Network Slicing in Industry 4.0 Applications

Abstraction Methods and End-to-End Analysis

Kalør, Anders Ellersgaard; Guillaume, Rene; Nielsen, Jimmy Jessen; Mueller, Andreas; Popovski, Petar

Published in:
IEEE Transactions on Industrial Informatics

DOI (link to publication from Publisher):
10.1109/TII.2018.2839721

Publication date:
2018

Document Version
Accepted author manuscript, peer reviewed version

Link to publication from Aalborg University

Citation for published version (APA):
Kalør, A. E., Guillaume, R., Nielsen, J. J., Mueller, A., & Popovski, P. (2018). Network Slicing in Industry 4.0 Applications: Abstraction Methods and End-to-End Analysis. *IEEE Transactions on Industrial Informatics*, *14*(12), 5419 - 5427. Article 8362975. Advance online publication. https://doi.org/10.1109/TII.2018.2839721

Network Slicing in Industry 4.0 Applications: Abstraction Methods and End-to-End Analysis

Anders E. Kalør, *Student Member, IEEE,* René Guillaume, Jimmy J. Nielsen, *Member, IEEE,* Andreas Mueller, *Member, IEEE,* and Petar Popovski, *Fellow, IEEE*

***Abstract*—Industry 4.0 introduces modern communication and computation technologies such as cloud computing and Internet of Things to industrial manufacturing systems. As a result, many devices, machines and applications will rely on connectivity, while having different requirements to the network, ranging from high reliability and low latency to high data rates. Furthermore, these industrial networks will be highly heterogeneous as they will feature a number of diverse communication technologies. Current technologies are not well suited for this scenario, which requires that the network is managed at an abstraction level which is decoupled from the underlying technologies. In this paper, we consider network slicing as a mechanism to handle these challenges. We present methods for slicing deterministic and packet-switched industrial communication protocols which simplifies the manageability of heterogeneous networks with various application requirements. Furthermore, we show how to use network calculus to assess the end-to-end properties of the network slices.**

***Index Terms*—Industry 4.0, industrial communication, network slicing, communication networks, cyber-physical systems, Internet of Things (IoT)**

I. Introduction

THE fourth industrial revolution, known as Industry 4.0, brings cyber-physical systems and Internet of Things (IoT) to industrial manufacturing systems [1], [2]. Furthermore, the number of interconnected physical devices will increase drastically, and they will continuously interact with local cloud services in order to act intelligently and flexibly. This introduces numerous challenges to industrial networks, which have traditionally been very static and strongly isolated [3], [4]. First, it is expected that many new technologies, comprising both wired and wireless connections, will gradually be introduced into production lines resulting in a very heterogeneous network [5]. Secondly, the network will have to serve a wide range of applications with different Quality-of-Service (QoS) requirements, ranging from traditional closed-loop control systems to event-driven sensors and Augmented Reality (AR) displays. For instance, control and alarm systems may require a delivery reliability in the order of $1 - 10^{-9}$ and end-to-end latencies in the range of 0.5–5 ms, while at the same time interactive applications require high data rates and moderate latencies [6], [7]. Finally, the increased system complexity also poses a challenge in managing the network and in particular the end-to-end QoS. This necessitates programmability of the network, as well as a framework for analyzing the end-to-end network characteristics [3], [8].

In this paper, we consider end-to-end network slicing as an architecture for handling the network complexity. Network slicing refers to the use of Software-Defined Networking (SDN) and Network Function Virtualization (NFV) to *slice* a network into logically isolated sub-networks [9]. Each network slice may have certain properties such as latency and reliability guarantees, and appears to applications as a single unified network in an abstract form that is independent of the specific underlying communication technologies. Furthermore, network slices may be constructed with restrictive access control e.g. that traffic in a certain slice cannot leave the internal network. As an example, a network slice may be constructed to offer very low latency communication from a group of wireless sensor devices to a cache in the edge of the network, or a database in the cloud. To guarantee a low latency service, communication and buffer resources along a path are allocated according to the expected aggregate data arrival from the sensors, and the traffic is served with high priority using preemptive packet schedulers. At the same time, in the same physical network, another network slice may reserve resources for specific service-oriented architectures, such as OPC UA [10], or for a high throughput best-effort service between another end-device and the factory cloud, e.g. to allow for downloading firmware updates while being logically isolated from critical control traffic. This is illustrated in Fig. 1.

Network slicing has previously been studied in the context of industrial systems as a way to manage the increasing complexity of manufacturing networks [11]–[13]. However, the concept has mainly been treated from an architectural point of view, and the problem of how to slice the physical networks has received less attention. Outside the industrial domain, network slicing has been studied extensively in the context of Internet protocols and 5G systems, where it is considered an essential technology for handling heterogeneous application requirements [14]–[17]. Unfortunately, these methods are targeted best-effort Internet protocols and cannot be directly applied to industrial networks which are currently dominated

The work of A. E. Kalør, J. J. Nielsen and P. Popovski has been in part supported by the European Research Council (Horizon 2020 ERC Consolidator Grant Nr. 648382 WILLOW) and the Danish Ministry of Higher Education and Science (EliteForsk Award, Grant Nr. 5137-00073B). The work by R. Guillaume and A. Mueller is supported by the Federal Ministry for Education and Research (BMBF) within the project Future Industrial Network Architecture (FIND) (16KIS0571).

A. E. Kalør, J. J. Nielsen, and P. Popovski are with the Department of Electronic Systems, Aalborg University, 9220 Aalborg, Denmark (e-mail: {aek,jjn,petarp}@es.aau.dk).

R. Guillaume and A. Mueller are with Corporate Sector Research and Advanced Engineering, Robert Bosch GmbH, 71272 Renningen, Germany (e-mail: {Rene.Guillaume,Andreas.Mueller21}@de.bosch.com).

