Software-Defined Vehicles (SDV) with Time-Sensitive Networking (TSN) and Software-Defined Networking (SDN)

Empowering Tomorrow's Automotive Software

www.etas.com

ETAS

Executive summary

The whitepaper outlines the usage of TSN (Time Sensitive Networking) for in-vehicle Ethernet networks, describes an in-vehicle QoS (Quality of Service) framework for TSN, and indicates TSN-related use cases for Software-Defined Vehicles (SDV) in context of development and operational (DevOps) lifecycle phases. Such a powerful in-vehicle communication system with the ability in support of dynamic changes requires dedicated control, operation, and management support, not only in the offboard area of a vehicle, particularly also vehicle onboard. The Software-Defined Networking (SDN) pattern could satisfy such architectural requirements. A correspondent Automotive SDN model is derived, which should serve the entire spectrum of SDN-based SDV system architecture variants.

The whitepaper is complemented by a glossary, a detailed bibliography, as well as a proposal how existing Hard-Wired Vehicles (HWV) may be transitioned for TSN and QoS support, and incrementally evolved towards SDN-based SDVs.

Targeted audience: the whitepaper provides some overview and introductory information for beginners in the subject matter but addresses primarily experts in systems engineering of in-vehicle E/E architectures and its vehicular communication systems, familiar also to a certain extend with the relevant underlying standardization framework.

The whitepaper represents a common activity by ETAS, TTTech and Bosch, and provides a shared view on the subject matter.

Table of contents

1. Introduction

In recent years, the amount of vehicle compute nodes (like electronic control units (ECU)), software code and therefore also communication services and interconnections typically found in vehicles has drastically increased. Furthermore, many different communication technologies in terms of physical protocol layer media and protocols are deployed simultaneously. Therefore, up to date, the development and deployment of network configurations for vehicles is a complex expert process, resulting in statically defined, operated and difficult to monitor networks.

Software-defined networking (SDN) is an approach to network design, control, operation, and management which abstracts the underlying network hardware infrastructure and the capabilities of individual network devices, even entire networks, and manages the network infrastructure and associated network services through software-based controllers (see also Fig. 8).

In traditional, unmanaged networking, network devices such as switches and routers make decisions about how to forward data packets based on predefined rules and configurations. In managed networks, these rules and configurations are handled during operation by network element managers (on a device level) or network managers (on a network level). These management services are located in the management plane. In contrast, SDN further decouples the control plane, responsible for determining how traffic is routed, from the data (also known as user) plane, which handles the actual forwarding of packets. This separation enables centralized management and programmability of the network.

SDN architectures typically contain three key components:

The **SDN controller,** the centralized brain of communication service control and associated network management

The **southbound network interfaces (SBI),** which enable communication between the SDN controller and network devices

The **northbound network interfaces (NBI),** which expose the SDN controller's functionality to higher-level network applications and services

In industrial settings, such as manufacturing plants, utilities, and transportation systems, SDN principles and technologies help to manage and optimize control systems in complex distributed architectures. Industrial networks often consist of a multitude of different devices that monitor and control physical processes, such as sensors, actuators, and control units. SDN enables centralized control and orchestration of network resources, which in turn enables dynamical provision, configuration, and operation of network infrastructure to support critical industrial processes and applications. By abstracting network functionality and providing a unified management interface, SDN simplifies network operations and enables greater flexibility, resiliency, and scalability in industrial environments.

The conception of a Software-Defined Vehicle (SDV) shall indicate a paradigm change to the past, which may be contrasted as Hard-Wired Vehicles (HWV). Scope is the vehicle-embedded distributed computing and communication system, the vehicle information system. The predicate "software-defined" is adopted from Software-Defined Networking (SDN), an established architectural pattern in ICT (Information & Communication Technologies). In brief, "software-defined" shall emphasize to (engineering) goals of openness and programmability of networks in particular, or distributed information systems in general. The SDN was developed by ICT Standards Development Organizations (SDO): origin, history and crucial SDN specifications may be found e.g., in [1].

The SDV conception adopts first and foremost openness and programmability [2, 3]:

Programmability relates primarily to a programming interface with the capability in (re-)pro-gramming in-vehicle resources, from perspective of control, operations, and management of the in-vehicle information system.[1]

Openness shall express the goal in following standards (in contrast to automotive proprietary approaches) with the option of open-source solutions for such standards.

1 The whitepaper focuses on in-vehicle network resources as well as communication services in relation to TSN, omitting the still existing other legacy in-vehicle communication technologies.

Both "software-defined" characteristics may be further concretized:

The ability of programmability shall be supported along the entire lifecycles of SDVs under consecutive **development and operational (DevOps)** phases, thus e.g., all existing configuration management activities might be considered from an (more or less open) programming interface.

The goal of openness shall at least address existing and reusable **ICT technologies** (like languages, protocols, architecture patterns) which are subject of use and integration in vehicles (here: IEEE-defined Ethernet, IETF-defined Internet protocol suites, ITU-T and ONF architectural frameworks)

The system context of an SDV in-vehicle communication system relates to a small area network. Such a network topological size allows to use the data link protocol layer as integration point for all type of communication services (rather than e.g., the network layer as in large area networks). The top in-vehicle network hierarchical level is expected to use Ethernet in SDVs (in-vehicle core network), with the option in penetrating and replacing existing non-Ethernet automotive communication technologies in future (if beneficial at all).

Such an all (communication) services-integrated Ethernet implies the support of services qualities (like Quality of communication Service (QoS), communication security, service availability, etc., - inter alia in context of functional safety) by the Ethernet data link protocol layer, enforcing the usage of communication service building blocks as defined within IEEE 802 TSN (Time-Sensitive Networking) standardization community.

In conclusion, this document focuses on:

The major challenge of planning, development, validation, verification, control, operation, and management of SDV-embedded TSN-based in-vehicle networks along DevOps lifecycle phases

The promise of a SDN-based SDV architecture as established, well-known solution pattern in ICT

1.1 Distributed in-vehicle computing systems (E/E architecture)

So-called E/E architectures [4] for SDVs represent essentially the "SDN object" of control, operations, and management.

The term E/E architecture stands for "electrical and electronic architecture" and means a coupled, two-plane architecture, consisting of: 1) an electrical energy or power distribution network plane; and 2) an information processing and communication network architectural plane. A third tag (E3) is sometimes added to E/E to indicate the vehicular propulsion technology: the third E then indicates an electric vehicle.

Our scope here is solely the "2nd E", i.e., the technical in-vehicle distributed information, computing, and communication system. The so-called "in-vehicle" context (like in [5]) with scope on a single vehicle and its onboard system only, may be enlarged to vehicular system contexts like the Intelligent Transportation System (ITS) architecture (e.g., [6]), or extended to a cluster of V2V-interconnected vehicles.[2]

The SDN pattern is basically applicable for all automotive system contexts (see also [7], [8], [9]).

This distributed information system follows DevOps lifecycle phases when applied to SDVs. Extended lifecycle models acknowledge that SDV-under-Ops may also be subjects of parallel development activities (i.e., SDV-under-Ops-under-Devs besides major development activities in SDV-under-Dev).

Vehicle application functions are typically partitioned in application category specific in-vehicle application domains (like control, body, infotainment, …). These vehicle applications (Apps) are mapped and deployed on E/E architectures. There are many variants and evolution scenarios of E/E architectures (see Fig. 1 and 2), differing in topological structure, resource staffing, constraint-levels, communication connectivity, and similar properties.

