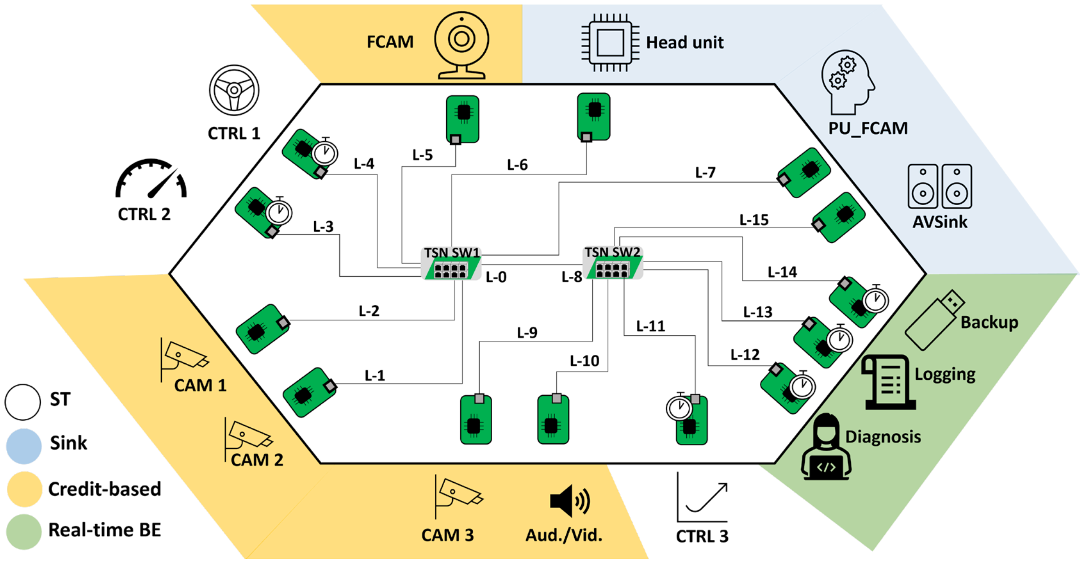
Mälardalen University Licentiate Thesis 316

Configuring and Analysing TSN Networks Considering Low-priority Traffic

Bahar Houtan



MÄLARDALEN UNIVERSITY SWEDEN

Mälardalen University Press Licentiate Theses
No. 316

CONFIGURING AND ANALYSING TSN NETWORKS CONSIDERING LOW-PRIORITY TRAFFIC

Bahar Houtan

2021

MÄLARDALEN UNIVERSITY
SWEDEN

School of Innovation, Design and Engineering

ISBN 978-91-7485-536-4
ISSN 1651-9256
Printed by E-Print AB, Stockholm, Sweden

"Do the difficult things while they are easy and
do the great things while they are small.
A journey of a thousand miles must begin with a single step."
Lao Tzu

Acknowledgements

Great thanks to my supervisors Saad, Mohammad, Masoud, and Mic. So far, I have learned a lot from my supervisors. Through the meetings and discussions with them, I was enlightened with the power of teamwork and how the idea is processed by listening, thinking, mutual understanding, and responding to the team's feed-backs. Thanks for being patient with me, trusting in me, and letting me grow.

I am eternally grateful to my family, especially my parents and brother for their endless support and love; for giving me the motivation to work hard and to never give up. My sincere gratitude goes to all my friends and colleagues in IDT, who were kind to me since the very first day I started at MDH; the Fikas, Guitar sessions, outdoor Yoga, Hackathons, and gatherings are the highlights of my times so far and here. Also, a special thanks to the nice and caring administrative staff at MDH who are always eager to help.

Finally, I find myself more than lucky to have met several amazing, motivating and supportive people in my life. I would like to express my deepest gratitude to all of them.

This work is supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA) via the DESTINE project (grant nr. 2018-02728).

Bahar Houtan
November, 2021,
Västerås, Sweden.

Abstract

The IEEE Time-Sensitive Networking (TSN) standards offer a promising solution to deal with the challenge of supporting high-bandwidth, low-latency, and predictable communication in distributed embedded systems. Although TSN provides a gate mechanism to support the low-jitter transmission of high-priority time-triggered traffic, it also brings complexity to the network design as the configuration of such mechanism together with support for low-priority transmission is non-trivial. Moreover, the combination of the gate mechanism and the Credit-Based Shaper (CBS) mechanism in TSN deals with many configuration parameters, hence finding the most suitable configuration is complex. To avoid this complexity, the Best-Effort (BE) class is sometimes used as an alternative channel to the classes that undergo the CBS mechanism, through which the real-time traffic without strict deadlines is transmitted with a minimum level of Quality of Service (QoS). On the other hand, end stations that operate based on legacy communication standards might not support TSN's traffic shaping mechanisms, hence network designers may need to assign legacy traffic to use the BE class in a TSN network.

To the extent of our knowledge, there is no mechanism to support QoS of the BE traffic class in a TSN network. In order to utilize the BE class as an alternative to other classes, for instance for timing-sensitive traffic, meeting the timing requirements of the traffic must be guaranteed. Therefore, the work in this thesis aims at developing techniques and solutions to support QoS of the lower-priority classes in TSN. In this regard, this work improves the scheduling of high-priority time-triggered traffic to reduce the latency of BE traffic and develops techniques to verify the timing properties of BE traffic considering the impact of all other traffic classes in TSN. Furthermore, the work in this thesis extends the existing end-to-end data-propagation delay analysis for distributed real-time systems based on TSN networks. Finally, the proposed techniques are verified and their applicability is demonstrated using automotive application use cases.

Sammanfattning

Standarderna för IEEE Time-Sensitive Networking (TSN) utgör en lovande lösning för att hantera utmaningarna med att stödja hög bandbredd, låg latens och förutsägbar kommunikation i distribuerade inbyggda system. Även om TSN tillhandahåller en grindmekanism för att stödja låg-jitter-överföring av högprioriterad tidsutlöst trafik, medför det samtidigt ökad komplexitet i nätdesignen, speciellt konfigurationen av mekanismerna tillsammans med stöd för lågprioriterad överföring. Kombinationen av grindmekanismen och mekanismen för Credit-Based Shaper (CBS) i TSN innefattar många konfigurationsparametrar, vilket gör det svårt att hitta den lämpligaste konfigurationen. För att undvika denna komplexitet används klassen Best-Effort (BE) ibland som en alternativ kanal för realtidstrafik utan strikta deadlines. BE överförs med den lägsta servicekvaliteten (QoS). Slutstationer baserade på äldre kommunikationsstandarder kanske inte stöder TSN trafikformningsmekanismer, vilket leder till att konstruktörerna måste tilldela denna trafik till BE -klassen i ett TSN -nätverk. Dock måste användningen av BE för realtidstrafik garanteras när det gäller att uppfylla tidskraven, till exempel svarstider och end-to-end latens. Därför syftar arbetet i denna avhandling till att utveckla tekniker och lösningar för att stödja QoS för de lägre prioriterade klasserna i TSN. Detta arbete förbättrar schemaläggningslösningarna för högprioriterad tidsstyrd trafik för att minska latens för BE-trafik. Förutom dettta, utvecklas i denna avhandling tekniker för att verifiera tidsegenskaperna för BE-trafik med hänsyn till effekterna av alla andra trafikklasser i TSN. Vidare utökar arbetet i denna avhandling den befintliga latensanalysen för end-to-end-dataöverföring i distribuerade realtidssystem baserade på TSN-nätverk. Slutligen verifieras och demonstreras tillämpningen av de föreslagna teknikerna genom användningsfall för fordonstillämningar.

List of Publications

Papers included in this thesis[1]

Paper A: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen. *Synthesising Schedules to Improve QoS of Best-effort Traffic in TSN Networks.* In the 29^{th} International Conference on Real-time Networks and Systems (RTNS 2021).

Paper B: Bahar Houtan, Albert Bergström, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen. *An Automated Configuration Framework for TSN Networks.* In the 22^{nd} IEEE International Conference on Industrial Technology (ICIT 2021).

Paper C: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Sara Afshar, Saad Mubeen. *Schedulability Analysis of Best-effort Traffic in TSN Networks.* In the 26^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2021).

Paper D: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen. *Supporting End-to-end Data-propagation Delay Analysis for TSN Networks.* Mälardalen Real Time Centre (MRTC) Technical Report, Report Nr. MDH-MRTC-339/2021-1-SE, Mälardalen University Press, Sweden, pending submission to a Journal.

[1]The included papers have been reformatted to comply with the thesis layout.

Other Relevant publications[2]

Paper E: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen. "*Work-in-Progress: Investigating the Effects of High Priority Traffic on the Best-Effort Traffic in TSN Networks.*" In IEEE Real-Time Systems Symposium (RTSS 2019).

Paper F: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen. "*Developing Predictable Vehicular Embedded Systems Utilizing Time-sensitive Networking – A Research Plan.*" In the Swedish National Computer Networking Workshop (SNCNW 2019).

[2]Not included in this thesis.

Contents

Part I

Thesis

Chapter 1

Introduction

Advanced functionality and features in many applications of embedded systems require high-bandwidth real-time communication among the embedded processors. For example, considering the automotive domain, where autonomous vehicles use several high data-intensive sensors, e.g., video cameras, radars, lidars, ultrasonic sensors, to mention a few [1]. The large amount of data acquired from these sensors needs to be processed and communicated among the Electronic Control Units (ECUs). For many years, legacy communication protocols, such as Controller Area Network (CAN) [2], have been widely used to provide real-time on-board communication in the automotive domain.