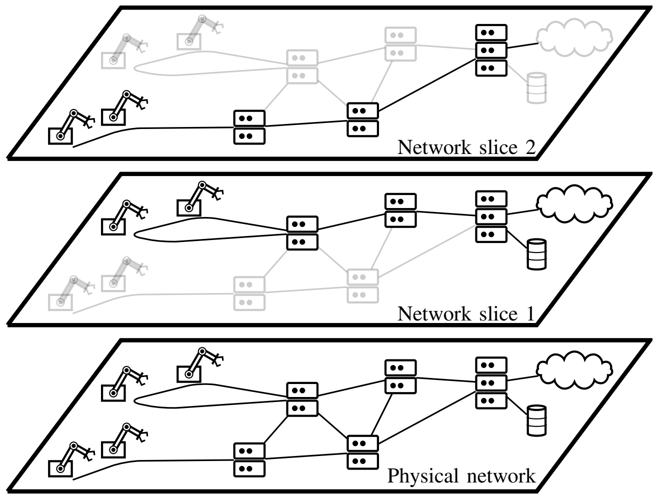


Fig. 1. The network slicing concept where each network slice contains a subset of the physical network resources.

by very specialized and often deterministic communication protocols [18]. However, due to the advantages of SDN in regard to manageability and flexibility, several SDN-based architectures for industrial networks and cyber-physical systems have been proposed recently, see e.g. [8], [19], [20] and references therein. Analysis of end-to-end QoS in SDN-based industrial networks using network calculus has been treated in [21]–[23] which also study algorithms for queue allocation and admission control in priority queue networks. However, while queuing networks are increasingly made available to industrial systems, e.g. using Ethernet TSN [3], it is unlikely that all networks in a manufacturing system will be replaced at once. Hence compatibility with traditional and current technologies, such as fieldbuses and cyclic master/slave industrial Ethernet protocols, as well as interoperability between technologies, is of vital importance [3], [5], [24]. This requires that the problem is approached from an abstraction level which is decoupled from the specific protocols, so that it can function across heterogeneous subsystems.

This paper presents methods for slicing both cyclic and switched industrial communication protocols in an abstract setting which is independent of the specific implementation of the underlying protocols. Using network calculus, we demonstrate how worst-case end-to-end properties of the network slices can be calculated, both for a specific use case and in a general setting, and compare it to simulation results. The remainder of the paper is organized as follows. Section II introduces methods for slicing industrial networks. Section III describes a personalized medicine manufacturing system, which is used as basis for an end-to-end analysis in Section IV. Finally, Section V presents numerical and simulation results, and the paper is concluded in Section VI.

II. Network Slicing for Industry 4.0

Industrial networks are often structured hierarchically as illustrated in Fig. 2 [25]. The factory units contain the individual devices such as actuators, sensors, etc., usually in a master/slave configuration connected through a deterministic and cyclic communication link. The factory units are interconnected by a factory network, which may also connect the devices to a local cloud, or to an external infrastructure such as the Internet. In addition, the factory network may contain computation and storage resources which can be used by the devices in the network. The factory network may be based on real-time Ethernet, in the presence of strict real-time requirements, or regular switched Ethernet and TCP/IP technologies that exploit statistical multiplexing and provide high throughput and interoperability with general-purpose hardware [25], [26]. The communication technologies that are used at the different levels, and even at the different units, are not necessarily interoperable, and may require gateways in order to exchange information [27]. Even if a gateway interconnects two real-time communication technologies, it is likely to introduce queuing since the cycles of the networks and the gateway may not be synchronized.

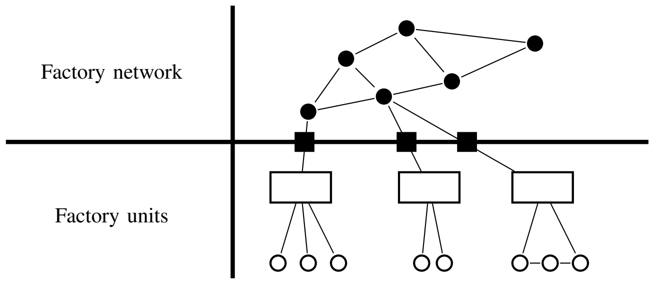


Fig. 2. Hierarchical industrial network architecture. White circles are factory unit devices that are controlled by master devices indicated by rectangles. The master devices are connected to the factory network through gateways (solid squares). The factory network consists of switches and computation/storage nodes (solid circles).

A. Slicing Factory Unit Networks

The factory unit networks are comprised of cyclic master/slave communication technologies with a fixed cycle time, and each cycle contains a number of pre-allocated and fixed-sized telegrams. The components in the factory units usually comprise sensors and actuators which periodically interacts with a controller in a closed-loop control system. However, as manufacturing systems evolve towards cyber-physical systems, it is likely that sensors and actuators will also become more intelligent and generate traffic sporadically, e.g. only transmit when a sensor value exceeds a certain threshold, or in case of anomalies [28]. Furthermore, the requirements to the network are likely to be different, e.g. infrequent aperiodic sensor readings may require higher reliability than periodic sensor readings transmitted several times per millisecond. In the following paragraphs we discuss three slicing schemes which provide different trade-offs in isolation, latency and reliability and utilization: Static telegram allocation, shared telegrams and telegram overwriting. To ease the presentation, we consider a scenario consisting of one deterministic application that transmits R_d telegrams of size N_d in every cycle, and K applications that exhibit a stochastic transmission behavior where the kth application in a given cycle transmits a random number of telegrams, R_k, of fixed size N_k. We do not cover the case where applications transmit telegrams of random size, since this is not a common scenario in practice, and because it requires applications to be able to read the frames before writing their data in order to detect telegram boundaries, which is often not feasible due to the short cycle times. Instead, the telegram size may be considered as an upper bound on the amount of data that need to be transmitted.

In a traditional configuration, the static telegram allocation, applications are assigned telegrams in each cycle according to the data that they will transmit. The deterministic application will then have allocated $R'_\text{d} = R_\text{d}$ telegrams of size N_d, and the reliability is entirely defined by transmission errors introduced by the communication link. For the stochastic applications, suppose R'_k telegrams are allocated to application k. Neglecting transmission and other error sources, and assuming that excess telegrams are not buffered but dropped, the reliability of the scheme, denoted by ζ_k, is the probability that a telegram is among the R'_k that are transmitted:

$$\zeta_k = 1 - \sum_{r=R'_k}^{\infty} \left(1 - \frac{R'_k}{r}\right) \Pr(R_k = r). \tag{1}$$

Furthermore, ignoring propagation delays, the latency experienced by the application is uniform in the cycle duration. This scheme provides a very high degree of isolation due to the separation of resources between applications. However, this comes at the cost of low utilization, since the resources are left empty during a cycle where less than R'_k telegrams are transmitted. Especially transmission patterns that exhibit strong burstiness and have strict requirements to ζ_k, are likely to result in low utilization.

