Towards a fully automotive ethernet–based vehicle infrastructure, 2019, URL https://ieee802.org/1/files/public/docs2019/dg-zinner-automotive-architecture-evolution-0319-v02.pdf.

[36] G. Xie, Y. Li, Y. Han, Y. Xie, G. Zeng, R. Li, Recent advances and future trends for automotive functional safety design methodologies, IEEE Trans. Ind. Inf. 16 (9) (2020) 5629–5642, http://dx.doi.org/10.1109/TII.2020.2978889.

[37] EAST-ADL domain model specification, V2.1.12, 2013, URL http://www.east-adl.info/Specification/V2.1.12/EAST-ADLSpecification_V2.1.12.pdf.

[38] K. Hänninen, et al., The rubus component model for resource constrained real-time systems, in: IEEE Symposium on Industrial Embedded Systems, 2008.

[39] S. Mubeen, H. Lawson, J. Lundbäck, M. Gålnander, K.L. Lundbäck, Provisioning of predictable embedded software in the vehicle industry: The rubus approach, in: IEEE/ACM 4th International Workshop on Software Engineering Research and Industrial Practice, SER&IP, 2017, pp. 3–9.

[40] S. Sentilles, A. Vulgarakis, T. Bures, J. Carlson, I. Crnkovic, A component model for control-intensive distributed embedded systems, in: International Conference on Component Based Software Engineering, CBSE, 2008, pp. 310–317.

[41] Catalog of specialized CORBA specifications. OMG group, 2011, URL http://www.omg.org/technology/documents/.

[42] X. Ke, K. Sierszecki, C. Angelov, COMDES-II: A component-based framework for generative development of distributed real-time control systems, in: 13th International Conference on Embedded and Real-Time Computing Systems and Applications, 2007.

[43] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, R. Ernst, System level performance analysis - the SymTA/S approach, Comput. Digit. Tech. 152 (2) (2005) 148–166, http://dx.doi.org/10.1049/ip-cdt:20045088.

[44] S. Mubeen, J. Mäki-Turja, M. Sjödin, MPS-CAN analyzer: Integrated implementation of response-time analyses for Controller Area Network, J. Syst. Archit. 60 (10) (2014) 828–841.

[45] M. Ashjaei, S. Mubeen, J. Lundbäck, M. Gålnander, K. Lundbäck, T. Nolte, Modeling and timing analysis of vehicle functions distributed over switched Ethernet, in: 43rd Annual Conference of the IEEE Industrial Electronics Society, 2017.

[46] P. Pop, M.L. Raagaard, S.S. Craciunas, W. Steiner, Design optimisation of cyber-physical distributed systems using IEEE time-sensitive networks, IET Cyber-Phys. Syst.: Theory Appl. 1 (1) (2016) 86–94, http://dx.doi.org/10.1049/iet-cps.2016.0021.

[47] M. Farzaneh, A. Knoll, An ontology-based Plug-and-Play approach for in-vehicle Time-Sensitive Networking (TSN), in: 7th IEEE Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON, 2016, pp. 1–8, http://dx.doi.org/10.1109/IEMCON.2016.7746299.

[48] V. Gavrilut, Design Optimization of IEEE Time-Sensitive Networks (TSN) for Safety-Critical and Real-Time Applications (Ph.D. thesis: 2018–500), Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2018, 2018.

[49] M. Farzaneh, S. Shafaei, A. Knoll, Formally verifiable modeling of in-vehicle time-sensitive networks (TSN) based on logic programming, in: IEEE Vehicular Networking Conference, VNC, 2016, pp. 1–4.

[50] S. Mubeen, M. Ashjaei, M. Sjödin, Holistic modeling of time sensitive networking in component-based vehicular embedded systems, in: 45th Euromicro Conference on Software Engineering and Advanced Applications, SEAA, 2019, pp. 131–139.

[51] The UML Profile for MARTE: Modeling and Analysis of Real-Time and Embedded Systems, OMG Group, 2010, URL http://www.omgmarte.org/.