These communication protocols have been reliable for the existing vehicular systems, but they cannot satisfy the requirements of the future's complex vehicular systems. For instance, realization of a higher level of autonomy in the vehicles is conditioned to handling high data-rate sensor readings to gather information of the vehicle environment while driving, and acting in real-time accordingly [3]. Moreover, the communications system need to be flexible to allow accommodation of new functions over time.

Time-Sensitive Networking (TSN) [4, 5, 6] standards have emerged as an attractive solution to meet the high-bandwidth and real-time communication requirements in these advanced applications of embedded systems. These standards enhance Switched Ethernet with deterministic properties. More specifically, TSN supports hard real-time, high bandwidth with low latency and low jitter traffic via a set of features. These features include offline scheduled time-triggered traffic, resource reservation for different classes of traffic, and clock synchronization, among others. Besides, it enables integrating diverse connected services with a wide range of criticality in an automotive network, ranging from the utmost critical functions, such as airbag, steering, and braking control, to non-critical value-added functions, such as video streaming.

1.1 Motivation and Research Challenges

Guaranteeing the predictability of the critical functions, while supporting a high level of Quality of Service (QoS) for non-critical functions is a challenge in the design of automotive systems [1, 7]. The most prominent TSN mechanisms, such as the Time-Aware Shaper (TAS) for Scheduled Traffic (ST) class and Credit-Based Shaper (CBS) for Audio-Video Bridging (AVB) traffic classes, are deemed effective in meeting this challenge.

Moreover, using the Best-Effort (BE) traffic class to communicate traffic with no strict deadlines is beneficial for integration of the legacy traffic that have no strict timing guarantees with TSN networks. This is a cost-effective solution for plug & play enhancement of the existing vehicular networks. Since only TSN replaces the existing networks, and the important legacy end stations can continue to be in use. Then, the designers can configure the TSN network to communicate traffic from legacy end stations through the BE class.

However in practice, adopting the TSN mechanisms using the legacy guidelines raises new challenges. Realizing easy to use design frameworks for TSN is essential for wide adaption of TSN in the industry. The contributions in this thesis aim at improving the design flexibility, and simplify the design process of TSN networks. In this thesis, we address three of these challenges:

1. The configuration complexity of various traffic shaping mechanisms in large scale network scenarios;
2. Integrating TSN networks in the legacy automotive systems; and
3. Timing verification of TSN-based distributed embedded systems.

The first challenge refers to the fact that TSN promises flexibility in coordinating various traffic classes with different criticality levels at the cost of increasing network configuration complexity. In return, it leads to increased complexity in the workload of the network designers, as they need to configure traffic shaping mechanisms. The complexity lies in finding the most suitable settings for each traffic shaping mechanism, which can become untraceable in case of larger topology or utilizing a combination of different shaping mechanisms.

An alternative solution to handle the configuration complexity is to avoid unnecessary assignment of some traffic to pass through shaping mechanisms. Consequently, these traffic are allocated to the rest of the TSN classes, i.e., BE, which has no QoS preservation by the shaper mechanisms. Hence, for the sake of simplifying the development process, in practice only some TSN mechanisms are selected. For example, only ST and BE classes are utilized to avoid complexities of CBS. Therefore, ensuring QoS of BE is highly important in the early applications of TSN in industry.

QoS of the BE class cannot be directly controlled, but a careful configuration of the shaped traffic can indirectly contribute to the overall QoS of the network even for the lowest priority traffic. Besides, this solution provides flexibility to design, configure and develop efficient TSN applications. A less complex configuration process will lead to faster network design and development which in turn affects the time-to-market of the solution, positively. This is convenient to industries that tend to provide new solutions to the users in the earliest time.

The second challenge is the integration of the legacy automotive systems to the TSN-based network. Although integrity of the legacy systems with the modern networks needs to be anticipated by the TSN standards, but the network designers can utilize the existing TSN features in a way to include such systems efficiently. For example, BE class can be utilized for communication of the traffic from such systems.

The third challenge lies in timing analysis over multiple end stations that communicate over TSN, which is an essential challenge in development of reliable TSN-based distributed automotive systems. The ambiguities in the standards make the end-to-end analysis of the data propagation through different TSN classes difficult. Because, calculating the end-to-end delays for the propagation of data through the chain of tasks which utilize different TSN classes requires different approaches. However, this problem can be handled by proper implementation of the end-to-end data-propagation delay analysis.

In summary this thesis takes the BE class as an alternative channel for the traffic that do not have strict timing requirements but require to satisfy a certain level of QoS. We assume that QoS of BE must be guaranteed to meet the timing requirements of the system. The first challenge to realize this is that there is no mechanism to preserve the QoS of BE. Secondly, there are no timing and schedulability analysis techniques for TSN which support BE traffic.

1.2 Research Goal

In this thesis, we present solutions and techniques contributing to the following Research Goal (RG):

Research Goal: Developing techniques to support scheduling and timing verification of TSN networks while considering the Quality of Service in low-priority traffic.

The main RG can be refined into three Sub-Goals (SG_i) as follows.

- **SG**$_1$: To develop scheduling solutions for ST traffic that reduce the response times of BE traffic.
- **SG**$_2$: To develop techniques to calculate bounds on the response times of BE traffic considering the impact of higher priority traffic in TSN networks.
- **SG**$_3$: To develop techniques to verify the end-to-end delays of various traffic classes in TSN networks.

1.3 Research Process

We use the hypothetico-deductive [8] research method for computer science. Figure 1.1 shows the research process.

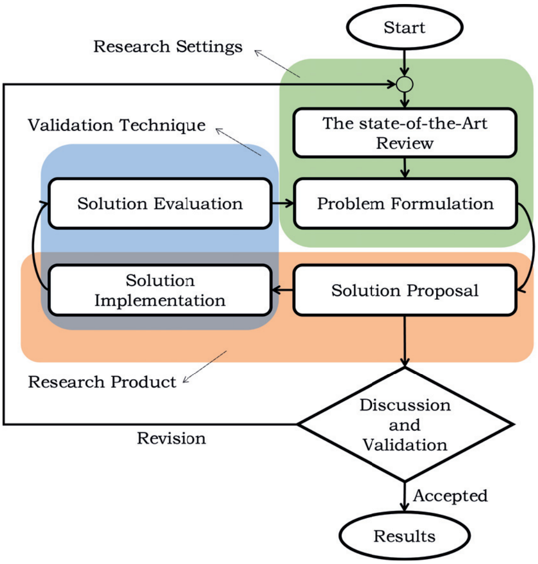


Figure 1.1: Research Methodology.

- **The state-of-the-Art Review**: In this stage, we performed an extensive state-of-the-art review on the schedulability analysis, schedule synthesis, and the existing simulation and modelling approaches.

- **Problem Formulation**: In this stage, we defined the overall research goal and subsequently the research sub-goals according to the research motivation. Besides, the challenges to fulfill the research sub-goals were identified.

- **Solution Proposal**: Novel solutions which address each research sub-goal were discussed and proposed. The implementation challenges for the solutions were identified. Consequently, the most feasible solution was selected to go through for the next step.

- **Solution Implementation** The selected solutions from the previous stage were implemented. Meanwhile, the reusability of the implemented solution was taken into account.

- **Solution Evaluation**: The solution was evaluated by an appropriate use case and approach, such as experimental, simulation, or theoretical analysis methods. We selected the most commonly used use cases and data sets in the vehicular embedded systems domain.

- **Discussion and Validation** This research was instantiated in a close collaboration with industrial partners: Volvo CE[1] and Arcticus Systems[2]. Therefore, in the process of this research, we received the industrial partners' feedback through discussions. The evaluation of solutions were conducted based on automotive use-cases. The evaluation results were discussed and their validity was examined. If the evaluation results were judged to require revision, we traced back to identify the source by continuing the state-of-the-art review. Furthermore, in such case, the research process was reiterated by refining the current solution, or defining a new solution. Finally, if the evaluation results were accepted, the process was finished by publishing the proposed solution.

The chapters of the thesis are organized as follows. Chapter 1 describes motivation for the thesis research and the challenges of developing vehicular embedded systems based on TSN. Then, the research goal and the research process are presented. Chapter 2 is a background review. Chapter 3 presents the thesis contributions and the extent of their applicability to the industrial needs. Chapter 4 gives an overview of the state of the art on schedulability analysis, schedule and gap synthesis, data-propagation delay analysis, simulation and modelling, all in the context of TSN. Finally, Chapter 5 concludes the

[1] https://www.volvoce.com/
[2] https://www.arcticus-systems.com/

thesis and points towards the possible future research. Afterwards, in Part II the papers included in this thesis are presented.

Chapter 2

Background

This chapter describes the background for the work presented in this thesis.

2.1 Embedded Systems

An embedded system is a combination of embedded hardware and embedded software [9]. The embedded hardware has usually limited available electrical and/or mechanical resources, such as memory, processing units, communication interface, sensors and actuators. An embedded hardware can be implemented as a micro-controller, Single-Board Computer (SBC) or Field-Programmable Gate Arrays (FPGA). Besides, an embedded software can be developed in a way to run a simple closed-loop function that utilizes the embedded hardware's available resources, such as in home appliance; or can be developed based on the communication between multiple embedded systems as a part of a bigger system and coordinate between different functionalities, i.e., in a vehicular control system.

2.1.1 Distributed embedded systems

A distributed embedded system is an embedded system in which the system's functionality is distributed over multiple nodes (end stations) that are connected via a bus or a network. An example is shown in Figure 2.1, where the distributed embedded system consists of four nodes: two sensor nodes, one processing node and one actuation node. The nodes are assumed to be connected via an onboard network. In order to provide the desired functionality, the sensor data is sent to the processing node by the sensor nodes. The processing node then performs the desired computations and sends the computed data to the actuation node.