In scenarios where the applications have low transmission rates, it may be beneficial to have multiple applications sharing the same telegrams in order to exploit the statistical multiplexing. However, as argued previously, it is usually not possible for an application to read the contents of a frame before writing to it, which means that there is a risk of overwriting telegrams from other applications. Suppose the telegrams of the K applications have equal length, $N_k = N'$ for all K, and denote the total number of allocated telegrams by $R' = \sum_k R'_k$ and the total number of transmitted telegrams by $R = \sum_k R_k$. We assume that an application writes to a random telegram (uniformly distributed), and that two writes to the same telegram result in failure of both transmissions. Furthermore, we ignore the fact that some applications may be more likely to succeed, e.g. due to being located closer to the master device in a ring topology. Under these assumptions, a transmission is successful if no other application writes to the same telegram:

$$\zeta_k = \frac{1}{\Pr(R > 0)} \sum_{r=1}^{\infty} \left(1 - \frac{1}{R'}\right)^{r-1} \Pr(R = r), \tag{2}$$

where we have normalized to condition on the event that at least one transmission occur ($R > 0$).

While the above scheme increases the utilization by increasing the number of applications that share the same resources, it still results in a low utilization for low arrival rates, especially if the reliability requirements are strict. A way to improve the utilization in these cases is to allow applications with strict reliability requirements to overwrite telegrams allocated to other applications. For instance, a closed-loop control system with periodic feedback may be operational during short interruptions in the feedback. In a system with both closed-loop control and applications with sporadic transmissions, the sporadic transmissions can overwrite the feedback transmissions without causing failure of the control system. Suppose $R'_\text{d} = R_\text{d}$ telegrams of size N_d are allocated to the control traffic. We retain the notation of N and R introduced in the previous scheme, and assume that $N \leq N_\text{d}$ so that the telegram size of the sporadic traffic can be contained within a control traffic. Under this scheme, the probability that an arbitrary control telegram is overwritten by any of the R sporadic telegrams is given by

$$\zeta_\text{d} = \sum_{r=0}^{\infty} \left(1 - \frac{1}{R_\text{d}}\right)^{r} \Pr(R = r). \tag{3}$$

Similar to the previous scheme, a sporadic transmission also fails if two or more transmissions overwrite the same control telegram:

$$\zeta_k = \frac{1}{\Pr(R > 0)} \sum_{r=1}^{\infty} \left(1 - \frac{1}{R_\text{d}}\right)^{r-1} \Pr(R = r). \tag{4}$$

B. Factory Networks

Factory networks are typically based on conventional (switched) or real-time Ethernet. In conventional Ethernet the frames are queued at each link, while real-time Ethernet usually supports both strict real-time traffic and conventional Ethernet traffic through different channels. Since the methods covered in the previous section can be directly applied to the strict real-time functionality of the protocols, we here focus on conventional Ethernet traffic, but we do not distinguish between whether it is served by conventional Ethernet links or by the Ethernet channel in real-time Ethernet technologies. We slice the networks by assigning a dedicated egress queue to each individual slice in each hop in the network. To handle the latency requirements in industrial scenarios, we employ a strict priority scheduler between the queues. Priority schedulers are widely supported by networking hardware through management APIs such as OpenFlow [29]. Several frameworks for analyzing the end-to-end latency in queuing networks have been proposed in literature, most notably queuing theory [30], stochastic network calculus (SNC) [31] and deterministic network calculus (DNC) [32]. Queuing theory seeks probabilistic quantities of queues such as the waiting time distribution and the mean number of items in the queue. While such results, in particular distributions, are useful for analyzing the end-to-end guarantees in a network, the calculations are often intractable for traffic and service time distributions that are not memoryless, which limits the range of applicable scenarios. SNC extends the number of applicable scenarios by seeking bounds on the latency distribution instead of exact results. However, while many traffic arrival distributions can be used in SNC, it falls short when traffic models are deterministic, such

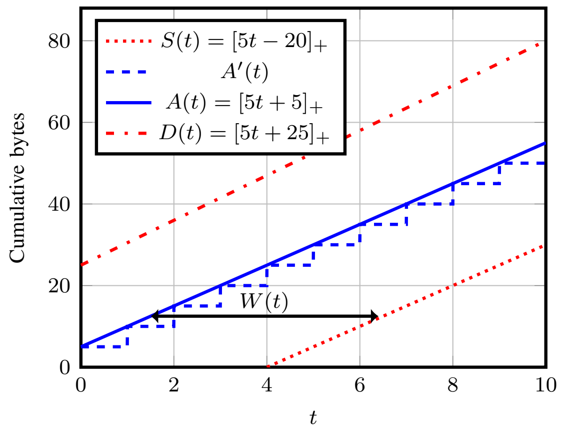


Fig. 3. Graphical representation of the quantities in DNC. Curves are chosen for illustrative purposes.

as periodic arrivals that are prominent in industrial networks. DNC provides worst-case bounds on the latency, and supports a wide range of traffic models as long as the arrivals can be upper bounded by some function. As the factory unit networks enforces an upper bound on the arrivals in each cycle, DNC is well suited for industrial applications. For this reason, we will focus on modeling the latency using DNC. For simplicity, we restrict the presentation of DNC to the case where traffic arrival are bounded by affine functions. A thorough and more general discussion of DNC can be found in e.g. [32], [33].

The theory of DNC is based on arrival and service curves which bound the cumulative number of bytes that arrive to and are served by a queue. Example curves, chosen for illustrative purposes, are illustrated in Fig. 3, where the dashed line $A'(t)$ is a periodic arrival curve that is bounded by the solid affine curve $A(t)$, and $S(t)$ and $D(t)$ are the service and departure curves, respectively. $W(t)$ is the waiting time experienced by the data arriving at time $t = 1.5$. We assume that the cumulative number of bytes generated by an application per time unit, say seconds, is upper bounded by an affine arrival function

$$A(t) = [\alpha t + \beta]_+, \tag{5}$$

where $[x]_+ = \max(x, 0)$. The parameters $\alpha \geq 0$ and $\beta \geq 0$ are referred to as rate and burst parameters, respectively. For example, the arrivals from an application that generates a frame of size N periodically every M seconds would be bounded by the affine function parameterized by $\alpha = N/M$ and $\beta = N$.

Similar to the arrivals, we assume that the rate at which bytes are extracted from a queue (typically modeling the serialization of frames) is lower bounded by the affine service function

$$S(t) = [\sigma t - \rho]_+, \tag{6}$$

where $\sigma \geq 0$ is the minimum service rate and $\rho \geq 0$ accounts for service given to bursts of higher priority. For the system to be stable we require $\sigma \geq \alpha$. The waiting time, i.e. the time a frame spends in the queue while waiting for service, is bounded by [33]

$$W(t) \leq \frac{\rho + \beta}{\sigma}. \tag{7}$$

If frames from multiple applications arrive to the same queue, the aggregate arrival rate is parameterized by $\alpha = \sum_j \alpha_j$ and $\beta = \sum_j \beta_j$.