2 V2V means direct Vehicle-to-Vehicle communication services and belongs to the overall category of V2X (Vehicle-to-Everything) communication interfaces. Vehicles operated in the public space of a society would require agreements with operators of PLMNs (Public Land Mobile Network) for QoS support across V2X communication interfaces. TSN will be vehicle externally not used for many V2X interfaces. QoS is here typically linked to Internet Protocol based services. An exception is the TSN service model in 5th Generation Mobile Networks (see 3GPP TS 23.501, System Architecture for the 5G System) which may offer e.g., a TSN bridging service for a group of vehicles.

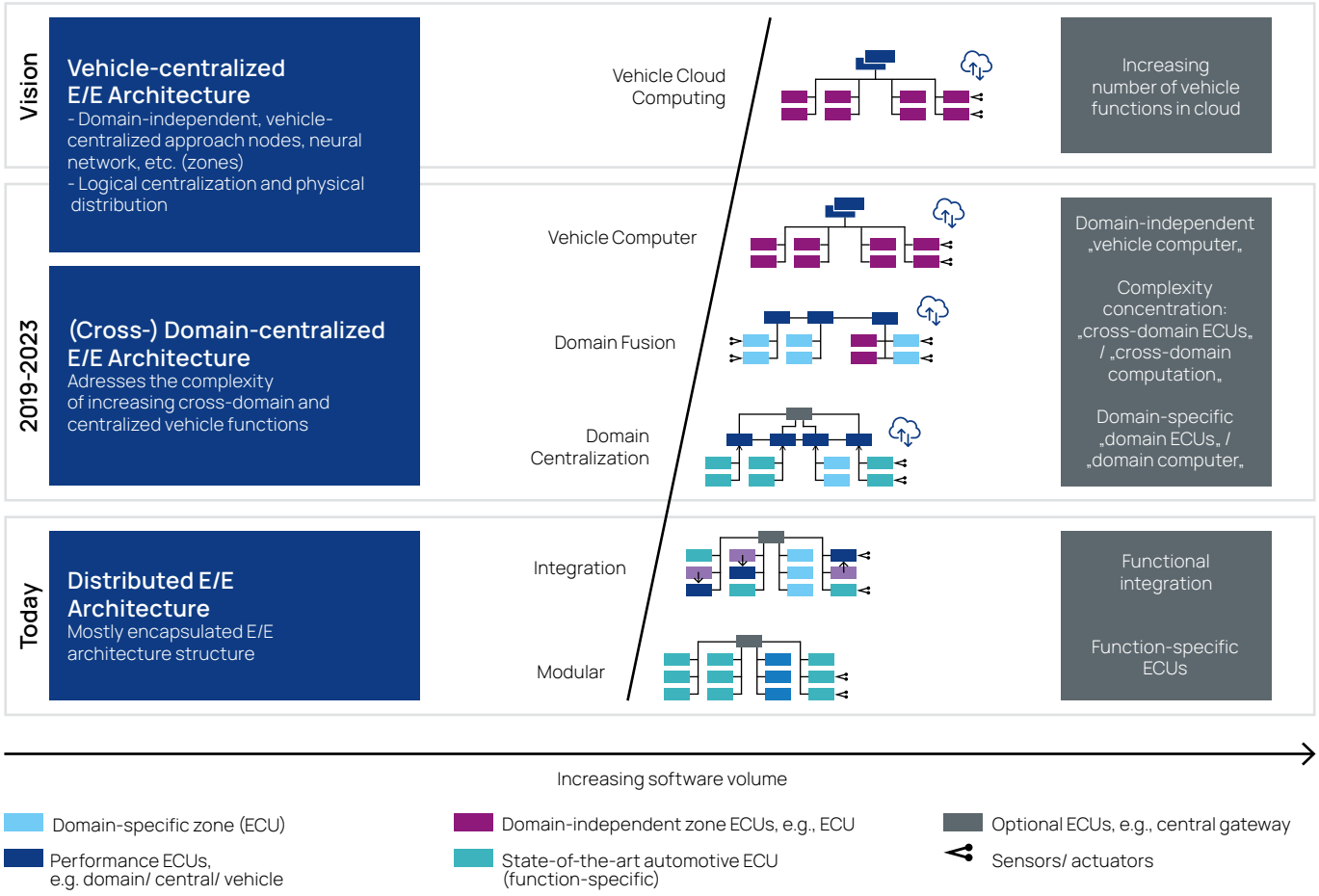


Figure 1: Vehicle computers and cloud connectivity will fundamentally change automotive E/E architectures (copy from [10], which reflects a roadmap vision from ca. 2017))

The distributed E/E system of SDVs shall support stringent system qualities, such as

- Qualities of system safety
- Qualities of system services
- Qualities of system securities
- etc.

which requires or implies

- Quality of communication service
- Quality of communication security
- Quality of communication reliability and availability
- etc.

at the level and context of the in-vehicle communication system.

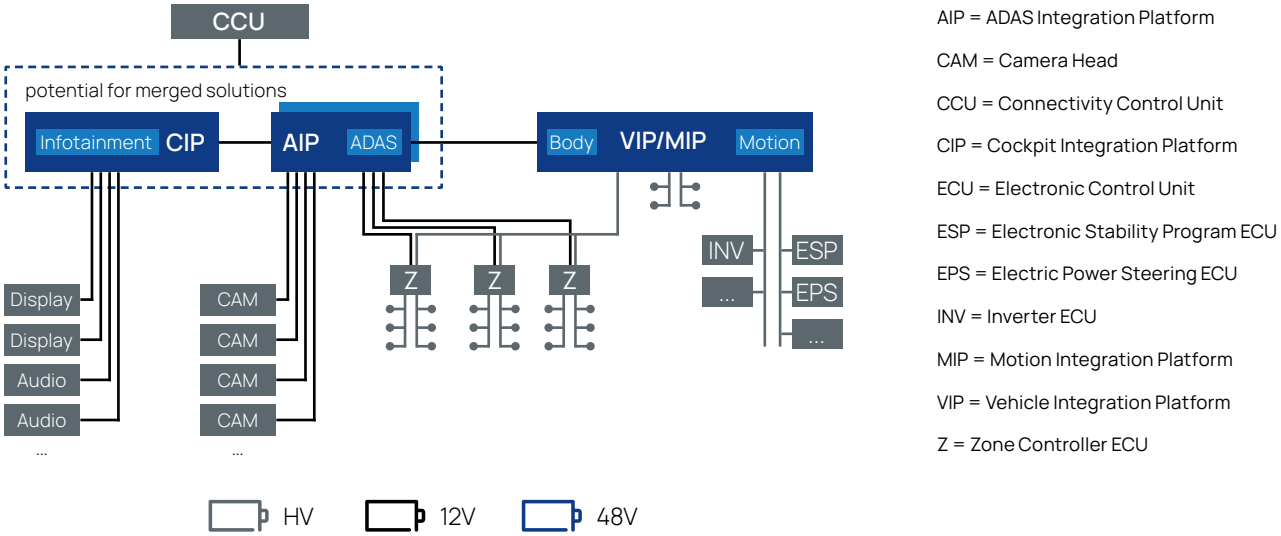


Figure 2: Exemplification of an automotive E/E architecture with a mixture of zonal, cross-domain and centralized vehicle compute nodes to serve varying levels of integration and centralization (copy from [11])

It's apparent that all such SDV system qualities (as non-functional requirements, NFRs) shall be independent of a particular E/E architecture. SDV system qualities are orthogonal to any selected technical deployment structure of the distributed in-vehicle computing system.

The distinction of an E/E architecture in multiple information planes is mandatory from in-vehicle communication services engineering and network engineering perspective.