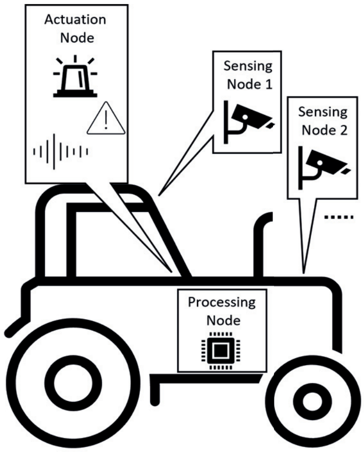


Figure 2.1: An example of distributed embedded system in a vehicle.

2.2 Real-time Systems

A real-time system is a system that is required to provide logically correct response (output) at correct times. In other words, these systems are required to meet all timing requirements that are specified on them. These systems can be categorized into hard and soft real-time systems. In a hard real-time system, missing a timing requirement can result in the system's failure. In contrast, a soft real-time system can tolerate infrequent violation of a timing requirement at the cost of degraded performance or quality of service.

2.3 Real-time Communication

2.3.1 Switched Ethernet

Switched Ethernet is an Ethernet-based network communication protocol where each switch in the network features a limited number of First-In-First-Out (FIFO) queues at the egress ports. Each queue can be associated to a priority, hence a message is transmitted by the queue corresponding to its priority level and transmitted based on its arrival time in the FIFO queue.

2.3.2 Time-sensitive Networking (TSN)

The IEEE TSN standards are based on the Switched Ethernet protocol. These standards provide real-time guarantees for ST by the TAS mechanism and band-width reservation for AVB traffic by the CBS mechanism. In the following sections, these mechanisms are introduced.

2.3.2.1 Time-aware Shaper (TAS)

A significant feature of the TSN standards is the TAS mechanism, an architecture to arbitrate traffic based on the priorities assigned to them. The class types assigned to TSN messages is recognized through a three-bit Priority Code Point (PCP) value in IEEE TSN 802.1Q [10] compatible frame headers. There are eight priority classes as defined by the TSN standards. The three-bit PCP values associated with each of the priorities in TSN are shown in Table 2.1. The value of PCP indicates the priority of traffic class.

Table 2.1: Priority Code Point (PCP).

PCP	Value
000	Background (BK)
001	Best-effort (BE)
010	Excellent-effort (EE)
011	Critical Application (CA)
100	AVB, B
101	AVB, A
110	Internetwork Control (IC)
111	Scheduled Traffic (ST/NC)

The IEEE 802.1Qbv standard [5] provides real-time guarantees for ST class by the TAS mechanism. The standard's architecture dedicates a queue to each TSN traffic class, which are controlled by a transmission selection known as gate mechanism. The example in Figure 2.2, illustrates the block diagram of the TAS mechanism, in which Q_0, Q_1, Q_2 and Q_3 correspond to classes ST or Control Data Traffic (CDT), AVB classes A or B, and BE class, subsequently. The Gate Control List (GCL) block contains fields associated with gates controlling the output of each queue up to the *ST hyperperiod*. The hyperperiod of the ST class is the least common multiple of the periods of all ST messages. In the GCL, 1 represents the "open" state to allow transmission through the queue, and 0 closes the gate to stop the transmission of a corresponding queue.

Moreover, the TSN standards enhance the Ethernet frame headers with the gate mechanism. By including extra four bytes of data to an Ethernet frame's header, as shown in Figure 2.3. The header contains the PCP field that holds the frame's PCP value. This would allow passing TSN frames through their dedicated queues. According to the PCP value specified in the TSN frame

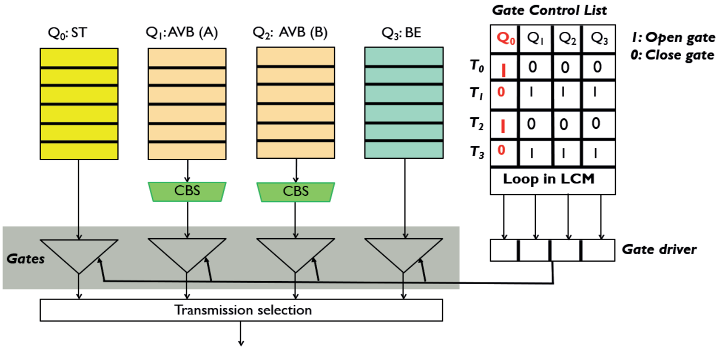


Figure 2.2: Time-Aware Shaper (TAS) block diagram.

header to each queue, the queue gate is enabled, which allows the data stored in the queue to pass through the link.

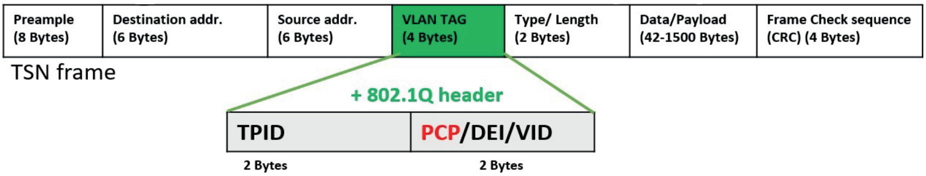


Figure 2.3: Time-Sensitive Networking (TSN) frame.

Traffic types can be assigned to TSN queues according to the system requirements. The most common traffic assignment policy is as follows. The high-priority scheduled time-triggered traffic is assigned to the ST class. This class is reserved for traffic with ultra-low latency demands, e.g., brake or engine control systems. The middle priority traffic is assigned to the AVB class. These classes are reserved for real-time applications with a high-bandwidth utilization or without strict deadlines, i.e., video/audio streaming. The CBS algorithm is applied to this traffic class to prevent bursts of transmission. The low-priority traffic is assigned to the BE class. This traffic class has no real-time guarantees and usually it can be used for periodic software updates, diagnostics, and data logging. Traffic with no TSN standards support, which might have non-strict real-time requirements, is normally allocated to the BE queue.

2.3.2.2 Credit-based Shaper (CBS)

CBS applies to two of the TSN traffic classes: class A and B. Class A has higher priority than class B. The transmission of these classes is arbitrated via credits. Each AVB class has a credit parameter which is defined according to the utilization or manually by the user. The credit can have four different states: idle, sending, stopped, or replenishing. A message from Classes A and B can be forwarded when the credit is zero or positive. In this case, the credit changes to the sending state, where the credit value is decreased with predefined rate, called the "send slope". When the credit is at the idle state, the credit value is increased with a rate specified by the "idle slope". When the message is preempted by other messages belonging to an express class, the credit is at the stopped state and the credit value freezes [11]. After the transmission of the message is completed the credit replenishes back to zero only if it was negative.

2.4 Schedulability Analysis

Schedulability analysis techniques show if all tasks within the real-time system will meet their timing requirements, e.g., deadlines. There are various schedulability analysis techniques, such as Response Time Analysis (RTA) [12], network calculus [13], eligible intervals [14, 15] and machine-learning-based approach [16, 17, 18]. We further elaborate on RTA which is the schedulability analysis technique focused in this thesis. RTA allows to compute worst-case response times of tasks in a node or messages in a network. These response times are then compared with the corresponding timing requirements (deadlines). If the computed worst-case response time of each task or message is smaller than or equal to the corresponding deadline then the system or the network is said to be schedulable respectively.

2.5 Data-propagation Delays

Real-time systems are often modelled with chains of tasks. In order to verify the timing behavior of these chains, not only their end-to-end response times need to be calculated and compared against the corresponding deadlines, but also the end-to-end data-propagation delays (age and reaction) should be calculated and compared with the corresponding age and reaction constraints. Consider a task chain consisting of three tasks τ_1, τ_2 and τ_3, as shown in Figure 2.4. The tasks in this chain are activated independently. The periods of activation for tasks τ_1, τ_2 and τ_3 are 8 ms, 8 ms and 4 ms, respectively. The Worst Case-Execution Time (WCET) of each task is assumed to be 1 ms. For simplicity, we assume that the priority of τ_1 is higher than the priority of τ_2 and the priority of τ_2 is higher than the priority of τ_3. The tasks use register-based communication, i.e., they communicate to each other and to their environment by means of writing data to and reading from the registers. The registers are of non-consuming type. This means that data stays in the register after the reader has read the data. Furthermore, the registers are over-writable, i.e., if the writer is faster than the reader then new data can be overwritten on the previous data in the register before the reader has read the previous data. The data read by τ_1 from Reg-1 corresponds to input of the chain. Similarly, the data written to Reg-3 by τ_3 corresponds to output of the chain.

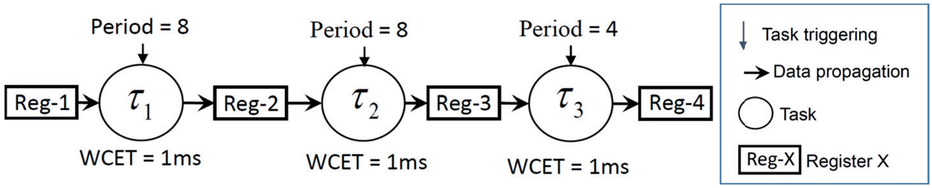


Figure 2.4: An example of a task chain that uses register-based communication.

As the tasks are activated independently and some tasks have different periods, the data can traverse through the chain via multiple paths from the input to the output of the chain as shown in Figure 2.5. These paths are called timed paths. Due to multiple timed paths, there can be various delays that the data can experience from the input to the output of the chain. Two such delays that are common in the automotive systems are *age* and *reaction* delays [19]. The age delay is the time elapsed between the arrival of data at the input and the latest availability of the corresponding data at the output. In the age delay analysis, we are interested to identify the longest time difference between the input data and the last sample of corresponding output data. In addition, the reaction delay corresponds to the earliest availability of the data at the first instance of the output corresponding to the data that *just missed* the read access at the input. Possible age and reaction delays for the chain in Figure 2.4 are shown in Figure 2.5. The age delay is important, in particular, for control applications where freshness of the data is of value. Whereas, the reaction delay is important in the applications where the first reaction to the input is of value.