[52] H. Kopetz, G. Bauer, The time-triggered architecture, Proc. IEEE 91 (1) (2003) 112–126.

[53] W. Steiner, An evaluation of SMT-Based schedule synthesis for time-triggered multi-hop networks, in: 31st IEEE Real-Time Systems Symposium, 2010, pp. 375–384.

[54] S.S. Craciunas, R.S. Oliver, Combined task- and network-level scheduling for distributed time-triggered systems, Real-Time Syst. 52 (2015) 161–200.

[55] H. Kopetz, A. Ademaj, P. Grillinger, K. Steinhammer, The time-triggered Ethernet (TTE) design, in: 8th IEEE international symposium on object-oriented real-time distributed computing, 2005, pp. 22–33.

[56] S.S. Craciunas, R.S. Oliver, M. Chmelík, W. Steiner, Scheduling real-time communication in IEEE 802.1Qbv time sensitive networks, in: Proceedings of the 24th International Conference on Real-Time Networks and Systems, RTNS '16, Association for Computing Machinery, 2016, pp. 183–192.

[57] R. Serna Oliver, S.S. Craciunas, W. Steiner, IEEE 802.1Qbv gate control list synthesis using array theory encoding, in: IEEE Real-Time and Embedded Technology and Applications Symposium, 2018, pp. 13–24.

[58] S.S. Craciunas, R.S. Oliver, An overview of scheduling mechanisms for time-sensitive networks, 2017.

[59] N. Reusch, L. Zhao, S.S. Craciunas, P. Pop, Window-based schedule synthesis for industrial IEEE 802.1Qbv TSN networks, in: 16th IEEE International Conference on Factory Communication Systems, 2020, pp. 1–4.

[60] F. Pozo, G. Rodriguez-Navas, H. Hansson, Schedule reparability: Enhancing time-triggered network recovery upon link failures, in: IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications, 2018, pp. 147–156.

[61] Y. Mi, J. Qu, J. Zhang, M. Yao, A scheduling algorithm of maximize the number of porosity for the time-triggered DIMA system, in: IEEE 3rd International Conference on Electronics Technology, 2020, pp. 68–73.

[62] F. Dürr, N.G. Nayak, No-wait packet scheduling for IEEE time-sensitive networks (TSN), in: Proceedings of the 24th International Conference on Real-Time Networks and Systems, Association for Computing Machinery, New York, NY, USA, 2016, pp. 203–212, http://dx.doi.org/10.1145/2997465.2997494.

[63] M. Pahlevan, R. Obermaisser, Genetic algorithm for scheduling time-triggered traffic in time-sensitive networks, in: IEEE 23rd International Conference on Emerging Technologies and Factory Automation, Vol. 1, 2018, pp. 337–344.

[64] M. Pahlevan, N. Tabassam, R. Obermaisser, Heuristic list scheduler for time triggered traffic in time sensitive networks, SIGBED Rev. 16 (1) (2019) 15–20.

[65] M.L. Raagaard, P. Pop, Optimization Algorithms for the Scheduling of IEEE 802.1 Time-Sensitive Networking (TSN), Technical Report, Technical University of Denmark, 2017.

[66] R. Mahfouzi, A. Aminifar, S. Samii, A. Rezine, P. Eles, Z. Peng, Stability-aware integrated routing and scheduling for control applications in Ethernet networks, in: Design, Automation Test in Europe Conference Exhibition, 2018.

[67] V. Gavrilut, B. Zarrin, P. Pop, S. Samii, Fault-tolerant topology and routing synthesis for IEEE time-sensitive networking, in: 25th International Conference on Real-Time Networks and Systems, 2017, pp. 267–276, http://dx.doi.org/10.1145/3139258.3139284.

[68] V. Gavriluţ, P. Pop, Scheduling in time sensitive networks (TSN) for mixed-criticality industrial applications, in: 14th IEEE International Workshop on Factory Communication Systems, WFCS, 2018, pp. 1–4.

[69] V. Gavrilut, L. Zhao, M.L. Raagaard, P. Pop, AVB-aware routing and scheduling of time-triggered traffic for TSN, IEEE Access 6 (2018) 75229–75243.