An SDN-based SDV is characterized by three information planes: The **user plane** for native data transfer services for distributed user applications, a network **control plane,** and a **management plane.**

It may be stated that any functional SDN-based SDV control and management plane architecture should be independent in multiple aspects, primarily:

Vehicle computing architecture agnostic (i.e., independent of the specific E/E architecture)	Vehicle operating system agnostic (like OEM-specific or preferred Automotive Operating Systems)	Vehicle software architecture framework agnostic (like AUTOSAR, JASPAR, Eclipse SDV, other)

In-vehicle operating systems are at two system abstraction levels: distributed OS across the entire computing system (sometimes integral part of middleware (MW)) and at the OS level of individual compute nodes (like QNX, RTA OSEK, Linux, etc). Functional SDN-based SDV architectures must clarify the specific roles of the in-vehicle operating system(s), controlling system and managing system due to their tight inter-operation and cooperation.

1.2 Applicable networking standards

Any QoS-attributed in-vehicle Ethernet network requires a fully engineered QoS architecture across all three in-vehicle network planes (user, control, and management). IEEE 802 as technology owner of Ethernet does not define comprehensive Ethernet QoS architectures, but rather provides indication on essential QoS building blocks as well as their usage (see e.g., informative Annexes I, L, N, P, Q, S or T in IEEE 802.1Q [12]). IEEE 802 derives QoS building blocks from general (ITU-T), IP-based (IETF) and Ethernet-based QoS architectures (MEF). E.g., see IEEE 802.1Q bibliography entries in Annex W: [B21] to [B34] (bibliography entries in IEEE standard, not in whitepaper here).

The standards foundation for all Ethernet QoS architectures (independent of a particular industry domain) will therefore be related to those standards which the IEEE references for QoS architecture and technology. Some of the most relevant standards are:

TSN QoS concepts like **traffic specification (Tspec), priority specification (Pspec), quality specification (Qspec)** are derived from the ITU-T Y.1222 traffic contract (TC) concept and the MEF 10.3 concepts of Ethernet service attributes and bandwidth profiles, and adapted to IEEE TSN.

For TSN, the most relevant standard is IEEE P802.1DG, the Automotive TSN Profile. The relationship of this coming standard to software-defined networking in an SDV is discussed in Section 3.

ITU-T Y.1291 [13]
as generic functional QoS architecture for all kind of packet-switched communication technologies (i.e., includes also Ethernet)

ITU-T Y.1222 [14]
for Ethernet QoS functions concerning Ethernet traffic control and congestion control (i.e., the TSN specific QoS functions are related to that basic Ethernet QoS functions)

ITU-T Y.1563 [15]
as basic framework for Ethernet traffic performance parameters (i.e., quality parameters which might be used in quality specifications (Qspec) and service level agreements (SLAg)

MEF 10.3 [16]
as an Ethernet QoS architecture supporting the most stringent QoS objectives (called carrier-gradeness) for Ethernet networks at the scale of Metropolitan Area Networks.

2. Market Drivers for Software-Defined Vehicle (SDV)

2.1 SDN-based SDV and its information planes

As said, any E/E architecture of an SDV may be divided into multiple information planes. In-vehicle applications may exchange (application) data, known as user plane (UP) information. The control of in-vehicle distributed applications, such as initialization or shut-down of functions, is performed on the control plane (CP). And all the information required for operations and management of the system is typically handled on the management plane (MP). In this document we incorporate control and management information into the "merged" MP for simplicity, so that we only need to distinguish between the UP and MP.[3]

The SDN architectural pattern belongs to both the CP and MP, id est a dedicated control and management subsystem within the E/E information system (see Fig. 7).

A key architectural feature is the usage of a fully centralized (SDN) controller/manager function [17], accountable and responsible typically for all network infrastructure resources and end-to-end communication services. A thorough examination and treatment of the information planes is essential for the success of the SDV

2.2 SDN-based SDV versus SDV E/E architectures

E/E architectures may be classified e.g., according to topological, hierarchical, functional concentration and in-vehicle physical deployment of computing elements. Dedicated vehicle application functions may be deployed centralized or distributed with deployment varying across the entire spectrum. Prominent E/E architecture patterns are known as e.g., domain, zone, central (see [10, 11]).

The vision of an SDN-based SDV shall be as independent as possible of specific E/E patterns. Such an orthogonality should facilitate future safe evolution concerning the transition of E/E architectures (see also the outlined incremental evolution scenario of automotive SDN in Appendix III). The careful deployment of the fully centralized (but probably nomadic) SDN controller/manager function to a specific E/E compute node will be a key architectural decision.

3 For experts: the merger of management and control planes is called Management-Control Continuum (MCC) (see e.g., ITU-T G.7701, G.7702, ONF TR-521). The proposed SDN Southbound Interface (SBI) represents essentially a MCC-type of interface, as a result of a fully centralized SDN approach.

3. In-Vehicle Networking Requirements

In this chapter we describe requirements that any method for developing and maintaining in-vehicle networks must provide or support. This includes organizational aspects such as "support for the entire development and operation (DevOps) lifecycle of an SDV", as well as technical elements like "support for all traffic types encountered in in-vehicle networks". Any approach to SDV developments needs to demonstrate that it can meet these requirements.

3.1 DevOps Lifecycle support

Software-Defined Vehicles (SDV) belong to the category of mobility systems under continuous development and operations (Mobility DevOps). Subsequent basic, static SDV-under-DevOps lifecycle model illustrates major lifecycle phases (Fig. 3):