Although the data-propagation delays are discussed in the context of single-core processors, these delays are equally valid in distributed embedded systems. Let us consider a distributed task chain in a distributed embedded system depicted in Figure 2.6, where two nodes are connected via a network. In this example, the tasks are activated periodically with periods of 6ms and 3ms, respectively. Task τ_1 in Node 1 sends a message to task τ_2 in Node 2 through the network.

Depending on the type of network, we may have different possible data

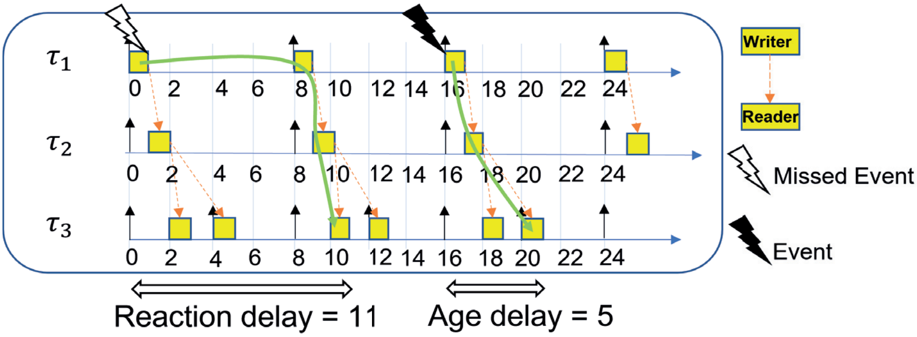


Figure 2.5: Age and Reaction delays in the task chain depicted in Figure 2.4.

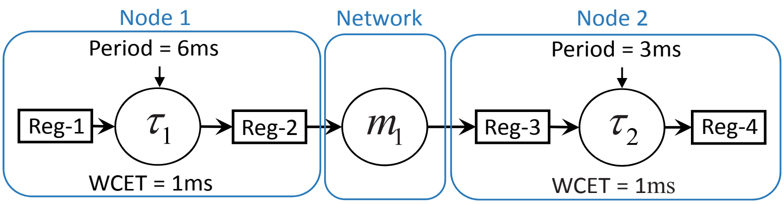


Figure 2.6: A multi-rate chain in a distributed embedded system.

path from the sender task to the receiver task. For example, when the network in not capable of initiating communication independent of the nodes, a message can only be queued for transmission at the network interface when the sending task sends it. This is the case of many network protocols, including CAN [20]. In this case, the message inherits period from its sender task. Furthermore, the data paths in a distributed task chain also depends upon if the network supports synchronization of nodes (e.g., Switched Ethernet) or not (e.g., CAN network). Figure 2.7 shows an execution trace when the nodes are synchronized in the system shown in Figure 2.6. The age and reaction delays in this distributed chain are also identified in Figure 2.7.

A possible execution trace of the distributed task chain in Figure 2.6 when the nodes are not synchronized is shown in Figure 2.8. To create worst case conditions when the nodes are not synchronized, we assume that the message receiving task τ_2 is activated "just before" the message is received at the receiver node. Hence, the current instance of τ_2 will miss the read access of the message. The message will be read by the next instance of τ_2 as shown in Figure 2.8. The corresponding age delay is also identified in this figure. To increase readability, we draw the same execution trace separately for the case of reaction delay in the distributed task chain (shown in Figure 2.6) when the

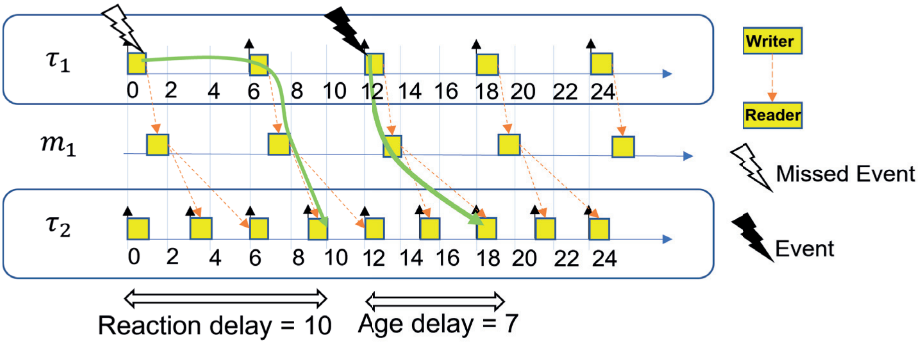


Figure 2.7: A possible execution trace for the distributed embedded system example shown in Figure 2.6 when source and destination nodes are synchronized.

nodes are not synchronized as depicted in Figure 2.9.

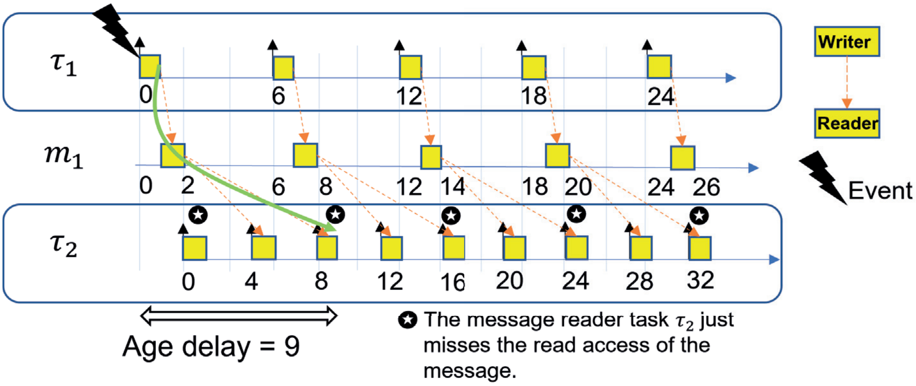


Figure 2.8: A possible execution trace for the distributed embedded system example shown in Figure 2.6 when source and destination nodes are not synchronized (age delay).

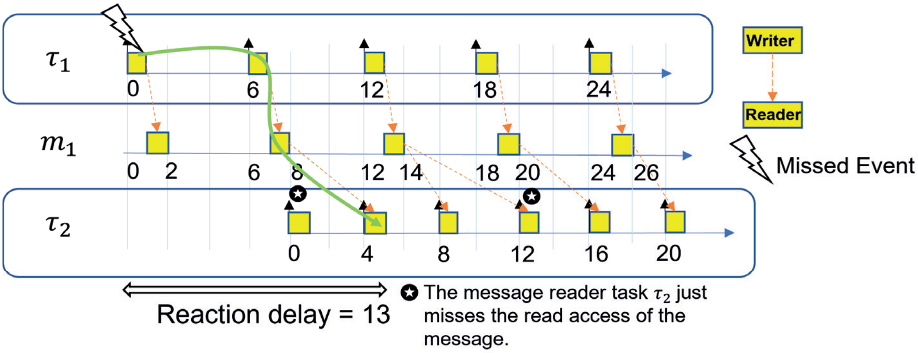


Figure 2.9: A possible execution trace for the distributed embedded system example shown in Figure 2.6 when source and destination are not synchronized (reaction delay).

Chapter 3

Thesis contributions

The research contributions (RC_i) in the thesis that address the research goal are explained in the following sections.

3.1 Technical Contributions

In the following sections, we explain the research contributions of this thesis.

3.1.1 Contribution 1: Providing a solution to schedule TSN network while supporting QoS for BE traffic

The first research contribution (RC_1) in this thesis comprises a technique to generate optimized schedules for TSN. The proposed technique is realized in the form of a schedule generator that synthesizes feasible schedules for hard-real-time ST traffic while considering the QoS for the BE traffic. RC_1 addresses the sub-goal SG_1 by providing a technique that gives the network designers the flexibility to schedule the ST traffic, while assuring lower response times for BE class.

3.1.2 Contribution 2: An automated configuration framework to improve usability and scalability of the TSN networks

In the second research contribution (RC_2), we propose a modular framework to automate offline scheduling in TSN networks to facilitate the design-time and pre-simulation automated network configurations as well as interpretation of the simulations results. The framework consists of four main modules, namely 1) traffic generator; 2) schedule synthesizer; 3) automated configurator and Graphical User Interface (GUI); and 4) results interpretation. Simulation tools can be very beneficial for the development of TSN-based systems at the early stages. Hence, configuration of the TSN simulation tools is as complex as configuration of a TSN network in practice. This framework integrates the optimized TSN schedule generator developed in (RC_1) with a widely used TSN simulation tool (NeSTiNg plug-in). This allows to automatically generate and visualize schedules for TSN switches from different perspectives, i.e, TSN flows, GCL, and the gate states at an egress port. Therefore, the configuration details are easily accessible, and consequently extraction of the response times is facilitated. This research contribution addresses SG_1.

3.1.3 Contribution 3: Extending the existing worst-case response-time analysis for TSN to support BE traffic

We identify that the existing worst-case response-time analysis for TSN does not support the BE traffic class. To address this gap in the state of the art, the third research contribution (RC_3) in this thesis extends the existing analysis [21] to support BE traffic class. The extended analysis addresses the sub-goal SG_2.