[70] B. Houtan, M. Ashjaei, M. Daneshtalab, M. Sjödin, S. Mubeen, Synthesising schedules to improve QoS of best-effort traffic in TSN networks, in: 29th International Conference on Real-Time Networks and Systems, 2021.

[71] V. Gavriluţ, P. Pop, Traffic-type assignment for TSN-based mixed-criticality cyber-physical systems, ACM Trans. Cyber-Phys. Syst. 4 (2) (2020).

[72] M. Farzaneh, S. Kugele, A. Knoll, A graphical modeling tool supporting automated schedule synthesis for time-sensitive networking, in: 22nd IEEE International Conference on Emerging Technologies and Factory Automation, 2017, pp. 1–8.

[73] A.T. d. Santos, B. Schneider, V. Nigam, TSNSCHED: Automated schedule generation for time sensitive networking, in: Formal Methods in Computer Aided Design, 2019, pp. 69–77.

[74] A. Nasrallah, V. Balasubramanian, A. Thyagaturu, M. Reisslein, H. ElBakoury, Reconfiguration algorithms for high precision communications in Time Sensitive Networks, in: IEEE Globecom Workshops, 2019.

[75] A. Nasrallah, A.S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, H. Elbakoury, Performance comparison of IEEE 802.1 TSN time aware shaper (TAS) and asynchronous traffic shaper (ATS), IEEE Access 7 (2019).

[76] Z. Zhou, M.S. Berger, S.R. Ruepp, Y. Yan, Insight into the IEEE 802 . 1 qcr asynchronous traffic shaping in time sensitive network, Adv. Sci. Technol. Eng. Syst. J. 4 (1) (2019) 292–301.

[77] J. Specht, S. Samii, Synthesis of queue and priority assignment for asynchronous traffic shaping in switched Ethernet, in: IEEE Real-Time Systems Symposium, RTSS, 2017, pp. 178–187.

[78] G.C. Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[79] G. Kaczynski, L. Lo Bello, T. Nolte, Deriving exact stochastic response times of periodic tasks in hybrid priority-driven soft real-time systems, in: 2007 IEEE Conference on Emerging Technologies and Factory Automation, EFTA 2007, 2007, pp. 101–110.

[80] J. Imtiaz, J. Jasperneite, L. Han, A performance study of Ethernet Audio Video Bridging (AVB) for industrial real-time communication, in: Conference on Emerging Technologies Factory Automation, 2009.

[81] K. Lee, S. Lee, M.H. Lee, Worst case communication delay of real-time industrial switched Ethernet with multiple levels, IEEE Trans. Ind. Electron. (2006).

[82] J.A.R. De Azua, M. Boyer, Complete modelling of AVB in Network Calculus framework, in: International Conference on Real-Time Networks and Systems, 2014.

[83] J. Diemer, D. Thiele, R. Ernst, Formal worst-case timing analysis of Ethernet topologies with strict-priority and AVB switching, in: International Symposium on Industrial Embedded Systems, 2012.

[84] U.D. Bordoloi, A. Aminifar, P. Eles, Z. Peng, Schedulability Analysis of Ethernet AVB Switches, in: International Conference on Embedded and Real-Time Computing Systems and Applications, 2014.

[85] X. Li, L. George, Deterministic delay analysis of AVB Switched Ethernet networks using an extended trajectory approach, Real-Time Syst. (2017).

[86] J. Cao, P.J.L. Cuijpers, R.J. Bril, J.J. Lukkien, Independent yet Tight WCRT analysis for individual priority classes in Ethernet AVB, in: International Conference on Real-Time Networks and Systems, 2016.

[87] T. Mai, N. Navet, J. Migge, A hybrid machine learning and schedulability analysis method for the verification of TSN networks, in: 15th IEEE International Workshop on Factory Communication Systems, 2019, pp. 1–8.