A single queue as considered above usually does not suffice when a wide range of application requirements are present, since some frames may need to receive higher priority. To analyze prioritization queuing using DNC, we need to introduce the concept of leftover service, which refers to the minimum service that is left to a flow after flows of higher priorities have been served. To simplify the analysis, we ignore the impact of frame blocking. This simplification may be justified by frame preemption mechanisms, e.g. introduced in Ethernet TSN [34]. If two arrival curves $A_1(t) = [\alpha_1 t + \beta_1]_+$ and $A_2(t) = [\alpha_2 t + \beta_2]_+$ are served by the same server $S(t)$, but $A_1(t)$ is prioritized higher than $A_2(t)$, then the leftover service for $A_2(t)$ is given as [33]

$$S_{\text{lo}}(t) \geq [(\sigma - \alpha_1)t - \rho - \beta_1]_+ , \tag{8}$$

while $A_1(t)$ is served by the entire service given by $S(t)$. Notice that DNC is not restricted to prioritization queueing, and several other scheduling schemes can be analyzed as well [33].

Finally, in order to obtain results across several queues in series, we need to obtain bounds on the departures of a queue. It can be shown that the departures, $D(t)$, from a system are bounded by [32]

$$D(t) \leq \left[\alpha t + \beta + \frac{\alpha\rho}{\sigma}\right]_+ , \tag{9}$$

which is again an affine bounded arrival function, but with the burst increased by $\alpha\rho/\sigma$.

III. Use Case: Personalized Medicine Manufacturing

To study how network calculus can be applied to analyze the end-to-end delays in a real manufacturing system, we define a simple Industry 4.0 use case for a system that produces personalized medicine. The system is described in an abstract way, which is independent of the specific communication technologies that are used. The system consists of C identical master/slave factory units which communicate using a cyclic industrial protocol. Each factory unit network is connected to a gateway that provides access to a factory network based on conventional switched Ethernet. The factory network is connected to the cloud through a switch (Fig. 4). We denote the three link levels by L1, L2 and L3.

Each of the C factory units contains a number of devices which are used to dispense a drug product into a container and to supervise the process. The amount of dispensed drug is gathered from the cloud in real-time, and the weight of the resulting product is stored in the cloud. The drug is dispensed by a pipetting machine mounted on a robotic arm, which is part of a closed-loop control system executed by the master device. Furthermore, the individual devices may raise alarms if an anomaly is detected, and the process can be followed from a AR display, which continuously streams video to the cloud.

TABLE I
END-TO-END REQUIREMENTS CONSIDERED IN THE USE CASE

Application	Source	Dest.	Type	Mean period	Size	Latency req.	Reliability req.	Priority
Control feedback	Robot	Master	Periodic	1 ms	128 B	1 ms	$1-10^{-4}$	1
Device alarms	Any	Cloud	Poisson	60 s	32 B	5 ms	$1-10^{-6}$	1
Patient info request	Master	Cloud	Periodic	200 ms	128 B	10 ms	$1-10^{-4}$	2
Scale readings	Scale	Cloud	Periodic	200 ms	512 B	100 ms	$1-10^{-6}$	3
AR stream	AR display	Cloud	Periodic	20 ms	20 kB	20 ms	$1-10^{-2}$	3

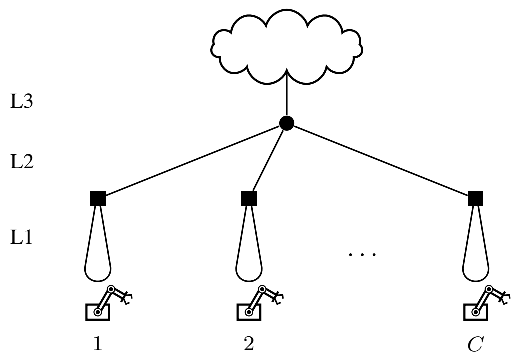


Fig. 4. Personalized medicine manufacturing network consisting of C factory units and three link levels: L1, L2 and L3.

We assume that the master/slave network within the factory units are based on 100 Mbit Ethernet with a cycle time of $\tau = 1$ ms, and that the factory network is based on switched 100 Mbit Ethernet with prioritization queuing. The frame delivery reliability of both the industrial and switched Ethernet links is $\zeta_{\text{link}} = 1 - 10^{-9}$. For simplicity, we only consider uplink traffic from the devices to the master device or the cloud. The traffic characteristics and latency/reliability requirements are listed in Table I, along with the considered priority scheme where a lower number represents higher priority. We note that finding an optimal priority scheme is a complex problem, and that the priorities used here are chosen mainly for illustrative purposes.

The use case is very general as it has only few assumptions to the underlying communication network: That the factory units deploy a cyclic master/slave communication protocol with reserved resources, and that the factory network is based on switched Ethernet with preemption and prioritization queuing. As a result, the same use case could apply both to wired and wireless factory units, and to several communication technologies.

IV. END-TO-END LATENCY ANALYSIS

In this section, we apply the network slicing methods described in Section II to the use case, and analyze end-to-end latency and reliability properties. For the reasons of clarity, our discussion and analysis will be at first focused to the device alarms and the patient info requests in the concrete use case. We will then provide directions for generalizing the analysis by applying suitable abstractions to other use cases. However, a formal description of the abstraction process is outside of the scope for this article.

For simplicity, we ignore propagation delays as well as potential overhead added by protocol headers in the network. To deal with the complexity involved in modeling multi-hop networks, we employ a conservative approach and consider each link independently. This method is conservative since bursts are considered at each link, and hence it does not comply with the "pay burst only once" principle [32]. Throughout the analysis we use milliseconds as the time unit.