3.1.4 Contribution 4: Extending the existing end-to-end data-propagation delay analysis

We identify that the existing end-to-end data-propagation delay analysis [19, 22, 23] does not support the analysis of distributed real-time systems that utilize TSN for network communication. Hence, the fourth research contribution (RC_4) extends the existing analysis to support distributed task chains in real-time systems that encompass all types of TSN traffic classes, including the ST, AVB, and BE traffic classes. The extended analysis incorporates the response-time analysis for the BE traffic class developed in RC_3. This research contribution addresses the sub-goal SG_3.

3.2 Included Papers

The research contributions are encapsulated in four scientific publications which are explained in the following subsections.

3.2.1 Paper A

Title: Synthesising Schedules to Improve QoS of Best-effort Traffic in TSN Networks

Authors: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen.

Status: Published in the 29^{th} International Conference on Real-Time Networks and Systems (RTNS), 2021.

Abstract: The IEEE Time-Sensitive Networking (TSN) standards' amendment 802.1Qbv provides real-time guarantees for Scheduled Traffic (ST) streams by the Time-Aware Shaper (TAS) mechanism. In this paper, we develop offline schedule optimization objective functions to configure the TAS for ST streams, which can be effective to achieve a high Quality of Service (QoS) of lower priority Best-Effort (BE) traffic. This becomes useful if real-time streams from legacy protocols are configured to be carried by the BE class or if the BE class is used for value-added (but non-critical) services. We present three alternative objective functions, namely Maximization, Sparse and Evenly Sparse, followed by a set of constraints on ST streams. Based on simulated stream traces in OMNeT++/INET TSN NeSTiNg simulator, we compare our proposed schemes with a most commonly applied objective function in terms of overall maximum end-to-end delay and deadline misses of BE streams. The results confirm that changing the schedule synthesis objective to our proposed schemes ensures timely delivery and lower delays in BE streams.

3.2.2 Paper B

Title: An Automated Configuration Framework for IEEE 802.1 Time Sensitive Networking

Authors: Bahar Houtan, Albert Bergström, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen.

Status: Published in the 22^{nd} IEEE International Conference on Indsutrial Technology (ICIT), 2021.

Abstract: Designing and simulating large networks, based on the Time Sensitive Networking (TSN) standards, require complex and demanding configuration at the design and pre-simulation phases. Existing configuration and simulation frameworks support only the manual configuration of TSN networks. This hampers the applicability of these frameworks to large-sized TSN networks, especially in complex industrial embedded system applications. This paper proposes a modular framework to automate offline scheduling in TSN networks to facilitate the design-time and pre-simulation automated network configurations as well as interpretation of the simulations. To demonstrate and evaluate the applicability of the proposed framework, a large TSN network is automatically configured, and its performance is evaluated by measuring end-to-end delays of time-critical flows in a state-of-the-art simulation framework, namely NeSTiNg.

3.2.3 Paper C

Title: Schedulability Analysis of Best-effort Traffic in TSN Networks

Authors: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Sara Afshar, Saad Mubeen.

Status: Published in the 26^{th} IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), 2021.

Abstract: This paper presents a schedulability analysis for the Best-Effort (BE) traffic class within Time Sensitive Networking (TSN) networks. The presented analysis considers several features in the TSN standards, including the Credit-Based Shaper (CBS), the Time-Aware Shaper (TAS) and the frame preemption. Although the BE class in TSN is primarily used for the traffic with no strict timing requirements, some industrial applications prefer to utilize this class for the non-hard real-time traffic instead of classes that use the CBS. The reason mainly lies in the fact that the complexity of TSN configuration becomes significantly high when the time-triggered traffic via the TAS and other classes via the CBS are used altogether. We demonstrate the applicability of the presented analysis on a vehicular application use case. We show that a network designer can get information on the schedulability of the BE traffic, based on which the network configuration can be further refined with respect to the application requirements.

3.2.4 Paper D

Title: Supporting End-to-end Data-propagation Delay Analysis for TSN Networks
Authors: Bahar Houtan, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen

Status: Mälardalen Real Time Centre (MRTC) Technical Report, Report Nr. MDH-MRTC-339/2021-1-SE, Mälardalen University Press, Sweden, pending submission to a Journal.

Abstract: End-to-end data-propagation delay analysis allows verification of important timing constraints, such as age and reaction, that are often specified on chains of tasks and messages in real-time systems. We identify that the existing analysis does not support distributed task chains that include the Time-Sensitive Networking (TSN) messages. To this end, this paper extends the existing analysis to allow the end-to-end timing analysis of distributed task chains that include TSN messages. The extended analysis supports all types of traffic in TSN, including the Scheduled Traffic (ST), Audio Video Bridging (AVB), and Best-Effort (BE) traffic. Furthermore, the extended analysis accounts for the synchronization among the end stations that are connected via TSN. The applicability of the analysis is demonstrated using an automotive-application case study.

3.3 Mapping of Research Goal to Contributions and Publications

This section provides the mapping among the research goal, the sub-goals, the research contributions and the publications. As it can be seen in Figure 3.1, there is a one-to-one mapping between the research contributions and publications. Both research contributions RC_1 and RC_2 address sub-goal SG_1. Whereas, the research contributions RC_3 and RC_4 address sub-goals SG_2 and SG_3 respectively.

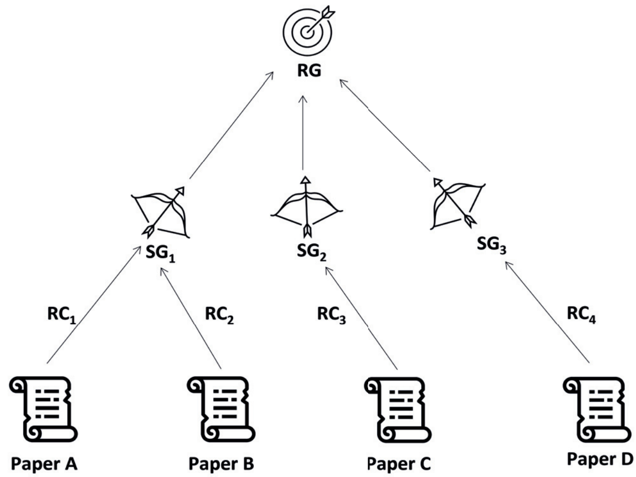


Figure 3.1: Mapping among the research sub-goals, research contributions and publications.

Chapter 4

State-of-the-Art Review

This chapter presents state of the art in: (i) schedulability analysis; (ii) schedule and gap synthesis; (iii) data propagation delay analysis; and (iv) TSN simulation and modeling approaches.

4.1 Schedulability Analysis

This section presents the existing techniques for the schedulability analysis of TSN networks. In general, there are four main techniques, including: (1) response-time analysis; (2) network calculus; (3) the concept of eligible intervals; and (4) TSN verification based on a machine learning technique.

Figure 4.1 shows a timeline of the existing schedulability analysis approaches. According to [7], the focus of the first works in the existing schedulability analysis has evolved from only supporting CBS to more sophisticated models, which include the combination of CBS with time-triggered traffic and frame preemption support. Moreover, the most recent works, e.g., [21], [24] and [25], include the support for preemption in the schedulability analysis for networks based on TSN standards.

4.1.1 Response-time Analysis

The work in [26] is one of the first works aiming at the utilization of Ethernet AVB for in-vehicle communication applications. The proposed response-time analysis only supports CBS. In such a system model, traffic passing through class A is assumed to be the highest priority traffic. Hence, the interference by any traffic by higher priority than class A is not considered.

The response time analysis has been further developed by some later works such as the work in [27] to support the hard real-time time-triggered traffic

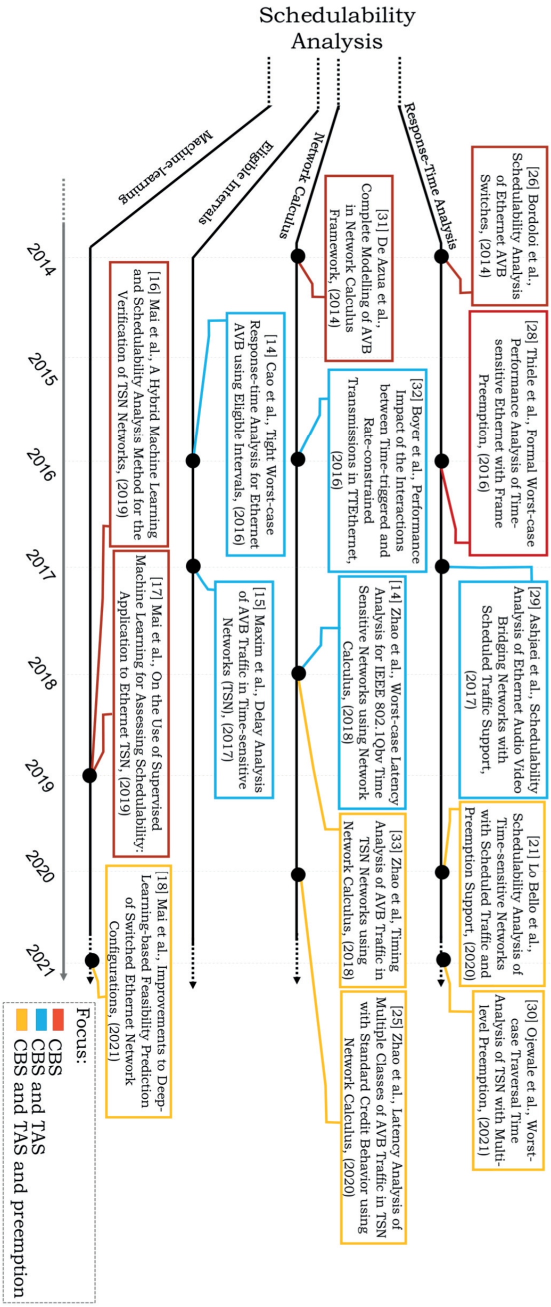


Figure 4.1: Timeline of schedulability analysis techniques for TSN since 2014.

in TSN, which needs to be shaped by deterministic traffic shaper mechanisms such as TAS or the peristaltic shaper. A comparison between these two shapers in [27] showed that shaping the hard real-time traffic based on the peristaltic shaper can cause larger blocking times on the message under analysis. Hence, the work in [28] was extended for TAS to provide a worst-case response-time analysis with the support for preemption.