[88] T.L. Mai, N. Navet, J. Migge, On the use of supervised machine learning for assessing schedulability: Application to ethernet TSN, in: 27th International Conference on Real-Time Networks and Systems, Association for Computing Machinery, 2019, pp. 143–153.

[89] A. FINZI, A. MIFDAOUI, F. FRANCES, E. LOCHIN, Network calculus-based timing analysis of AFDX networks with strict priority and TSN/BLS shapers, in: IEEE 13th International Symposium on Industrial Embedded Systems, SIES, 2018, pp. 1–10.

[90] A. Finzi, A. Mifdaoui, Worst-case timing analysis of AFDX networks with multiple TSN/BLS shapers, IEEE Access 8 (2020) 106765–106784.

[91] D. Maxim, Y.-Q. Song, Delay analysis of AVB traffic in Time-Sensitive Networks (TSN), in: International Conference on Real-Time Networks and Systems, 2017.

[92] D. Thiele, R. Ernst, J. Diemer, Formal worst-case timing analysis of Ethernet TSN's time-aware and peristaltic shapers, in: Vehicular Networking Conference, 2015.

[93] L. Zhao, P. Pop, Z. Zheng, Q. Li, Timing analysis of AVB traffic in TSN networks using network calculus, in: Real-Time and Embedded Technology and Applications Symposium, 2018.

[94] L. Zhao, P. Pop, S. Craciunas, Worst-case latency analysis for IEEE 802.1Qbv time sensitive networks using network calculus, IEEE Access (2018).

[95] M. Ashjaei, G. Patti, M. Behnam, T. Nolte, G. Alderisi, L. Lo Bello, Schedulability analysis of ethernet audio video bridging networks with scheduled traffic support, Real-Time Syst. 53 (4) (2017) 526–577.

[96] G. Alderisi, G. Patti, L. Lo Bello, Introducing support for scheduled traffic over IEEE audio video bridging networks, in: IEEE International Conference on Emerging Technologies and Factory Automation, Cagliari, Italy, 2013.

[97] J. Cao, M. Ashjaei, P.J.L. Cuijpers, R.J. Bril, J.J. Lukkien, An independent yet efficient analysis of bandwidth reservation for credit-based shaping, in: 14th IEEE International Workshop on Factory Communication Systems, 2018, pp. 1–10.

[98] M. Ojewale, P.M. Yomsi, B. Nikolić, Multi-level preemption in TSN: Feasibility and requirements analysis, in: IEEE 23rd International Symposium on Real-Time Distributed Computing, 2020.

[99] M. Ashjaei, M. Sjödin, S. Mubeen, A novel frame preemption model in TSN networks, J. Syst. Archit. 116 (2021).

[100] D. Thiele, R. Ernst, Formal worst-case performance analysis of time-sensitive Ethernet with frame preemption, in: 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation, ETFA, 2016.

[101] L. Zhao, P. Pop, Z. Zheng, H. Daigmorte, M. Boyer, Latency analysis of multiple classes of AVB traffic in TSN with standard credit behavior using network calculus, 2020, ArXiv.

[102] L. Lo Bello, M. Ashjaei, G. Patti, M. Behnam, Schedulability analysis of time-sensitive networks with scheduled traffic and preemption support, J. Parallel Distrib. Comput. 144 (2020) 153–171.

[103] OMNeT++ discrete event simulator, 2020, (Accessed on Sept. 2020), URL https://omnetpp.org/.

[104] A. Varga, OMNeT++, in: K. Wehrle, M. Güneş, J. Gross (Eds.), Modeling and Tools for Network Simulation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 35–59, http://dx.doi.org/10.1007/978-3-642-12331-3_{}3.

[105] NS-3: Network simulator, 2020, (Accessed on Sept. 2020), URL https://www.nsnam.org/.

[106] G. Alderisi, G. Iannizzotto, L. Lo Bello, Towards 802.1 Ethernet AVB for advanced driver assistance systems: a preliminary assessment, in: IEEE 17th Conference on Emerging Technologies Factory Automation, Krakow, Poland, 2012.