A. Use Case Analysis

We first consider the device alarms which are transmitted to the cloud. Suppose that we at the device level, L1, allocate R_{alarms} bytes in each cycle according to one of the proposed schemes so that we obtain a worst-case latency equal to the cycle time, τ, of 1 ms with probability $\zeta_{\text{L1,alarms}}$. R_{alarms} enforces a limit to the number of bytes that arrive to link L2 at the factory network. Specifically, at most R_{alarms} bytes arrive in each cycle, so that the arrivals are bounded by $A_{\text{L2,alarms}}(t) = [R_{\text{alarms}}t + R_{\text{alarms}}]_+$ bytes, where t denotes the elapsed time in milliseconds. Since device alarms have highest priority, the queue is served by the 100 Mbit link with rate $\sigma = 0.125 \cdot 10^5$ bytes/ms and $\rho = 0$. It follows from (7) that the waiting time at link L2 is bounded by $W_{\text{L2,alarms}} \leq R_{\text{alarms}}/(0.125 \cdot 10^5)$ ms.

The aggregate arrival from all C factory units arrive to the high-priority queue to link L3. The arrivals from each factory unit are bounded by (9) as $D_{\text{L2,alarms}}(t) \leq [R_{\text{alarms}}t + R_{\text{alarms}}]_+$ bytes. It follows that the arrivals to link L3 are bounded by $A_{\text{L3,alarms}}(t) = CD_{\text{L2,alarms}}(t)$, which yields a waiting time of $W_{\text{L3,alarms}} \leq CR_{\text{alarms}}/(0.125 \cdot 10^5)$ ms. The worst-case end-to-end latency, L_{alarms}, is then given by the sum of the cycle time τ and the waiting times at L2 and L3:

$$L_{\text{alarms}} \leq \tau + \frac{R_{\text{alarms}}}{0.125 \cdot 10^5}(1 + C). \quad (10)$$

Similarly, the end-to-end reliability is the product of reliabilities at each link. The reliability of link L1 is given by the packet delivery probability of the network slicing scheme and the reliability of the Ethernet link, i.e. $\zeta_{\text{L1,alarms}}\zeta_{\text{link}}$, while links L2 and L3 each has reliability ζ_{link}. Hence, the end-to-end reliability is

$$\zeta_{\text{alarms}} = \zeta_{\text{L1,alarms}}(\zeta_{\text{link}})^3. \quad (11)$$

Following the same procedure for the patient info requests and allocating $R_{\text{req}} = 128$ bytes at L1 and taking the periodicity of 200 ms into account, the arrivals are bounded by $A_{\text{L2,req}}(t) = [R_{\text{req}}/200t + R_{\text{req}}]_+$. The leftover service remaining after the device alarms in the high-priority queue have been served is given by (8) as

$$S_{\text{L2,req}}(t) = [(0.125 \cdot 10^5 - R_{\text{alarms}})t - R_{\text{alarms}}]_+, \tag{12}$$

$$S_{\text{L3,req}}(t) = [(0.125 \cdot 10^5 - CR_{\text{alarms}})t - CR_{\text{alarms}}]_+. \tag{13}$$

Similarly, the patient info requests arriving to link L3 are bounded by (9):

$$A_{\text{L3,req}}(t) = [\alpha_{\text{L3,req}} t + \beta_{\text{L3,req}}]_+ \tag{14}$$

with

$$\alpha_{\text{L3,req}} = CR_{\text{req}}/200, \tag{15}$$

$$\beta_{\text{L3,req}} = CR_{\text{req}} + \frac{CR_{\text{req}}/200 R_{\text{alarms}}}{0.125 \cdot 10^5 - R_{\text{alarms}}}. \tag{16}$$

As a result, the worst-case end-to-end latency is

$$L_{\text{req}} \leq \tau + \frac{R_{\text{alarms}} + R_{\text{req}}}{0.125 \cdot 10^5 - R_{\text{alarms}}} + \frac{CR_{\text{alarms}} + \beta_{\text{L3,req}}}{0.125 \cdot 10^5 - CR_{\text{alarms}}}, \tag{17}$$

with reliability

$$\zeta_{\text{req}} = (\zeta_{\text{link}})^3. \tag{18}$$

B. Generalization

The analysis can be generalized to systems with the same hierarchical network structure by considering the factory unit and factory networks separately, and by using the fact that the factory unit network defines an upper bound on the arrivals to the factory network. Consider a factory unit application which transmits at most R bytes periodically every ηth cycle according to one of the proposed slicing schemes, and let τ denote the cycle time. An application transmission succeeds with a reliability ζ_{d} which is given by the slicing scheme, as described in Section II, and the reliability of the underlying technology. R and τ dictates the affine arrival bound which arrives to the factory network as

$$A(t) = \left[\frac{R}{\eta\tau} t + R\right]_+, \tag{19}$$

with reliability ζ_{d}. At the factory network, suppose the frame is routed through the links $P = \{p_1, p_2, \ldots, p_n\}$. Each link p_i has delay W_{p_i} and reliability ζ_{p_i} so that the worst-case end-to-end delay is

$$L = \tau + \sum_{p_i \in P} W_{p_i}, \tag{20}$$

with reliability

$$\zeta = \zeta_{\text{d}} \prod_{p_i \in P} \zeta_{p_i}. \tag{21}$$

The link reliabilities ζ_{p_i} are independent of the traffic and completely described by the physical link characteristics. However, the queuing times W_{p_i} depend on the traffic characteristics at each link, and must be calculated in the order in which the links are traversed by the traffic.

To obtain the queuing times, consider an arbitrary link in the path, which is equipped with a number of prioritization queues. Let $q^{(i)} = \{f_i^{(1)}, f_i^{(2)}, \ldots, f_i^{(N)}\}$ be the flows sharing queue i at the link, and let each flow $f_i^{(l)}$ be characterized by the arrival curve $A_i^{(l)}(t) = [\alpha_i^{(l)} t + \beta_i^{(l)}]_+$. Furthermore, define the strict queue prioritization order $\epsilon_1, \epsilon_2, \ldots, \epsilon_K$, so that queue j is served before k if $\epsilon_j < \epsilon_k$. The minimum service given to a flow $f_j^{(l)} \in q^{(j)}$ is obtained from (8) where $A_1(t)$ is the aggregate of all other flows with higher or equal priority as $f_j^{(l)}$. Using the notation introduced here, the minimum service given to $f_j^{(l)}$ is parameterized by

$$\sigma_j^{(l)} = \sigma + \alpha_j^{(l)} - \sum_{k \in [1,K]: \epsilon_k \leq \epsilon_j} \sum_{f_k^{(i)} \in q^{(k)}} \alpha_k^{(i)}, \tag{22}$$

$$\rho_j^{(l)} = \rho - \beta_j^{(l)} + \sum_{k \in [1,K]: \epsilon_k \leq \epsilon_j} \sum_{f_k^{(i)} \in q^{(k)}} \beta_k^{(i)}, \tag{23}$$

where σ and ρ describe the total service available to the queues at the node. The worst-case queuing delay and the departures are readily given by (7) and (9) as

$$W_j^{(l)} \leq \frac{\rho_j^{(l)} + \beta_j^{(l)}}{\sigma_j^{(l)}}, \tag{24}$$

$$D_j^{(l)}(t) \leq \left[\alpha_j^{(l)} t + \beta_j^{(l)} + \frac{\alpha_j^{(l)} \rho_j^{(l)}}{\sigma_j^{(l)}}\right]_+. \tag{25}$$

By using $D_j^{(l)}(t)$ as arrival bound at the next link, the procedure can be repeated until the waiting times at all links have been obtained.