The work in [29] considered the CBS, and TAS mechanisms in combination, where the TAS mechanism was a variation of the TSN standard. Furthermore, the work in [21] proposed a worst-case response-time analysis that takes into account TAS, CBS, and different variations of frame preemption support (i.e., with and without Hold and Release mechanisms according to the standard). Finally, the work in [30] introduced a worst-case traversal analysis, that is based on a prior technique which introduced a system model allowing multiple preemption levels [24].

4.1.2 Network Calculus

Network calculus is a well-known technique to calculate the worst-case delays in networks. A network calculus analysis for AVB was introduced in [31]. Later, the work in [32] provided an approach to integrate the timing analysis for the periodic time-triggered traffic and the rate-constrained sporadic traffic in TTEthernet. Though TTEthernet has several similarities to TSN, it does not feature the CBS mechanism.

Within the context of TSN, the works in [33, 34] considered the influence of the scheduled traffic on the AVB traffic in preempted mode. The focus of this work is to employ network calculus to find tighter worst-case delay bounds for classes A and B. The work was further developed in [25] with an analysis based on network calculus for the case of having multiple AVB classes in the system model. It included the effects of traffic classes such as ST, A, and B. Finally, the work in [35] presented a solution that utilizes the network calculus to include the credit behaviour while generating ST schedules.

4.1.3 Machine-learning

An interesting approach for verification of TSN networks and their schedulability analysis using machine learning techniques was followed in the works presented in [16, 17]. This approach mainly focuses on verifying the feasibility of TSN configurations by combining a schedulability analysis and a machine learning technique. The work in [18] further improved the approach by applying deep learning-based techniques. However, these techniques are not primarily proposed to provide predictability guarantees for TSN networks, yet are interesting for verifying the TSN network configurations.

4.1.4 Eligible intervals

Besides the aforementioned works, a technique based on the concept of eligible intervals was proposed in the context of AVB network, considering solely CBS mechanism in [14]. Further, the work in [15] used the same concept to analyze the delays of classes A and B in the presence of the ST class. However, using this technique was not used in more complex models in TSN timing analysis.

To sum up, the previous schedulability analysis techniques for TSN addressed several features, but still limited to analysis for classes A and B. This seems intuitive as the BE class was not supposed to be used for traffic with deadline guarantees. However, to avoid the complexity of CBS traffic, the designers of TSN applications might need to assign the BE class instead of CBS as the class for the legacy traffic. In this thesis, we focus on the worst-case response-time analysis, which is a predominant schedulability analysis technique in the industry, in particular, automotive domain [36, 37].

4.2 Schedule and Gap Synthesis

The scheduling problem for ST class of TSN has been perceived from two different aspects. Firstly, schedule synthesis focuses only on the constraints that lead to the deterministic scheduling of the hard-real time ST frames. Similarly, the gap synthesis ensures that ST frames satisfy the real-time requirements, while it also regards other QoS factors of the TSN network. The following sub-sections present the existing works regarding the schedule and gap synthesis perspectives.

4.2.1 Schedule Synthesis

The TSN network designers can synthesize ST schedules and configure the gates by the GCL to ensure the deterministic flow of ST frames. The schedule synthesis approaches can be grouped into two branches as shown in Figure 4.2: (i) Satisfiability Modulo Theorems (SMT) and Optimization Modulo Theorem (OMT), and (ii) heuristics and meta-heuristic approaches. Most of the works in the schedule synthesis use SMT solvers to search all the possibilities for the best suitable ST schedules. SMT solvers are automated theorem provers to prove satisfiability and validity of first-order logical statements by examining each possible combination of variables in the search space. The output of the

SMT solvers is a random value for the SMT variable that satisfies the specified constraints. To improve scalability, some works have either used a new type of optimized SMT solver called OMT or Mixed-Integer Programming (MIP). The OMT Solvers have become popular as they enable optimizing SMT Solver variables based on user-defined objective functions and allow to solve linear optimization problems over satisfiable SMT formulas [38]. In the following, the existing works in the literature are introduced.

Craciunas et al. [39] present an overview of the logical constraints that play a role in generating schedules for the ST frames, such as constraints on the allowed range of the frame offsets: (1) link; (2) stream; (3) end-to-end; and (4) interleaving. These constraints are used by a constraint-based solver, such as SMT/OMT, to generate offsets of the ST frames per link.

Hashemi et al. [40] take a more specific focus on facilitating network configuration and the ease of modelling the TSN network. For example, the work in [41] proposes a step-by-step modelling approach based on logic programming for configuration, verification, and reconfiguration of TSN networks. The proposed model is defined by Prolog language, which is a fact and rule-based language and is proven to be applicable for modelling TSN network traffic. Besides, the work in [42] introduces a graphical modelling tool, TSN Declarative Network Manager (DNM). Furthermore, a method to model TSN networks is proposed in [43, 44], which is compatible with eclipse 4diac framework, an open source infrastructure for distributed industrial process measurement and control systems [45].

The work in [46] proposes a heuristic Genetic Algorithm (GA) scheduling scheme. The motivation to use GA is to decrease the schedule computation time. The results comparison is made between the proposed GA and their Heuristic List Scheduler (HLS). The GA dramatically outperforms HLS. In another work by Pahlevan et al. [47] following the same constraints as their previous work, they prove that computation speed and scalability of HLS outperforms the widely used Integer Linear Programming approaches.

4.2.2 Gap Synthesis

Gap synthesis is a solution for creating porous ST schedules, and can effectively address some QoS challenges in TSN networks. Five different aspects of gap synthesis are introduced and discussed by Steiner et al. [48]: (i) a priori schedule variation: defining a set of constraints on the ST offsets, to force slack slots in the schedules.; (ii) a posteriori schedule variation: a less computationally expensive method to gain slack slots by post-processing the schedules synthesized for ST.; (iii) combined schedule variation: combination

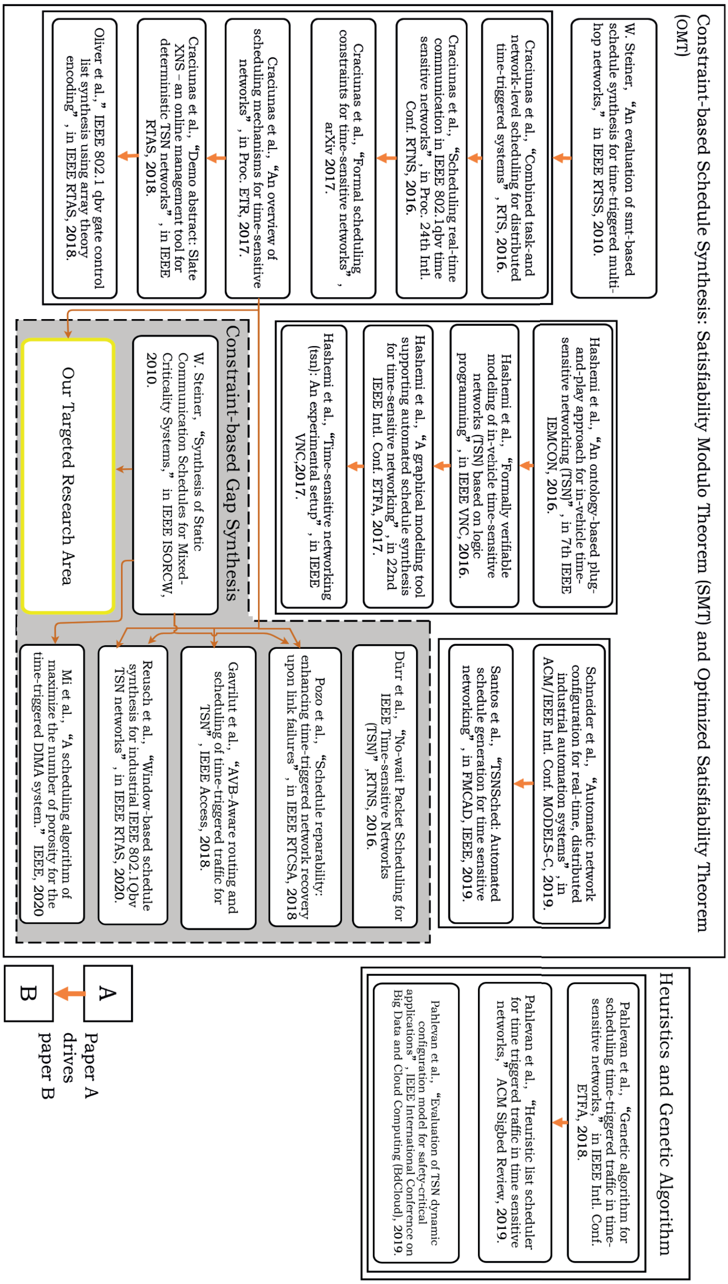


Figure 4.2: Overview of the schedule synthesis approaches

of the first two methods.; (iv) interpretation: discretizing the time-line into equally sized slots regarding the maximum frame size or system synchronization precision. Specifying the transmission ratio of different traffic types based on the time slots.; and (v) Combined Schedule Variation/Integration: the combination of A, B, C, and D in three steps by firstly adding extra constraints, then post-processing, and finally adjusting time slot ratios for time-triggered and rate-constrained traffic.