[107] A. Sabry, A. Omar, M. Hammad, N. Abdelbaki, AVB/TSN protocols in automotive networking, in: 2020 15th International Conference on Computer Engineering and Systems, ICCES, 2020, pp. 1–7, http://dx.doi.org/10.1109/ICCES51560.2020.9334667.

[108] 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: Vehicular Networking Conference, VNC, Seoul, South Korea, 2012.

[109] J. Falk, D. Hellmanns, B. Carabelli, N. Nayak, F. Dürr, S. Kehrer, K. Rothermel, NeSTiNg: Simulating IEEE time-sensitive networking (TSN) in OMNeT++, in: Proceedings of the 2019 International Conference on Networked Systems (NetSys, Garching b. München, Germany, 2019.

[110] CoRE4INET simulation model, 2020, (Accessed on Sept. 2020), URL https://github.com/CoRE-RG/CoRE4INET.

[111] T. Steinbach, H. Dieumo Kenfack, F. Korf, T.C. Schmidt, An extension of the OMNeT++ INET framework for simulating real-time ethernet with high accuracy, in: Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques, SIMUTools '11, ICST, Brussels, Belgium, Belgium, 2011, pp. 375–382.

[112] J. Jiang, Y. Li, S.H. Hong, A. Xu, K. Wang, A time-sensitive networking (TSN) simulation model based on OMNET++, in: 2018 IEEE International Conference on Mechatronics and Automation, 2018, pp, 643–648.

[113] P. Heise, F. Geyer, R. Obermaisser, TSimNet: An industrial time sensitive networking simulation framework based on OMNeT++, in: 2016 8th IFIP International Conference on New Technologies, Mobility and Security, 2016, pp. 1–5.

[114] H. Baniabdelghany, R. Obermaisser, A. Khalifeh, Extended synchronization protocol based on IEEE802.1AS for improved precision in dynamic and asymmetric TSN hybrid networks, in: 2020 9th Mediterranean Conference on Embedded Computing, 2020, pp. 1–8.

[115] Riverbed modeler–OPNET, 2020, URL https://www.riverbed.com.

[116] M. Pahlevan, J. Schmeck, R. Obermaisser, Evaluation of TSN dynamic configuration model for safety-critical applications, in: 2019 IEEE Intl Conf on Parallel Distributed Processing with Applications, Big Data Cloud Computing, Sustainable Computing Communications, Social Computing Networking, ISPA/BDCloud/SocialCom/SustainCom, 2019, pp. 566–571.

[117] TCN timeanalysis, 2020, URL https://www.timecriticalnetworks.com/products/tcn-timeanalysis.

[118] I. Kuon, J. Rose, Measuring the gap between FPGAs and ASICs, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 26 (2) (2007) 203–215.

[119] S. Trimberger, J.J. Moore, FPGA security: Motivations, features, and applications, Proc. IEEE 102 (8) (2014) 1248–1265.

[120] B. Falsafi, B. Dally, D. Singh, D. Chiou, J.J. Yi, R. Sendag, FPGAs versus GPUs in data centers, IEEE Micro 37 (1) (2017) 60–72.

[121] T. Uchida, Hardware-based TCP processor for Gigabit Ethernet, IEEE Trans. Nucl. Sci. 55 (3) (2008) 1631–1637.

[122] N. Alachiotis, S.A. Berger, A. Stamatakis, Efficient PC-FPGA communication over Gigabit Ethernet, in: 10th IEEE International Conference on Computer and Information Technology, 2010, pp. 1727–1734.

[123] S. Shreejith, P. Mundhenk, A. Ettner, S.A. Fahmy, S. Steinhorst, M. Lukasiewycz, S. Chakraborty, VEGa: A high performance vehicular ethernet gateway on hybrid FPGA, IEEE Trans. Comput. 66 (10) (2017) 1790–1803.

[124] F. Groß, T. Steinbach, F. Korf, T.C. Schmidt, B. Schwarz, A hardware/software co-design approach for Ethernet controllers to support time-triggered traffic in the upcoming IEEE TSN standards, in: IEEE Fourth International Conference on Consumer Electronics Berlin, 2014, pp. 9–13.