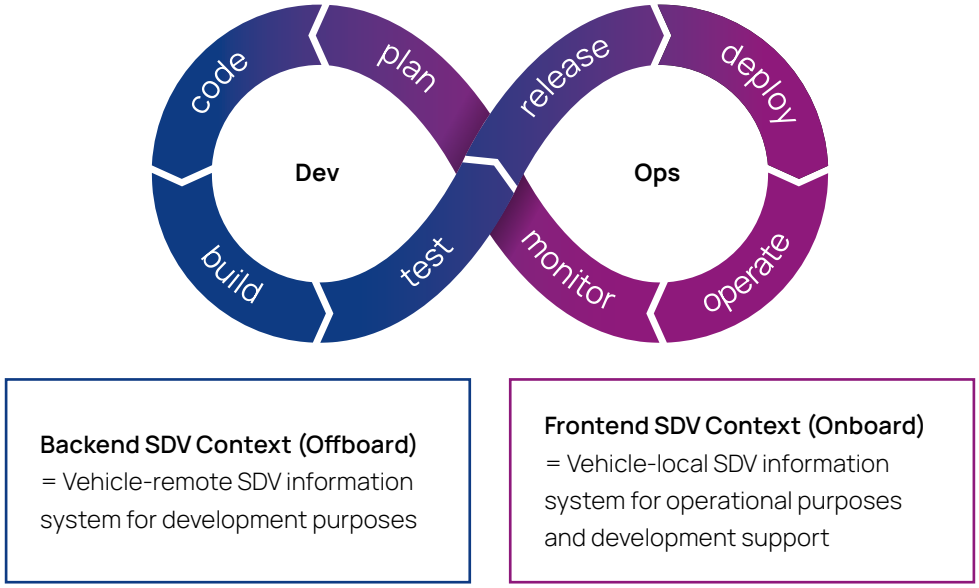
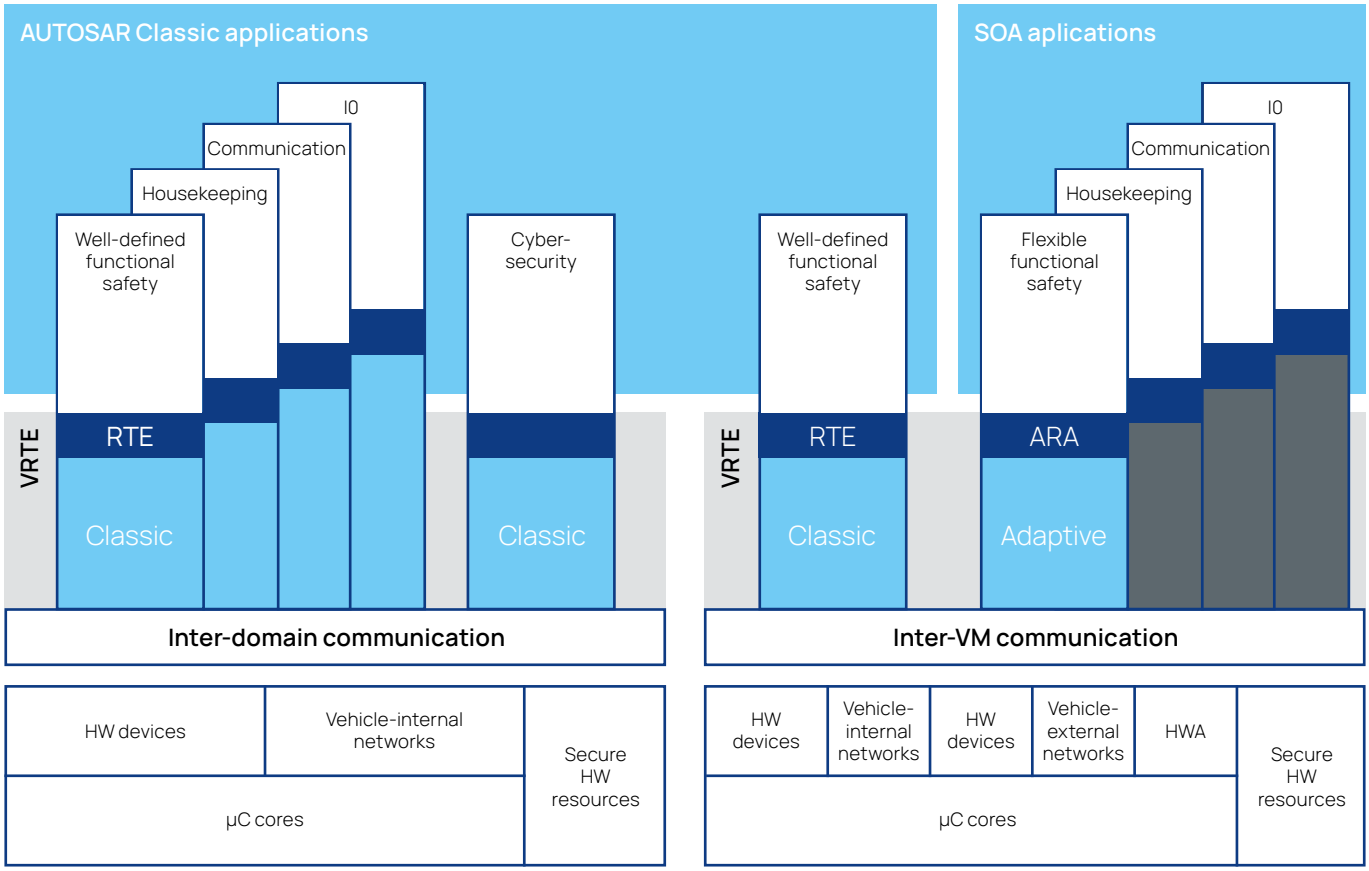


Figure 3: Lifecycle model of consecutive or/and parallel phases of development (Dev) and operations (Ops)

Each SDV lifecycle phase might be further detailed. For instance, the build phase could be divided in the initial vehicle manufacturing phase and later, more software-oriented rebuild phases. Or the operate phase should be subdivided in several vehicle modes of operation according to vehicle mobility state models, etc.

3.2 Support for Classic and Adaptive AUTOSAR

To support and integrate into well-established automotive ecosystems like AUTOSAR (see Figure 4) is another decisive requirement for the SDN-based SDV. A commercial success is only possible, if compatibility with currently used and proven technology is ensured. Appendix I briefly addresses this challenge for AUTOSAR and its Asian pendant JASPAR.



SOA = service oriented architecture HWA = hardware abstraction VA = virtual machine

Figure 4: Basic structure of vehicle compute software with AUTOSAR Classic and Adaptive components (copy from [10])

3.3 Eclipse SDV and SDN support?

In contrast to SDN, the open technology platform for SDV software "Eclipse SDV" (https://sdv.eclipse.org/) does not explicitly define a control or management plane in its framework and ecosystem. It may be expected that dedicated automotive SDN support would affect primarily the Eclipse SDV functions around their "orchestration & management" layer.

3.4 In-vehicle communication traffic types

Traffic types classify the traffic present in a network depending on their respective QoS requirements (cf. [12], Annex I). Historically, they were meant to establish a priority and drop precedence; however, in networks hosting critical functionality, this distinction is considered insufficient due to extended QoS requirements.

To distinguish between the different traffic types, at least several priorities are required as supported by priority tagged frames in Ethernet. However, for the operation of streams and more advanced planning capabilities, the use of different VLANs is also necessary and supported by Ethernet.

The following table provides an overview over different traffic types with their description and possible mapping to QoS requirements and TSN capabilities expected in in-vehicle networks.

Traffic Type	Description	Possible mapping to network QoS/TSN[4] capabilities
Critical Application	Very critical applications transmitting cyclic or sporadic packets over the network, where data delivery of each packet is guaranteed to occur at all registered receivers at or before a predictable time with no tolerance to lose frames, e.g. emergency call (eCall), emergency braking	High-priority traffic with guaranteed bandwidth (TAS) and stream redundancy (FRER)
Cyclic	Cyclic data where delivery of each packet is guaranteed to occur at all registered receivers within a predictable timespan starting when the packet is transmitted by the sender and ending when the packet is received with a very limited tolerance to lose data, e.g. Sensor (Radar, Lidar, MEMS (micro-electromechanical systems))	Network-synchronous talkers and listeners and guaranteed scheduled bandwidth (TAS), possibly FRER and PSFP
Events	Sporadic data where delivery of each packet is guaranteed to occur at all registered receivers within a predictable timespan starting when the packet is transmitted by the sender and ending when the packet is received with a tolerance to lose data, e.g. V2X, events/warnings/alarms	Asynchronous multicast talkers, medium-high-priority streams with limited transmission rate (CBS)
Network Control	Traffic for network and network node related infrastructure services, e.g., provision of clock synchronization (IEEE 802.1AS).[5,6,7]	Medium priority traffic class with guaranteed bandwidth
Configuration & Diagnostics	Management plane traffic with scope on configuration deployment and ad-hoc or continuous collection of diagnostics data (Automotive Diagnostics on Ethernet, Network health), e.g. management protocols , legacy diagnostic protocols like DoIP, ...	Medium priority unicast traffic with rate-limited bandwidth (CBS, ATS)
Best Effort	The native forwarding network service without any explicit QoS support. The typical default category of loss-tolerant, delay-tolerant and bitrate-tolerant traffic types.	Low-priority traffic
Audio/Video (Infotainment)	Audio/Video data for in-vehicle infotainment where delivery of each packet in a stream is guaranteed to occur at all registered receivers within a predictable timespan starting when the packet is transmitted by the sender and ending when the packet is received with a tolerance to lose data, e.g. phone calls, Road Noise Cancellation, Audio/Video Streams from a Head Unit to Speaker, Amplifier, Digital Signal Processor (DSP).	Medium priority multicast traffic with rate-limited bandwidth (CBS)
NOTE 1 – Automotive in-vehicle traffic type profiling is also conducted by standards development organizations. E.g., the in-vehicle traffic type categorization by JASPAR [38], or the preliminary automotive network traffic classification by IEEE P802.1DG in draft version D1.4.		