Existing works apply gap synthesis for different applications. Pozo et al. [49] investigate the generation of porous schedules to achieve a time span for the execution of a link reparation post-processing procedure. In addition, the work by Gavrilut et al. [50] proposed a two-stage approach for scheduling ST traffic considering the best possible routing for the ST traffic, while taking into account the schedulability of the AVB traffic (i.e., A and B). Furthermore, Reusch et al. [51] propose a window-based schedule synthesis approach that maximizes the remaining bandwidth. Since, the proposed solution requires exclusion of the ST deadline miss constraint, it requires a post-processing iterative optimization heuristic to transform schedules to not only generate gaps, but to guarantee satisfaction of ST deadline. A similar work by Yang et al. [52] aims at improving ST schedule with regards to the QoS of the rate-constrained traffic.

4.3 Data Propagation Delay Analysis

Distributed real-time systems are often modelled with chains of tasks and network messages. In order to verify the timing behavior of these chains, not only their end-to-end response times need to be calculated and compared against the corresponding deadlines, but also the end-to-end data-propagation delays (age and reaction) should be calculated and compared with the corresponding age and reaction constraints. These constraints were first discussed in [53] and corresponding analysis was presented in [19, 23]. There are several later works that have also presented the data-propagation delay analysis for different systems models [23, 54]. The work in [55] has considered the end-to-end delay analysis for the distributed embedded systems using CAN as the communication protocol. Moreover, the work in [56] considers the systems that use Ethernet networks. The work in [57] and [23] presented the end-to-end delay analysis for automotive applications, where some of the techniques have been also implemented in tools to support component-based software development, e.g., [55] and [58].

The work in [59, 60] presents some open research problems to address challenges that vehicular embedded software designers face when utilizing the existing industrial tools. For example, in case of modelling, timing verification, simulation and testing and automotive code synthesis utilizing Rubus ICE. Accordingly, different worst-case assumptions in the timing verification of the distributed embedded systems will lead to over/under approximation of the delays. However, in [59] the assumptions are discussed for the CAN protocol. Although in case of TSN standards, different assumptions need to be considered, such as synchronization of the end stations, and the presence of different shaping mechanisms for different TSN classes.

The work in [61], proposes timing models from component-based multi-criticality vehicular embedded systems, which are applicable to be utilized to model and analyze different communication protocols. Although, the proposed scheme in [61] is applicable to be adapted for TSN networks but it has not provided any details on how the model can be utilized based on TSN mechanisms.

Finally, the work in [62] is based on the aforementioned works with the focus on end-to-end analysis at a higher abstraction level according to Rubus Component Model (RCM). RCM was chosen over AUTOSAR due to being lighter in terms of execution time and memory consumption, besides providing more low-level details.

To the best of our knowledge, there is no work in the literature to consider a TSN network and the traffic classes according to the TSN standards in a distributed embedded system with timing analysis of various end-to-end delays. This thesis presents an end-to-end delay analysis considering data chains in TSN networks with all types of traffic classes, including ST, AVB and BE classes.

4.4 TSN Simulation and Modeling

Besides the analytical schedulability analysis techniques, simulation and modelling of TSN networks can also contribute to more efficient configuration design and scheduling of TSN networks [7]. OMNeT++ is an open-source discrete event simulation platform, primarily used for constructing simulations of networks[1]. INET[2] is another widely used open-source simulation framework that models wired, wireless and mobile networks. There are two most commonly used simulation frameworks for TSN built on OMNeT++: (i) Core4INET [63] (ii) NeSTiNg [64]. Firstly, Core4INET allows simulation of TSN networks based on various standards, e.g., IEEE 802.1Q, IEEE P802.1p VLANs and Priorities, IEEE 802.1 AVB, and TTEthernet (AS6802).

[1]https://omnetpp.org/

[2]https://inet.omnetpp.org/

Secondly, NeSTiNg supports several TSN features, including the scheduled traffic (IEEE 802.1Qbv), frame preemption (IEEE Std 802.1Qbu and IEEE Std 802.3br), credit-based shaper (IEEE Std 802.1Qav), and time synchronization (IEEE Std 802.1AS). Though, the aforementioned simulation frameworks are ongoing projects, and the TSN features are partially implemented in them.

Chapter 5

Summary and Future Works

In this thesis we developed techniques and solutions that would lead to supporting the QoS of BE. More specifically, the thesis proposed scheduling solutions that reduce the latency of BE traffic, and presented techniques to verify the timing properties of BE traffic considering the impact of all other traffic classes in TSN. Additionally, this thesis proposed techniques and extensions to the existing end-to-end data-propagation delay analysis for distributed real-time embedded systems to support TSN. The proposed techniques are implemented in a framework, and their applicability is verified and demonstrated by automotive application use cases.

The largest test network that we considered for the evaluation in this thesis took a lot of time for the generation of optimized schedules, i.e., approximately 10 hours. This is due to using constraint programming and limitations of the constraint solver that we applied, Z3 [65]. For future works, we aim at looking into possibility of using faster solutions such as genetic algorithms or simulated annealing to prepare ST schedules for larger networks. The proposed solutions are planned to be compared with the other similar scheduling solutions. Moreover, we have not considered other important parameters which influence the QoS of BE traffic. For instance, AVB classes can also interfere with the BE traffic, while AVB traffic can also be interfered or preempted by the ST class. As a result, the CBS mechanism can be included in the ST scheduling constraints as well as in the objective functions.

Bibliography

[1] L. Lo Bello, R. Mariani, S. Mubeen, and S. Saponara. Recent Advances and Trends in On-Board Embedded and Networked Automotive Systems. *IEEE Transactions on Industrial Informatics*, November 2019.

[2] CAN Specification. Bosch. *Robert Bosch GmbH, Postfach*, 50, 1991.

[3] Christina Rödel, Susanne Stadler, Alexander Meschtscherjakov, and Manfred Tscheligi. Towards Autonomous Cars: The Effect of Autonomy Levels on Acceptance and User Experience. ACM, September 2014.

[4] IEEE. IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks. *IEEE Std. 802.1Q-2018 (Revision of IEEE Std. 802.1Q-2014)*, 2018.

[5] IEEE. *IEEE Std. 802.1Qbv, IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks, Amendment 25: Enhancement for Scheduled Traffic*, 2015.

[6] IEEE. *IEEE Std. 802.1Qbu-2016: IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks, Bridges and Bridged Networks - Amendment 26: Frame Preemption*, 2016.

[7] M. Ashjaei, L. Lo Bello, M. Daneshtalab, G. Patti, S. Saponara, and S. Mubeen. Time-Sensitive Networking in Automotive Embedded Systems: State-of-the-Art and Research Opportunities. *Journal of Systems Architecture*, August 2021.

[8] Gordana Dodig-Crnkovic. Scientific Methods in Computer Science. In *Proceedings of the Conference for the Promotion of Research in IT at New Universities and at University Colleges*, December 2002.

[9] Michael Barr and Anthony Massa. *Programming Embedded Systems: With C and GNU Development Tools*. O'Reilly Media, Inc., 2006.

[10] IEEE. IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks. *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)*, 2018.

[11] Mohammad Ashjaei, Mikael Sjödin, and Saad Mubeen. A Novel Frame Preemption Model in TSN Networks. *Journal of Systems Architecture*, June 2021.

[12] Mathai Joseph and Paritosh K. Pandya. Finding Response Times in a Real-time System. *The Computer Journal*, January 1986.

[13] Jean-Yves Le Boudec and Patrick Thiran. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Springer-Verlag, Berlin, Heidelberg, 2001.

[14] J. Cao, P. J. L. Cuijpers, R. J. Bril, and J. J. Lukkien. Tight Worst-case Response-time Analysis for Ethernet AVB using Eligible Intervals. In *IEEE World Conference on Factory Communication Systems*, May 2016.

[15] Dorin Maxim and Ye-Qiong Song. Delay Analysis of AVB Traffic in Time-sensitive Networks (TSN). In *IEEE Proceedings of the International Conference on Real-time Networks and Systems*, October 2017.

[16] T. L. Mai, N. Navet, J. Migge. A Hybrid Machine Learning and Schedulability Analysis Method for the Verification of TSN Networks. In *International Workshop on Factory Communication Systems*, July 2019.

[17] T. L. Mai, N. Navet, J. Migge. On the use of Supervised Machine Learning for Assessing Schedulability: Application to Ethernet TSN. In *Proceedings of the International Conference on Real-time Networks and Systems*, November 2019.

[18] T. L. Mai and N. Navet. Improvements to Deep-learning-based Feasibility Prediction of Switched Ethernet Network Configurations. In *International Conference on Real-time Networks and Systems*, April 2021.

[19] N. Feiertag, K. Richter, J. Nordlander, and J. Jonsson. A Compositional Framework for End-to-end Path Delay Calculation of Automotive Systems under Different path Semantics. In *Workshop on Compositional Theory and Technology for Real-time Embedded Systems*, December 2008.

[20] ISO 11898-1. Road Vehicles–Interchange of Digital Information–Controller Area Network (CAN) for High-speed Communication, ISO Standard-11898, November 1993.

[21] L. Lo Bello, M. Ashjaei, G. Patti, and M. Behnam. Schedulability Analysis of Time-sensitive Networks with Scheduled Traffic and Preemption Support. *Journal of Parallel and Distributed Computing*, October 2020.

[22] Saad Mubeen, Jukka Mäki-Turja, and Mikael Sjödin. Support for End-to-End Response-time and Delay Analysis in the Industrial Tool Suite:

Issues, Experiences and a Case Study. *Computer Science and Information Systems*, January 2013.