The proposed framework can be automated and used even in large-scale networks with many link levels. However, it assumes that the traffic has been assigned queue priorities, which is itself a difficult problem especially for large networks with many traffic sources. The problem of assigning traffic to queues is outside the scope of this paper, but we remark that methods proposed in the literature for DNC (e.g. [21]–[23]) can be directly applied to the factory network using the framework presented here. However, extending the methods to include allocation of computational and storage resources represents a challenge since current techniques for this problem consider a very generic latency model, where links have constant delay but are capacity constrained, which is not directly applicable to real networks [12], [35].

V. Numerical Results

In this section, we investigate the end-to-end latencies and reliabilities derived in the previous section for the overwriting and prioritization queuing slicing schemes presented in Section II.

We first consider the resources for the device alarms within in each factory unit network. Since the number of alarms in each cycle is random, we can either reserve a fixed number of resources in each cycle, or we can allow alarms to overwrite the cyclic control traffic, which has a lower reliability requirement. Allocating a fixed number of resources results in a low utilization, while overwriting control traffic introduces a decrease in the reliability of the control traffic. Since the rate of alarms is very low compared to the cycle time, the overwriting scheme is a promising approach for this use case.

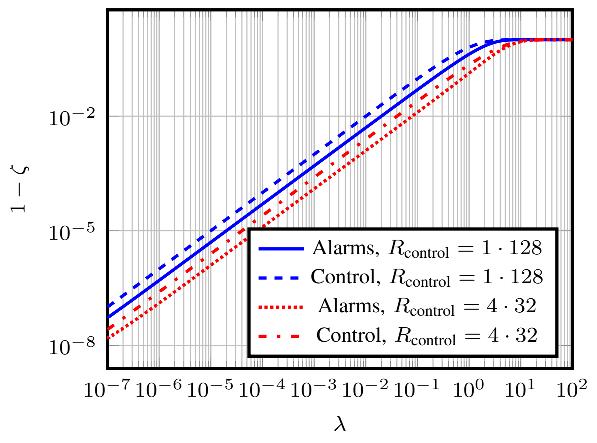


Fig. 5. End-to-end failure rate of control and device alarm traffic in the network slicing scheme based on overwriting for various alarm arrival rates.

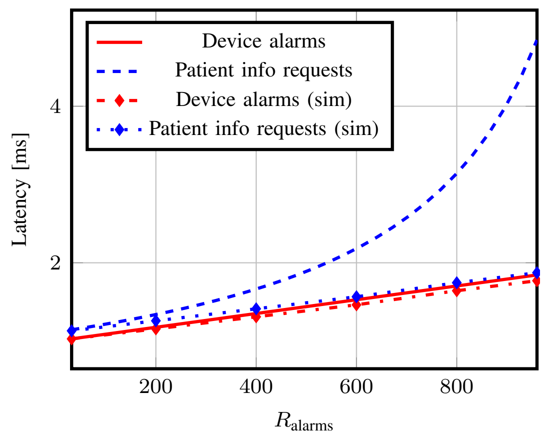


Fig. 6. End-to-end latency of device alarms and patient info requests for various alarm arrival rates.

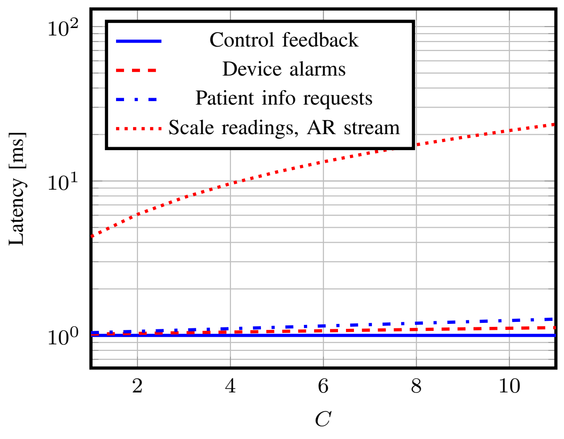


Fig. 7. End-to-end latency of the applications in the use case using the given prioritization scheme.

In the overwriting scheme, the reliability of the alarm traffic is given by (11) with $\zeta_{\text{L1,alarms}}$ obtained from (4). Similarly, the reliability of the control traffic, which only uses link L1, may be obtained as $\zeta = \zeta_{\text{L1,control}}\zeta_{\text{link}}$ with $\zeta_{\text{L1,control}}$ obtained from (3). The resulting reliability is illustrated in Fig. 5, where the end-to-end frame failure rates $(1-\zeta)$ for the alarm and control traffic are shown for different values of the mean alarm arrivals per cycle, λ. We consider the two cases where the control traffic is composed of either 1 telegram of 128 bytes, and of 4 telegrams of 32 bytes, denoted by $R_{\text{control}} = 1 \cdot 128$ and $R_{\text{control}} = 4 \cdot 32$. We assume that entire telegrams are overwritten in the overwriting scheme and that each telegram contains an integrity check so that it can be detected whether the original data has been overwritten. Therefore, only one alarm can be delivered per cycle when the control traffic is transmitted in a single telegram, and a single alarm results in complete failure of the control information. In the other case, if the control traffic is divided into 4 telegrams, then up to 4 alarms can be transmitted during a cycle, and one alarm transmission only causes failure of a single control telegram. This results in a lower failure rate as can be seen in the figure.

At the expected device alarm inter-arrival time of 60 s ($\lambda \approx 1.7 \cdot 10^{-4}$) from the use case, the reliability requirement of the control traffic ($\zeta = 1 - 10^{-6}$) is satisfied in the case where the control traffic is transmitted in 4 telegrams. Furthermore, the delivery reliability of the device alarm traffic at this point is also sufficient, and thus the overwriting slicing scheme would be a reasonable choice since it results in high utilization (100%). By comparison, if 32 bytes were allocated in each cycle only to the device alarms, it would on average only be used once every 60 seconds, yielding a utilization of approximately 0.02%, and would in addition occupy 32 bytes more of the frame than the overwriting scheme.