[125] C. Liß, M. Ulbricht, U.F. Zia, H. Müller, Architecture of a synchronized low-latency network node targeted to research and education, in: IEEE 18th International Conference on High Performance Switching and Routing, 2017, pp. 1–7.

[126] W. Quan, W. Fu, J. Yan, OpenTSN: an open-source project for time-sensitive networking system development, CCF Trans. Netw. 3 (2020) 51–65, http://dx.doi.org/10.1007/s42045-020-00029-8.

[127] J. Yan, W. Quan, X. Yang, W. Fu, Y. Jiang, H. Yang, Z. Sun, TSN-builder: Enabling rapid customization of resource-efficient switches for time-sensitive networking, in: 2020 57th ACM/IEEE Design Automation Conference (DAC, 2020, pp. 1–6.

[128] D. Reinhardt, U. Dannebaum, M. Scheffer, M. Traub, High performance processor architecture for automotive large scaled integrated systems within the European Processor Initiative Research Project, SAE Int. (2019) (Accessed: 15 Oct. 2020).

[129] A.K. Sundar Rajan, A. Feucht, L. Gamer, I. Smaili, N.D. M., Hypervisor for consolidating real-time automotive control units: Its procedure, implications and hidden pitfalls, J. Syst. Archit. 82 (2018) 37–48.

[130] R. Tabish, R. Mancuso, S. Wasly, et al., A real-time scratchpad-centric OS with predictable inter/intra-core communication for multi-core embedded systems, Real-Time Syst. 55 (2019) 850–888.

[131] C. Corbett, E. Schoch, F. Kargl, F. Preussner, Automotive Ethernet: security opportunity or challenge? in: M. Meier, D. Reinhardt, S. Wendzel (Eds.), Sicherheit 2016, Lecture Notes in Informatics (LNI), Sicherheit, Schutz Und ZuverläSsigkeit, Gesellschaft für Informatik e.V., Bonn, 2016, pp. 45–54.

[132] L. Baldanzi, L. Crocetti, M. Bertolucci, L. Fanucci, S. Saponara, Analysis of cybersecurity weakness in automotive in-vehicle networking and hardware accelerators for real-time cryptography, in: S. Saponara, A. De Gloria (Eds.), Applications in Electronics Pervading Industry, Environment and Society, Springer International Publishing, Cham, 2019, pp. 11–18.

[133] M. La Manna, L. Treccozzi, P. Perazzo, S. Saponara, G. Dini, Performance evaluation of attribute-based encryption in automotive embedded platform for secure software over-the-air update, Sensors 21 (2) (2021) http://dx.doi.org/10.3390/s21020515, URL https://www.mdpi.com/1424-8220/21/2/515.

[134] O. Avatefipour, et al., State-of-the-art survey on in-vehicle network communication CAN-bus security and vulnerabilities, Int. J. Comput. Sci. Netw. 6 (6) (2017).

[135] R. Alves, et al., A glimpse into the future of travel and its impact on marketing, in: IEEE-SA Ethernet and IP at Automotive Technology day (EIPATD), 2019, URL https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/other/eipatd-presentations/2019/D2-01_ALVES-Design_and_Implementation_of_IDS_for_AVB-TSN_Networks.pdf, (Accessed: 15 Oct. 2020).

[136] R. Mahfouzi, A. Aminifar, S. Samii, P. Eles, Z. Peng, Security-aware routing and scheduling for control applications on Ethernet TSN networks, ACM Trans. Des. Autom. Electron. Syst. 25 (1) (2019) http://dx.doi.org/10.1145/3358604.

Mohammad Ashjaei is a senior lecturer in the Complex Real-Time Systems (CORE) and the Heterogeneous Systems - Hardware Software Co-design (HERO) research groups at Mälardalen University in Sweden. Mohammad has received his Ph.D. degree in Computer Science in November 2016 from Mälardalen University. His main research interests include real-time systems, real-time distributed systems, scheduling algorithms on networks and processors, schedulability analysis techniques, resource reservation and reconfiguration mechanisms for real-time networks. He is also giving lectures on various topics related to embedded systems and data communication networks.