[23] Matthias Becker, Dakshina Dasari, Saad Mubeen, Moris Behnam, and Thomas Nolte. End-to-end Timing Analysis of Cause-Effect Chains in Automotive Embedded Systems. *Journal of Systems Architecture*, October 2017.

[24] M. A. Ojewale, P. M. Yomsi, and B. Nikolić. Multi-level Preemption in TSN: Feasibility and Requirements Analysis. In *IEEE International Symposium on Real-time Distributed Computing*, May 2020.

[25] L. Zhao, P. Pop, Z. Zheng, H. Daigmorte, and M. Boyer. Latency Analysis of Multiple Classes of AVB Traffic in TSN with Standard Credit Behavior using Network Calculus. *IEEE Transactions on Industrial Electronics*, May 2020.

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

[27] D. Thiele, R. Ernst, and J. Diemer. Formal Worst-case Timing Analysis of Ethernet TSN's Time-aware and Peristaltic shapers. In *IEEE Vehicular Networking Conference*, January 2015.

[28] D. Thiele and R. Ernst. Formal Worst-case Performance Analysis of Time-sensitive Ethernet with Frame Preemption. In *IEEE International Conference on Emerging Technologies and Factory Automation*, November 2016.

[29] M. Ashjaei, G. Patti, M. Behnam, T. Nolte, G. Alderisi, and L. Lo Bello. Schedulability Analysis of Ethernet Audio Video Bridging Networks with Scheduled Traffic Support. *Real-Time Systems*, February 2017.

[30] M. A. Ojewale, P. M. Yomsi, and B. Nikolić. Worst-case Traversal Time Analysis of TSN with Multi-level Preemption. *Journal of Systems Architecture*, June 2021.

[31] J. A. R. De Azua and M. Boyer. Complete Modelling of AVB in Network Calculus Framework. In *Proceedings of the International Conference on Real-time Networks and Systems*, October 2014.

[32] M. Boyer, H. Daigmorte, N. Navet, and J. Migge. Performance Impact of the Interactions between Time-triggered and Rate-constrained Transmissions in TTEthernet. In *European Congress on Embedded Real Time Software and Systems* , March 2016.

[33] L. Zhao, P. Pop, Z. Zheng, and Q. Li. Timing Analysis of AVB Traffic in TSN Networks using Network Calculus. In *Real-Time and Embedded Technology and Applications Symposium*, August 2018.

[34] Luxi Zhao, Paul Pop, and Silviu S Craciunas. Worst-case Latency Analysis for IEEE 802.1 Qbv Time-sensitive Networks using Network Calculus. *IEEE Access*, 2018.

[35] Lisa Maile, Kai-Steffen Hielscher, and Reinhard German. Network Calculus Results for TSN: An Introduction. In *IEEE Information Communication Technologies Conference*, May 2020.

[36] Jukka Mäki-Turja. *Engineering Strength Response-Time Analysis — A Timing Analysis Approach for the Development of Real-Time Systems*. PhD thesis, Mälardalen University, May 2005.

[37] S. Mubeen, H. Lawson, J. Lundbäck, M. Gålnander, and K. Lundbäck. Provisioning of predictable embedded software in the vehicle industry: The rubus approach. In *4th IEEE/ACM International Workshop on Software Engineering Research and Industrial Practice*, May 2017.

[38] Nikolaj Bjørner, Anh-Dung Phan, and Lars Fleckenstein. νZ - An Optimizing SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems*. Springer Berlin Heidelberg, 2015.

[39] Silviu S Craciunas, R Serna Oliver, and TC AG. An Overview of Scheduling Mechanisms for Time-sensitive Networks. *Proceedings of the Real-time Summer School, Technical Report*, 2017.

[40] M. H. Farzaneh and 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*, October 2016.

[41] M. H. Farzaneh, S. Shafaei, and A. Knoll. Formally Verifiable Modeling of In-vehicle Time-sensitive Networks (TSN) based on Logic Programming. In *IEEE Vehicular Networking Conference*, December 2016.

[42] Morteza Hashemi Farzaneh, Stefan Kugele, and Alois Knoll. A Graphical Modeling Tool Supporting Automated Schedule Synthesis for Time-sensitive Networking. In *22nd IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE, September 2017.

[43] TSNSched. `https://github.com/ACassimiro/TSNsched`. [Online], 2021-11-18.

[44] Aellison Cassimiro T dos Santos, Ben Schneider, and Vivek Nigam. TSNSCHED: Automated Schedule Generation for Time-sensitive Networking. In *2019 Formal Methods in Computer Aided Design*. IEEE, October 2019.

[45] 4diac - An Open Source Infrastructure. `https://www.eclipse.org/4diac/`. [Online], 2021-11-18.

[46] M. Pahlevan, R. Obermaisser. Genetic Algorithm for Scheduling Time-Triggered Traffic in Time-Sensitive Networks. In *23rd International Conference on Emerging Technologies and Factory Automation*, September 2018.

[47] Maryam Pahlevan, Nadra Tabassam, and Roman Obermaisser. Heuristic List Scheduler for Time-triggered Traffic in Time-sensitive Networks. *ACM Sigbed Review*, February 2019.

[48] Wilfried Steiner. Synthesis of Static Communication Schedules for Mixed-criticality Systems. In *2011 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*. IEEE, March 2011.

[49] F. Pozo, G. Rodriguez-Navas, and H. Hansson. Schedule Reparability: Enhancing Time-triggered Network Recovery upon Link Failures. In *IEEE International Conference on Embedded and Real-time Computing Systems and Applications*, August 2018.

[50] V. Gavriluţ, L. Zhao, M. L. Raagaard, and P. Pop. AVB-aware Routing and Scheduling of Time-triggered Traffic for TSN. *IEEE Access*, November 2018.

[51] N. Reusch, L. Zhao, S. S. Craciunas, and P. Pop. Window-based Schedule Synthesis for Industrial IEEE 802.1qbv TSN Networks. In *IEEE International Conference on Factory Communication Systems*, April 2020.

[52] Y. Mi, J. Qu, J. Zhang, and M. Yao. A Scheduling Algorithm of Maximize the Number of Porosity for the Time-Triggered DIMA System. In *2020 IEEE 3rd Intl. Conference on Electronics Technology*, May 2020.

[53] Friedhelm Stappert, Jan Åke Jönsson, Jürgen Mottok, and Rolf Johansson. A Design Framework for End-To-End Timing Constrained Automotive Applications. In *Embedded Real Time Software and Systems Conference*, May 2010.

[54] Matthias Becker, Dakshina Dasari, Saad Mubeen, Moris Behnam, and Thomas Nolte. Synthesizing Job-level Dependencies for Automotive Multi-rate Effect Chains. In *2016 IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications*, August 2016.

[55] Saad Mubeen, Jukka Mäki-Turja, and Mikael Sjödin. Support for End-to-End Response-Time and Delay Analysis in the Industrial Tool Suite: Issues, Experiences and a Case Study. *Computer Science and Information Systems*, January 2013.

[56] Mohammad Ashjaei, Nima Khalilzad, Saad Mubeen, Moris Behnam, Ingo Sander, Luís Almeida, and Thomas Nolte. Designing End-to-end Resource Reservations in Predictable Distributed Embedded Systems. *Real-Time Systems*, June 2017.

[57] Saad Mubeen, Jukka Mäki-Turja, and Mikael Sjödin. Communications-oriented Development of Component-Based Vehicular Distributed Real-Time Embedded Systems. *Journal of Systems Architecture*, January 2014.

[58] Mohammad Ashjaei, Saad Mubeen, John Lundbäck, Mattias Gålnander, Kurt-Lennart Lundbäck, and Thomas Nolte. Modeling and Timing Analysis of Vehicle Functions Distributed over Switched Ethernet. In *43rd Annual Conference of the IEEE Industrial Electronics Society*, October 2017.

[59] Saad Mubeen, Mattias Gålnander, Alessio Bucaioni, John Lundbäck, and Kurt-Lennart Lundbäck. Timing Verification of Component-based Vehicle Software with Rubus-ICE: End-user's Experience. In *IEEE/ACM 1st International Workshop on Software Qualities and their Dependencies*. IEEE, May 2018.

[60] Saad Mubeen, Harold Lawson, John Lundbäck, Mattias Gålnander, and Kurt-Lennart Lundbäck. Provisioning of Predictable Embedded Software in the Vehicle Industry: The Rubus Approach. In *4th International Workshop on Software Engineering Research and Industry Practice at the 39th International Conference on Software Engineering*. ACM, May 2017.

[61] Saad Mubeen, Mattias Gålnander, John Lundbäck, and Kurt-Lennart Lundbäck. Extracting Timing Models from Component-based Multi-criticality Vehicular Embedded Systems. In *15th International Conference on Information Technology : New Generations*, April 2018.

[62] Saad Mubeen, Thomas Nolte, Mikael Sjödin, John Lundbäck, and Kurt-Lennart Lundbäck. Supporting Timing Analysis of Vehicular Embedded Systems through the Refinement of Timing Constraints. *International Journal on Software and Systems Modeling*, January 2017.

[63] J. Jiang, Y. Li, S. H. Hong, A. Xu, and K. Wang. A Time-sensitive Networking (TSN) Simulation Model-based on OMNET++. In *2018 IEEE International Conference on Mechatronics and Automation*, 2018.

[64] Jonathan Falk, David Hellmanns, Ben Carabelli, Naresh Nayak, Frank Dürr, Stephan Kehrer, and Kurt Rothermel. NeSTiNg: Simulating IEEE Time-sensitive Networking (TSN) in OMNeT++. In *Proceedings of the International Conference on Networked Systems*. IEEE, March 2019.

[65] Z3: Theorem Prover from Microsoft Research. `https://pypi.org/project/z3-solver/`. [Online], 2021-11-18.