The telegrams that are destined to the cloud need to be forwarded by the gateways to the switched network, where the telegrams are served according to their priority. Obviously, applications that are given high priority influence the latency of the applications with lower priority. This is illustrated in Fig. 6 for the patient info requests for various values of R_{alarms}. The simulation results are obtained by simulating the scenario with periodic arrivals under the assumption that the factory units are synchronized, and observing the maximum experienced latency. Due to an increased serialization time, an increasing R_{alarms} causes an increase in the latency experienced by both the device alarms, as evident from (10), and the patient info requests, as described by (17). Since an increased number of arrivals also results in an increased burst size, the patient info request latency obtained using network calculus increases more than the device alarm latency. Furthermore, in a system with more queuing priorities, the impact of bursts would be amplified at each prioritization queue all the way to the queue with lowest priority. However, as the simulation results show, this tendency does not occur in practice. This is due to the affine approximation of the periodic arrivals used in the network calculus calculations. In particular, the affine approximation assumes that high priority bytes arrive during two cycles, which interrupts the low priority traffic and hence introduces an additional delay. In the case with periodic arrivals, the low priority traffic is served without interruptions from the point where the high priority traffic has been served until the next cycle.

We now consider the question of how the end-to-end latency is affected by the number of factory units in the network,

C. Fig. 7 shows the end-to-end latencies obtained using the network calculus methodology for all applications in the use case, where we assume that the control traffic is transmitted as $4 \cdot 32$ bytes. Notice that the total number of generated bytes exceeds the capacity of the links when C reaches 12, which is why only values of $C < 12$ are considered. The scale readings and the AR stream share the same latency as they share the same priority and follow the same path in the network. As can be seen, the latency requirements for control feedback, device alarms and the patient info requests are satisfied for all the considered values of C. However, the AR stream latency requirement fails when C reaches 10. Although it is expected that the high data rate applications with strict latency requirements will fail first, it also states the limit of deterministic network calculus, since the relatively low reliability requirement of the AR stream has not been exploited. In fact, one could be greedy in this situation and drop AR packets as long as the reliability requirement is satisfied in order to reduce the data rate.

VI. Conclusion

Industry 4.0 introduces a wide range of new requirements to industrial networks demanded by applications that interact with cloud services. This introduces several challenges in regard to the management and control of the network, and traditional technologies such as SDN are not well suited for the heterogeneous industrial networks. This paper investigates how network slicing can be used to deal with this complexity by introducing programmability and flexibility to industrial networks. We have presented methods for slicing cyclic and switched industrial communication protocols, and analyzed their trade-offs in utilization, reliability and isolation. Furthermore, we have introduced an Industry 4.0 use case that illustrates how network slicing can be used to handle the diverse requirements. Based on the use case, we have demonstrated how deterministic network calculus can be used to systematically analyze end-to-end latencies of network slices. The presented work can be used to apply existing techniques and algorithms such as queue allocation and admission control to the industrial domain.

References

[1] R. Drath and A. Horch, "Industrie 4.0: Hit or hype?" *IEEE industrial electronics magazine*, vol. 8, no. 2, pp. 56–58, 2014.

[2] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in *IEEE International Conference on Automation, Quality and Testing, Robotics*, 2014, pp. 1–4.

[3] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 17–27, 2017.

[4] J. Q. Li, F. R. Yu, G. Deng, C. Luo, Z. Ming, and Q. Yan, "Industrial internet: A survey on the enabling technologies, applications, and challenges," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1504–1526, 2017.

[5] T. Sauter, "The three generations of field-level networks—evolution and compatibility issues," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 11, pp. 3585–3595, 2010.

[6] 3GPP, "Service requirements for next generation new services and markets," 3rd Generation Partnership Project (3GPP), TS 22.261 v16.0.0, 06 2017.

[7] P. Schulz, M. Matthe, H. Klessig, M. Simsek, G. Fettweis, J. Ansari, S. A. Ashraf, B. Almeroth, J. Voigt, I. Riedel, A. Puschmann, A. Mitschele-Thiel, M. Müller, T. Elste, and M. Windisch, "Latency critical iot applications in 5g: Perspective on the design of radio interface and network architecture," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 70–78, 2017.

[8] E. Molina and E. Jacob, "Software-defined networking in cyber-physical systems: A survey," *Computers & Electrical Engineering*, 2017.

[9] NGMN Alliance. (2015, 2) 5G white paper. Accessed: 22/09/2017. [Online]. Available: http://www.ngmn.org/5g-white-paper.html

[10] S.-H. Leitner and W. Mahnke, "OPC UA–service-oriented architecture for industrial applications," *ABB Corporate Research Center*, 2006.

[11] H. P. Huth and A. M. Houyou, "Resource-aware virtualization for industrial networks: A novel architecture combining resource management, policy control and network virtualization for networks in automation or supervisory control and data acquisition networks," in *2013 International Conference on Data Communication Networking (DCNET)*, 7 2013, pp. 1–7.

[12] W. Mandarawi, A. Fischer, A. M. Houyou, H.-P. Huth, and H. de Meer, *Constraint-Based Virtualization of Industrial Networks*. Springer International Publishing, 2016, pp. 567–586.

[13] Y. W. Ma, Y. C. Chen, and J. L. Chen, "SDN-enabled network virtualization for industry 4.0 based on iots and cloud computing," in *2017 19th International Conference on Advanced Communication Technology (ICACT)*, 2 2017, pp. 199–202.

[14] Y. Yiakoumis, K.-K. Yap, S. Katti, G. Parulkar, and N. McKeown, "Slicing home networks," in *Proceedings of the 2nd ACM SIGCOMM workshop on Home networks*. ACM, 2011, pp. 1–6.

[15] N. Nikaein, E. Schiller, R. Favraud, K. Katsalis, D. Stavropoulos, I. Alyafawi, Z. Zhao, T. Braun, and T. Korakis, "Network store: Exploring slicing in future 5g networks," in *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture*. ACM, 2015, pp. 8–13.

[16] M. R. Sama, X. An, Q. Wei, and S. Beker, "Reshaping the mobile core network via function decomposition and network slicing for the 5g era," in *Wireless Communications and Networking Conference Workshops (WCNCW), 2016 IEEE*. IEEE, 2016, pp. 90–96.