Table 1 - Typical in-vehicle communication traffic types

Table 1 provides an overview of typical in-vehicle communication traffic types.

4 The acronyms used are: Time-Aware Shaper (TAS), Frame Replication and Elimination (FRER), Credit-Based Shaper (CBS), Asynchronous Traffic Shaper (ATS), Per-Stream Filtering and Policing (PSFP)

5 The IEEE 802.1Q "network control" concept provides a rough indication, however without any detailing of management services in various traffic types. For instance, there are different QoS objectives (and traffic subtypes between management services like performance monitoring, fault management, alarm reporting or system reachability (by operator and manager) under severe overload conditions or security threats.

6 There are many network operational protocols running in the management plane with respect to basic Ethernet and Internet network services such as SPT, ARP, DHCP, DNS, ICMP, IGMP, LLDP, routing protocols at layer 2 and 3. All that (MP)-protocols consume layer 2 bitrate, to be considered by network resource management.

7 In-vehicle Ethernet should be basically "carrier-grade", which would imply support of IEEE 802.1 Ethernet OAM (= link-associated Operation, Administration, and Maintenance related traffic flows) and CFM (Connectivity Fault Management) [12, clauses 18, 19, 20, Annex J]. The CFM services are integrated again in e.g., the in-vehicle discovery, operational state supervision, fault management, or online testing services.

3.5 Network plane specific traffic considerations

Primarily user plane related communication services are subject to definition via the Northbound interface, including latency and reliability requirements, due to their immediate relation to in-vehicle user applications. The other traffic types are considered part of the base network infrastructure; they need to be set up and working before control and management plane entities use their services to establish streams for critical traffic.

3.6 Priorities and traffic classes

The primary mechanism in Customer Virtual Local Area Network (C-VLAN) bridges[8] to segregate different types of traffic from each other are the transmit queues, or traffic classes. These essentially enable spatial separation and isolation of different traffic types upon transmission. It also enables an IEEE 802.1Q CNC entity (represented here by the combined Control/Management Plane entity Centralized Network Controller & Manager) to apply different planning strategies, depending on desired stream- or traffic type properties as well as device capabilities. It shall be noted that essentially a wide variety of device capabilities is possible due to the modular nature of the so-called "TSN functions". For example, a certain bridge port may or may not expose Enhancements for Scheduled traffic (SCHED, formerly known as 802.1Qbv), or Asynchronous Traffic Shaper (ATS, formerly known as 802.1Qcr); the same applies to the number of available traffic classes per port.The assignment of a certain packet[9] to a traffic class is controlled by the Traffic Class Table ([12], cl. 8.6.6), which contains one target traffic class per priority.

3.7 Traffic types mapping

In TSN, any set of designed traffic types has to be mapped to at most eight traffic classes. As the number of transmit queues can vary between devices and ports and due to the nature of the indirect control of the transmit queues via the traffic class table, we define a mapping between traffic types and ports, as IEEE 802.1Q-LANs are specified with 8 priorities (0..7).available. For now, and for the sake of simplicity we assume the default mapping between priority and traffic classes ([12], Table 8-5). The default mapping of priorities to traffic classes considers every possible number of traffic classes, theoretically ranging from 1 to 8. Where the mapping shows (almost) an identity mapping in case that 8 traffic classes are supported, lower numbers of traffic classes naturally requires merging of different priorities in the same traffic class, thus introducing restrictions to the control/management plane regarding logical separation of traffic types

3.8 Traffic Engineering (TE) of in-vehicle communication traffic

The introduced traffic types as abstraction of in-vehicle communication service types represent traffic source models, defined by the associated traffic specification (Tspec) as a communication traffic descriptor. Such models are often initially of hypothetical nature, still inaccurate due to uncertain assumptions. A pretty normal situation in network engineering. The better the knowledge about traffic types, the more accurate the traffic source models, the more effective would be admission control, and the more efficient the network resource management.
Given that, the SDN-based SDV could and should apply a continuous traffic engineering process as outlined by e.g., [18 (Fig. 1), 19 (Fig. 5-2)], a basic proceeding in managing high quality networks. A typical requirement for SDN-based SDVs would be a probe-based traffic monitoring of traffic sources,

Traffic Type	Priority	Traffic Class (8 Queues)
Critical and Cyclic Applications	7	7
Alarms	6	6
Network control	5	5
Audio/Video Streams	4	4
Configuration & Diagnostics	3	3
Best-Effort	0-2	0-2

by the endstation itself or the first bridge. Traffic sources are modelled by YANG (Yet Another Next Generation) data models (see Section 5.1). The acquired data will be stored in the agent-local management datastores and could be later postprocessed by the vehicle local or remote manager. A typical knowledge ramp-up and fine-tuning process conducted by the SDV management plane.

8 For experts: the in-vehicle Ethernet network is basically engineered in a "flat manner" due to the assumption of a single authority, incorporating all roles like network service provider, network infrastructure provider, operator and manager. Consequently, such an IVN does not support hierarchical partitioning like provider bridge network (PBN) operation (see [12], clause 16]), which would imply different VLAN type roles like e.g., Service VLAN (S-VLAN). However, SDV IVNs-under-Ops are also subjects of continuous development, which may be addressed by interconnecting an onboard, Ethernet-based development system, which is required to be logically separated from the E/E system. Such onboard constellations may lead to advance in-vehicle network engineering, based on PBN principles, double-tagged VLANs, etc.

9 Protocol Data Units (PDU) such as a Layer 2 Ethernet MAC frame, Layer 3 IP datagram, Layer 4 packet.

3.9 IEEE P802.1DG - the automotive TSN Profile

IEEE P802.1DG (Time-Sensitive Networking (TSN) Profile for Automotive, at the time of writing not yet a finalized IEEE standard and therefore prefixed with "P") defines a set of capabilities for TSN-based – i.e. with aspects of real-time, determinism, and network convergence - automotive networks [20]. Therefore, this standard is an obvious point of reference for in-vehicle networking. However, when integrating with a software-defined networking (SDN) architecture, several gaps and challenges can be identified:

Control and management plane integration:

IEEE P802.1DG goes into quite some detail on the data (or user) plane mechanisms for TSN networking but lacks substantial references or detailed specifications for control and management plane integration with an SDN controller and in-vehicle network manager. This is not surprising because P802.1DG does not target an SDN architecture. SDN controllers require comprehensive control over network elements to enforce policies and configurations. There are two aspects that both need to be addressed so that an SDN controller can efficiently manage the network control plane: traffic modeling (how traffic parameters are represented) and protocol integration (the way how these parameters are stored and distributed).

Only by defining both aspects in sufficient detail and without ambiguity can SDN controllers and in-vehicle network managers manage TSN features in a standardized way. In this document we discuss both aspects in the sections about Northbound and Southbound interfaces of the SDN controller, respectively (see also Fig. 8).

Even though such definitions are in scope of any comprehensive network profile specification, the initial release of P802.1DG excludes the in-vehicle (TSN) network control plane and network management plane entirely, as well as the management layer of network management as such. P802.1DG scopes individual network elements only, but not the network in its entirety.

Policy monitoring:

Due to the lack of any notion of an SDN controller, IEEE P802.1DG also lacks any definition for the enforcement of network policies[10] defined by an SDN controller. Policy monitoring is critical for ensuring that network behavior aligns with the desired outcomes defined by network configuration management entity (like the IEEE 802.1Q CNC), especially in networks with mixed criticality traffic. The lack of such monitoring mechanisms in the IEEE P802.1DG TSN profile can result in suboptimal network performance, loss of real-time capabilities, and even security vulnerabilities.