Lucia Lo Bello is an Associate Professor at the University of Catania, Italy. She received the M.S. degree in Electronic Engineering and the Ph.D. degree in Computer Engineering in 1994 and 1998, respectively. She was also Guest Professor at Mälardalen University, Sweden (2014) and a Visiting Researcher with the Department of Computer Engineering, Seoul National University, Korea (2000-2001). She authored or coauthored more than 160 technical papers in the area of real-time embedded systems, automotive communications, industrial wireless sensor networks. In 2008 she was the recipient of the IEEE Industrial Electronics Society Early Career Award. She is a member of IEC, CENELEC and CEI and actively participates to standardization activities in the area of industrial communication. She is the Scientific Responsible for the University of Catania for several national and international research projects in the areas of the factory automation and real-time embedded systems and networks. She is Senior Member of the IEEE and Associate Editor of several journals, including the IEEE Transactions on Industrial Informatics, the Journal of Systems Architecture and the Springer Real-Time Systems journal. She is the current IES Secretary.

Masoud Daneshtalab is currently a Professor at Mälardalen University in Sweden. He is co-leading the Heterogeneous System research group at Mälardalen University (http://www.es.mdh.se/hero/). Since 2016 he is in Euromicro board of Director, a faculty member of the HiPEAC network, and a permanent associate editor of Elsevier MICPRO. His research interests include interconnection networks, hardware/software co-design, and deep learning acceleration and optimization. He has published 2 books, 8 book chapters, and over 200 refereed international journals and conference papers.

Gaetano Patti received the M.S. and the Ph.D. degrees from the University of Catania, Catania, Italy, in 2013 and 2017, respectively. He is a Postdoctoral Researcher with the Department of Electrical, Electronics, and Computer Engineering, University of Catania. Since 2012, he has been working in the field of real-time industrial networks, automotive networks, wireless sensor and actuator networks (WSANs), powerline communications, and networks for mobile robot applications. He coauthored more than 30 papers published in the proceedings of international conferences and in important international journals. Dr. Patti is currently a Member of the IEEE Industrial Electronics Society Technical Committee on Factory Automation (Subcommittee on In-Vehicle Embedded Systems)

Sergio Saponara is Full Professor of Electronics at University of Pisa (UNIPI) and IEEE DL. He co-authored about 300 scientific publications and is AE or GE of many journals, including IEEE TII, IEEE VTM IET EL, and IEEE CEMAG. He is director of the school "Enabling Technologies for Industrial IoT", VP of Bachelor and Master Electronic Engineering degrees, and responsible of the CrossLab Industrial IOT and I-CAS lab and leader of UNIPI in the European Processor Initiative (EPI) H2020 project.

Saad Mubeen is an Associate Professor at Mälardalen University, Sweden. He is co-leading the Heterogeneous System research group at the University (http://www.es.mdh.se/hero/). He has previously worked in the vehicle industry as a Senior Software Engineer at Arcticus Systems and as a Consultant for Volvo Construction Equipment, Sweden. He is a Senior Member of IEEE and a Co-chair of the Subcommittee on In-vehicle Embedded Systems within the IEEE IES Technical Committee on Factory Automation. His research focus is on model- and component-based development of predictable embedded software, modeling and timing analysis of in-vehicle communication, and endto-end timing analysis of distributed embedded systems. Within this context, he has co-authored over 135 publications in peer-reviewed international journals, conferences and workshops. He has received several awards, including the IEEE Software Best Paper Award in 2017. He is a PC member and referee for several international conferences and journals respectively. He is a guest editor of IEEE Transactions on Industrial Informatics (TII), Elsevier's Journal of Systems Architecture and Microprocessors and Microsystems, ACM SIGBED Review, and Springer's Computing journal. He has organized and chaired several special sessions and workshops at the international conferences such as IECON, ICIT and ETFA. For more information see http://www.es.mdh.se/staff/280-Saad_Mubeen.