[17] 3GPP, "Study on architecture for next generation system," 3rd Generation Partnership Project (3GPP), TR 23.799 v14.0.0, 12 2016.

[18] P. Gaj, J. Jasperneite, and M. Felser, "Computer communication within industrial distributed environment—a survey," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 182–189, 2013.

[19] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, and A. V. Vasilakos, "Software-defined industrial internet of things in the context of industry 4.0," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7373–7380, 10 2016.

[20] D. Li, M. T. Zhou, P. Zeng, M. Yang, Y. Zhang, and H. Yu, "Green and reliable software-defined industrial networks," *IEEE Communications Magazine*, vol. 54, no. 10, pp. 30–37, 10 2016.

[21] J. W. Guck and W. Kellerer, "Achieving end-to-end real-time quality of service with software defined networking," in *2014 IEEE 3rd International Conference on Cloud Networking (CloudNet)*, 10 2014, pp. 70–76.

[22] J. W. Guck, M. Reisslein, and W. Kellerer, "Function split between delay-constrained routing and resource allocation for centrally managed QoS in industrial networks," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 6, pp. 2050–2061, 12 2016.

[23] J. W. Guck, A. V. Bemten, and W. Kellerer, "DetServ: Network models for real-time QoS provisioning in SDN-based industrial environments," *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 1003–1017, 12 2017.

[24] D. Schulz, "Network models for the industrial intranet," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–10.

[25] K. C. Lee, S. Lee, and M. H. Lee, "Worst case communication delay of real-time industrial switched ethernet with multiple levels," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 5, pp. 1669–1676, 2006.

[26] K. C. Lee and S. Lee, "Performance evaluation of switched ethernet for real-time industrial communications," *Computer standards & interfaces*, vol. 24, no. 5, pp. 411–423, 2002.

[27] T. Sauter and M. Lobashov, "How to access factory floor information using internet technologies and gateways," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 699–712, 2011.

[28] L. L. Bello, E. Bini, and G. Patti, "Priority-driven swapping-based scheduling of aperiodic real-time messages over ethercat networks," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 741–751, 6 2015.

[29] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[30] L. Kleinrock, *Queueing systems, volume 2: Computer applications*. wiley New York, 1976, vol. 66.

[31] M. Fidler and A. Rizk, "A guide to the stochastic network calculus," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 92–105, 2015.

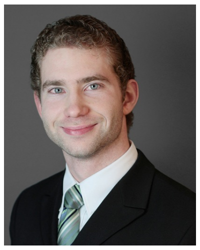
[32] J.-Y. Le Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the internet*. Springer Science & Business Media, 2001, vol. 2050.

[33] A. Van Bemten and W. Kellerer, "Network calculus: A comprehensive guide," Technische Universität München—Lehrstuhl für Kommunikationsnetze, Tech. Rep. 201603, 03 2016.

[34] IEEE 802.1 Time-Sensitive Networking Task Group. Accessed: 19/05/2017. [Online]. Available: http://www.ieee802.org/1/pages/tsn.html

[35] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, and G. S. Paschos, "The algorithmic aspects of network slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, 8 2017.

Anders E. Kalør Anders E. Kalør (S'17) received the B.Sc. degree in computer engineering and the M.Sc. degree in networks and distributed systems from Aalborg University, Denmark, in 2015 and 2017, respectively. He is currently pursuing a Ph.D. degree in the area of wireless communications and networking at Aalborg University. His research interests include MAC layer design and optimization for wireless systems, and networking.



René Guillaume Rene Guillaume received his B.S. and M.S. degrees in Electrical Engineering from the University of Duisburg-Essen, Germany, in 2010 and 2012, respectively. After being a visiting student research collaborator at Princeton University in 2012, now he is pursuing the PhD degree in Electrical Engineering. Since 2015, Rene is a research engineer within the Corporate Research Department of Robert Bosch GmbH, Germany, and joined the team on communication and network technology. His research interests include security and privacy in wired and wireless networks, Software-defined Networking and Time-sensitive Networking.

Jimmy J. Nielsen Jimmy Jessen Nielsen (S'06-M'08) is associate professor at the Department of Electronic Systems, Aalborg University, Denmark. He obtained his Ph.D. in Wireless Communications from Aalborg University in 2011. His research interests are on traffic models for Internet of Things and M2M systems, reliable and low latency communications, wireless networks, and performance analysis. He has authored and co-authored more than 40 publications in conferences, journals, and books. He has served as reviewer for various IEEE journals and conferences.

Andreas Mueller Dr. Andreas Mueller is leading the group focusing on communication and network technology within the Corporate Research Department of Robert Bosch GmbH in Stuttgart, Germany. As part of this role, he is coordinating Bosch's research activities in the area of future industrial connectivity infrastructures as well as the topic "5G for Industry 4.0" within the Bosch group. In addition to that, he is the Chairman of the Board of the "5G Alliance for Connected Industries and Automation" (5G-ACIA), which recently has been established in order to make sure that 5G for the industrial domain becomes a success. Prior to joining Bosch, Andreas was a Research Staff Member at the Institute of Telecommunications of the University of Stuttgart, Germany, where he was contributing to the further development of the 3GPP Long Term Evolution towards LTE-Advanced. Besides, he was working as a Systems Engineer for Rohde & Schwarz, developing a novel software-defined radio based communication system for the German Armed Forces. Andreas holds a German Diploma degree as well as a Ph.D. degree in Electrical Engineering (with distinction) and a M.Sc. degree in Information Technology, all from the University of Stuttgart, Germany.

Petar Popovski Petar Popovski (S'97-A'98-M'04-SM'10-F'16) is a Professor of Wireless Communications with Aalborg University. He received the Dipl. Ing. degree in electrical engineering and the Magister Ing. degree in communication engineering from the "Sts. Cyril and Methodius" University, Skopje, Republic of Macedonia, in 1997 and 2000, respectively, and the Ph.D. degree from Aalborg University, Denmark, in 2004. He has over 300 publications in journals, conference proceedings, and edited books. He holds over 30 patents and patent applications. He is currently a Steering Committee Member of IEEE SmartGridComm and previously served as a Steering Committee Member of the IEEE INTERNET OF THINGS JOURNAL. He is a holder of a Consolidator Grant from the European Research Council (ERC), recipient of the Danish Elite Researcher Award and a member of the Danish Academy for technical sciences (ATV). He is currently an Area Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. His research interests are in the area of wireless communication and networking, and communication/information theory.