Scalability:

IEEE P802.1DG does not address scalability concerns when TSN mechanisms are deployed across large, heterogeneous networking environments. Scalability requirements should be specified as a range of spatial (i.e., topological) dimension of in-vehicle networks (as small area networks), e.g., by using a measure like network distance. Scalability requirements need to be defined when considering "nontrivial" system context models where TSN domains might be extended over the vehicle local context only (e.g., TSN communication services across a cluster of vehicles interconnected via direct V2V interfaces, or via V2X and mobile edge hosted 5G TSN services, or a vehicle with diagnostic connectivity for development purposes). For this whitepaper, we will only consider scalability within the scope of the in-vehicle network, so this gap is not as relevant as the ones mentioned earlier.

Configuration and management:

The profile does not define how TSN configurations can be managed in scenarios that require dynamic updates of communication requirements and/or dynamic orchestration of software components. SDN, however, provides dynamic network configuration capabilities to applications and integrators. The absence of guidelines on how to handle TSN configurations dynamically may make the profile insufficient for use cases that require such dynamic configuration capabilities during the lifecycle of an SDV.

Northbound APIs:

No standardized northbound APIs and control protocols that SDN controllers can use to abstract and manage TSN components are defined by the automotive profile.

Interoperability:

IEEE P802.1DG does not address interoperability with existing TSN management protocols and modeling languages such as NETCONF/YANG, which are suitable and even necessary for SDN deployments.

Without stringent standards for interoperability, issues are highly likely when implementing an SDN-controlled environment, complicating network management and potentially leading to fragmentation and inconsistencies in network operations.

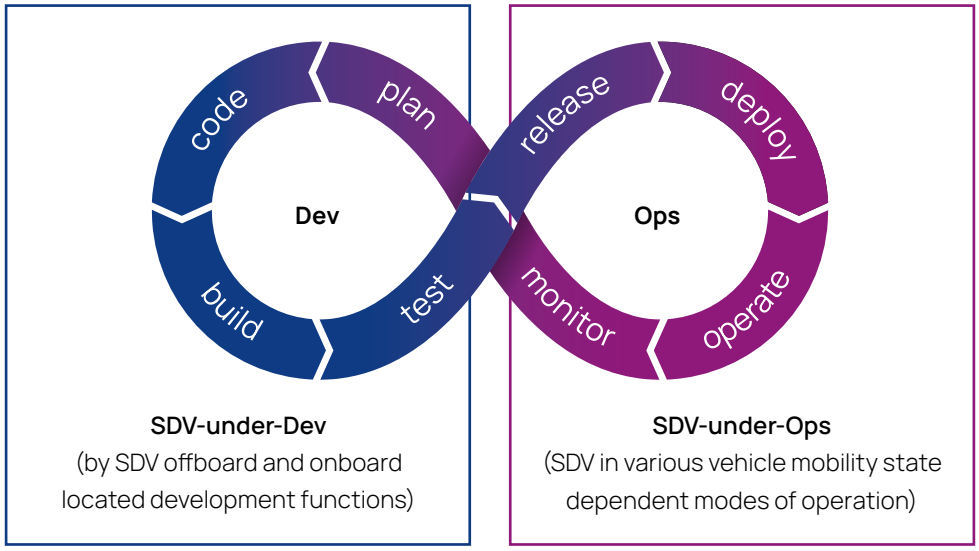
10 For experts: network policies related here to policy rules applied on traffic flows of Ethernet MAC frames like TSN streams. The constituents of such policy rules are rule conditions (for traffic identification) and rule actions (which determining the particular network service like QoS support here).

4. Use Cases

Use cases for Vehicles-under-Development are already well established. Hence we'd like to focus on TSN-related use cases for SDVs-under-Operation.

4.1 Generic and TSN-dedicated SDV function layers

The functionality of the SDV information system may be structured in the layers of application, communication and resource functions in general. Figure 7 outlines the scope on TSN-related SDV functionality, we'd like to focus mainly on in-vehicle networking and communication services, looking at the DevOps lifecycle model of Fig. 5:



Feature = layered SDV function of type:

1. Application function („App")
2. Communication function or
3. Resource function

Scope Whitepaper: Feature = SDV function with impact on In-Vehicle TSN:

1. Distributed application requiring TSN stream(s)
2. TSN-based communication services
3. Network functions based on physical/virtual TSN resources

Figure 5: The SDV DevOps lifecycle

The concept of a feature ("Merkmal") refers to a self-contained function or a property of such a function.

4.2 TSN mutability concept for SDV-under-DevOps

SDV mutability ("Wandlungsfähigkeit") in general and for the in-vehicle TSN in particular requires the adoption of the ability of pervasive interoperability and observability, resulting in TSN configurability and supervision capabilities, supported by a SDN-oriented control and management paradigm. TSN mutability includes all physical and virtual TSN infrastructure resources as well as all software-realized TSN communication services, consequently divisible in hundreds of TSN-related use cases. In the following sections, we'd like to consider some of them.

4.3 TSN-related use cases for SDV-under-Dev lifecycle phases

In-vehicle network engineering and in-vehicle communication services engineering is basically conducted in the development lifecycle. The TSN SDV-under-Dev focus is dedicated to all static network and communication service objects, such as initially planned communication matrix and all associated static TSN streams (i.e., permanent end-to-end connection services). Such TSN engineering business is already challenging due to the variety of traffic types, different QoS objectives, careful, efficient TSN resource management and the like.

However, we focus on the much tougher problem and look at some selected use cases from the vehicle operational phase ("Wirkbetrieb").

4.4 TSN-related use cases for SDV-under-Ops lifecycle phases

Agile automotive software development, associated with early and continuously value delivery, requires a holistic DevOps lifecycle engineering model.

An example envisioned feature cycle (Fig. 6) demonstrating the capability offered by this SDN-based SDV concept is that a particular vehicle or mobility fleet is reporting issues, either autonomously (by UC#8, e.g., onboard operational state supervision, alarm reporting, anomaly detection), or through explicit human report. The feature relates to a certain SDV function, like Software-in-the-Loop (SiL) artefacts, but also Hardware-implemented functionality, dependent on TSN communication.

Scope of example use case selection:
- not SDV-under-Dev because represents legacy in- vehicle network and communication services engineering use cases
- focus on SDV-under-Ops due to novelty, additional required SDV capabilities and operational challenges

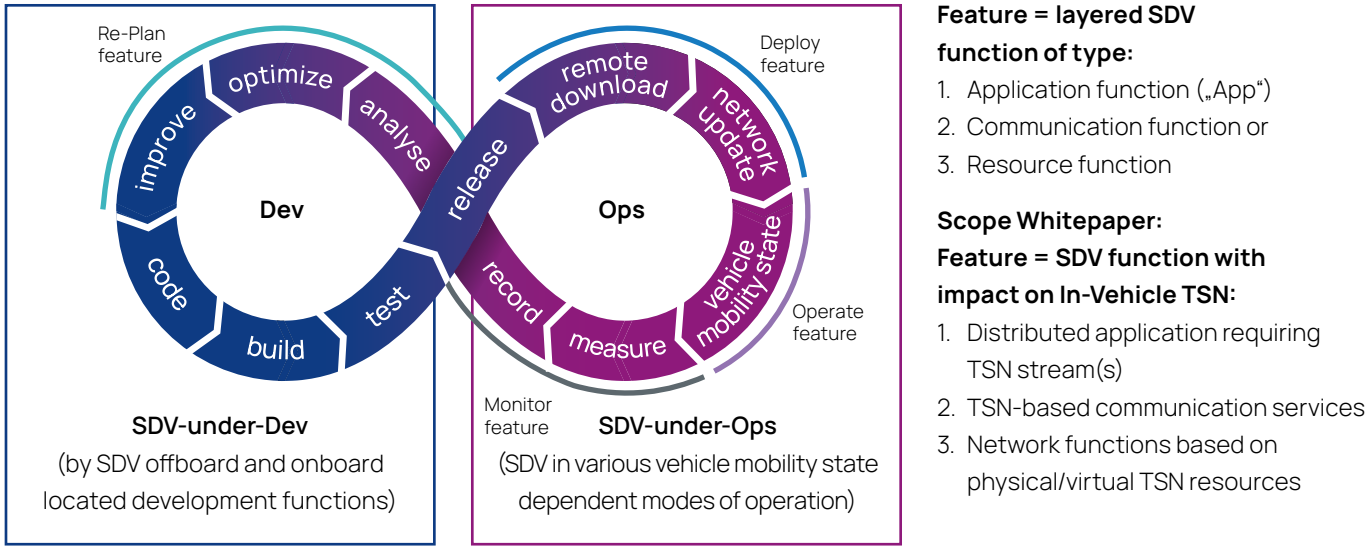


Figure 6: Simplified view of selected use case examples for vehicles under operation

In order to investigate the observed or reported issues, a development team compares the planned with the actual in-vehicle TSN state, modifies (UC#2) the runtime environment (e.g., for filtered TSN data acquisition) on some portion of the vehicle fleet to provide more specific state data from the in-vehicle TSN system, which is logged to onboard storage (i.e., network management datastores and logging files, UC#7,8). It must be noted that "in-vehicle TSN state" is a fairly general statement: it subsumes the entire variety of stateful TSN objects (e.g., network element local or network wide global, service layer dependent, or self-contained TSN network functions). For instance, a typical planning object is the intended "TSN stream communication matrix", i.e., the number and topological type of all TSN streams which are expected to be in-service when the SDV IVN is initialized. Furthermore, such use cases are normally inherently coupled with fault management and alarm reporting services (omitted here) for addressing unsuccessful scenarios.

Supplementary, a dedicated online test function (again TSN service based) might be temporarily activated (UC#1).

Automotive knows already an extensive framework of diagnosis services (standardized e.g., by AUTOSAR [21], the ISO defined Unified Diagnostic Services (UDS) [22] or the ASAM work on SOVD (Service-oriented Vehicle Diagnostics) [23]. Any SDN-based SDV architecture is required to over-

come the existing silofication and to allow a fully integrated diagnostics approach. Such SDV-under-Diagnosis capabilities (legacy or new) are expected to be fully integrated in the SDN-based SDV information and TSN context. That is feasible (because the SDV management plane deals with diagnostic information) and should be achievable in the long-term (by adaptation, outphasing, integration of legacy technologies).

Iterative analysis and evaluation of the collected TSN data sets might involve Digital Twins (DT) of the in-vehicle TSN system, allowing validation and verification tasks in the virtual SDV space. Such model-based development activities demand for topology-related YANG models are beyond the scope of this whitepaper.

Finally, this analysis might result in the conclusion to update the identified existing feature (UC#2), leading to modified TSN traffic demands, and to complement the feature by a new application function (UC#3). Both use cases will impact in-vehicle TSN traffic demands, either by the modification of existing TSN streams or the establishment of new TSN streams (in whatever communication topology).

The effectiveness, performance, and quality gates of the intended in-vehicle TSN changes needs to be cross-checked (UC#7,8). Even if the expected probabilities of reported alarms (UC#8) and mandated TSN updates (UC#2) might be very low, they could be operational critical or safety critical. Consequently, integral support of fault management and alarm reporting (ITU-T Recommendations M.3703, X.733 provide the framework for that management service categories) is mandatory. Support of fault tolerant behaviour, protection of dedicated SDV resources, the ability in transitioning to safety uncritical states are all mission critical capabilities, hence mandatory functional requirements for the SDN-based SDV controlling, operating, and managing system.

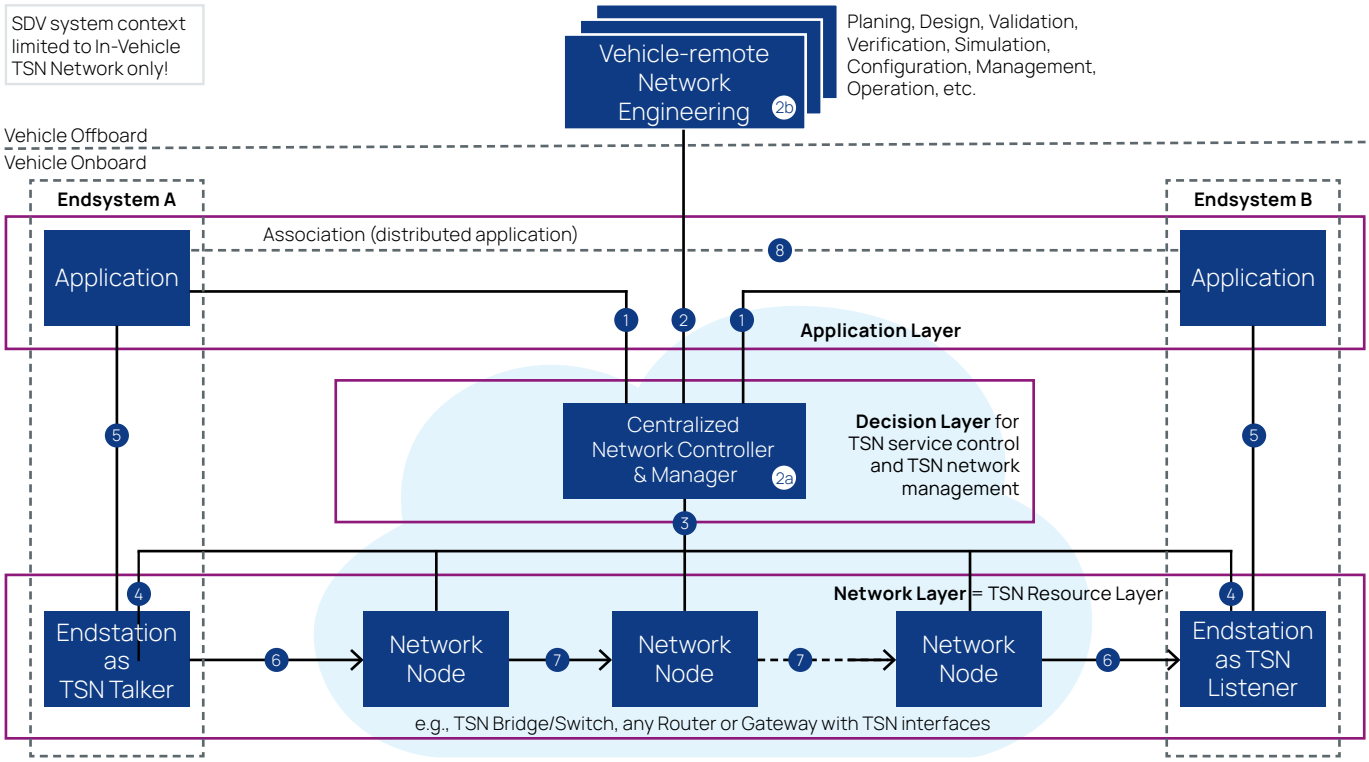
The initial lack of assured data and knowledge about the traffic demands of new user application functions may lead to uncertainty and risk, which will be addressed and mitigated by an initially too-conservative TSN traffic engineering approach, when the TSN streams are originally established. A probe-based monitoring of the TSN talkers (UC#8) allows the improvement of the associated TSN traffic source model. A refined workload model for the entire in-vehicle TSN system may be much better estimated, allowing the subsequent tuning of the distributed TSN resource allocations, leading to improved resource efficiencies (UC#4) as well as adjusted service level assurances (SLAs, UC#5) of the new TSN stream(s), without QoS impact of all existing TSN streams.

All outlined use cases impact the in-vehicle TSN-under-Ops system.[11] Each use case will lead to procedures in the SDV control and/or management plane. Let's look at the triggers in more detail.

11 All existing and new requested communication services use commonly shared communication resources, basically leading to mutual dependencies (and possible interferences in QoS-unmanaged, best-effort systems). However, the SDN-based SDV approach allows avoidance of such interferences by applying system-wide resource management (planning, reservation, allocation, monitoring, etc of dedicated SDV resources).

4.5 TSN-related use cases from SDN-based SDV perspective

To better understand the TSN related use cases for an SDN-based SDV, first an overview over the proposed architectural model is provided in Fig. 7 (see Section 6 for more details). This model is suitable to address all the following example TSN use cases.



Function

Network Service Layer (not Protocol Layer)

In-Vehicle Network (IVN) incl. Endsystems

Traffic (undirectional, User Plane)

1 Application-triggered TSN Communication Service

2 Third Party-triggered TSN Communication Service

2a by Vehice-local Manager

2b by Vehice-remote Manager

3 Southbound Interface (SBI)

Communication technologies (management protocols here):
3.1 IETF NETCONF (default)
3.2 IETF CORECONF (if constraint network devices)
3.3 Automotive proprietary (if backward compatibility with legacy automotive software systems like AUTOSAR, JasPar)

4 Endsystem-located TSN network resources are principally managed by the central management authority „network manager"

5 Access Point of Communication Service by Application (relates to an ITU-T X.200 (Layer x)-SAP (Service Access Point))

6 User-to-Network Interface (UNI) = E-UNI (if Ethernet-based User Plane, ITU-T Y.2113)

7 Network-to-Network Interface (NNI) = E-NNI (if Ethernet-based User Plane, ITU-T Y.2113)

8 Association (ITU-T X.200), i.e., logical relationship between all application endpoints (APP)-EP of the distributed application function.

Figure 7: Simplified SDN-based SDV functional architecture with focus on in-vehicle TSN

SDV control/management plane procedures can be triggered by different entities in this model. The following table provides an overview over the possible trigger for the introduced use cases.

Location of trigger:	Discussed use cases (UC):
Application endpoint (1)	2, 3
Vehicle local controller/manager (2a)	4, 5, 7, 8
Vehicle remote controller/manager (2b)	4, 5, 7, 8
NOTE 1 – Example mappings represent only a very high system abstraction level view.	
NOTE 2 – The procedures are also called (in SDN standards) "user-controlled" (1) and "operator-controlled" (2a,b).	

The introduced examples of basic use cases could be part in higher-level user scenarios. Two examples may illustrate such embeddings:

Example I: automotive traffic engineering

The operational context relates to the previous section on Traffic engineering (TE) of in-vehicle communication traffic. Let's suppose that an SDV was engineered and planned initially with a static TSN communication matrix with hundreds of concurrent TSN streams.[12] A future SDV Devs phase is used for the introduction of new or/and the adaptation of existing vehicle applications leading to a situation with a mixture of sufficient and insufficient knowledge about the underlying TSN stream traffic characteristics. E.g., now, 5% of the TSN streams lacking not-yet-precise-enough traffic profile descriptions. That TSN streams where consequently traffic engineered in a conservative manner by using a constant peak bitrate as worst-case assumption. The SDV becomes operational because all TSN streams will provide the expected QoS objective, of course at the expense in wasting network resources, which might be significant.

Therefore, we can derive the following typical user scenarios:

Monitor all not-yet-well-known TSN streams in order to build-up knowledge about more accurate traffic source models (relates to UC#8)

Modify the (static) TSN communication matrix by adapting the traffic specification of such TSN streams (relates to UC#5). Such an adaptation should be performable in dedicated SDV mobility states for an SDV-under-Ops.

Example II: conditional, temporary QoS objectives dependent on changing level of automated driving

In an in-vehicle network, the QoS requirements of certain data transfer services can vary depending on the current driving mode. For autonomous driving, large amounts of sensor data from radar, LiDAR or camera systems with high QoS demands must be transported through the network, processed, and set points must be transmitted to the actuators. In contrast, during manual driving, some of this traffic may be irrelevant or has significantly lower requirements regarding QoS. During parking, proximity sensor data becomes highly important. Therefore, in-vehicle networks can greatly benefit from flexible network management that optimizes network utilization according to the current driving mode. SDN can meet these needs, as it allows for the deployment of new network configurations during the transition between different driving modes. This approach enhances network utilization, reduces possible overprovisioning, optimizes the network configuration for each driving mode, and even enables new functionality such as an energy-saving mode for the network layer.

This user scenarios relates to e.g., UC#1 and UC#8, or the automotive SDN level 5 increment of "multiple-configuration set" support (see Subsection Description of automotive SDN levels in Appendix III).

5. Elements of the SDN for SDV

5.1 Standardized management data models (YANG)

Any effective and efficient SDV control and management plane located system functions are dependent on a single, common representation of the SDV (user plane functions). This is a fundamental requirement from control, operations, and management, leading to standardized information (or pure data) models as common representation forms in information systems [24]. Such a data modeling language is YANG with its integrated management-oriented language features. Like the prominent distinction between configuration data and state data. The applicability of YANG for automotive SDN is further detailed in Subsection Leveraging YANG-based protocols for the SBI in an SDN-based SDV.

12 There are typically multiple thousands of intra-vehicle application-level communication associations (→ Communication Matrix). They are multiplexed to a smaller number of lower layer connections (here TSN streams) by upper layer protocols. The specific protocol layer, protocol and multiplexing scheme is conditional, e.g., dependent on automotive communication paradigms like "signal-oriented", "service-oriented", "data-centric".

5.2 SDN Controller "Northbound" Interface

In the following we will focus on the Fully Centralized model ([12], cl. 46.1.1.3) for stream configuration. This model defines the CNC (Centralized Network Configuration) and CUC (Centralized User Configuration), constituting a control and management plane- and a user plane entity respectively. In the context of SDV system context the CNC – comprising TSN service control and -management functions – resides in the Decision Layer. Conversely, the CUC conceptually resides in the Application Layer. The CUC may have different characteristics – it could be an (online) server instance or an (offline) planning or management tool; in either case, the CUC acts as a proxy to communicate QoS requirements for user plane communication to the CNC.

Based on a forementioned relationship between CUC and CNC – or application and decision layer, respectively – the Northbound Interface enables access to the service provider (CNC), establishing a Service Level Agreement (SLAg [25]) in general and an embedded Service Quality Agreement (SQA [26]). Consequently, due to the established SLAg, the service provider (CNC) is also responsible for Service Level Assurance (SLAs [25]), which is accomplished by means of monitoring and diagnostics. Especially the network diagnostics may require integration into the higher contexts on vehicle level, since manifold diagnostic facilities are already established in the automotive domain.

The Northbound Interface (NBI) covers the two control/management plane interfaces 1 and 2 in the architecture model outlined by Fig. 8. The pure control plane interface 1 may be qualified as Application Control Interface (ACI or NBI:ACI) because allowing "programming of communication services" directly by user applications (i.e., user-controlled TSN streams). Interface 2 covers the vehicle remote communication part of the NBI. The programming of an application function (AF) via an API is always subject of the programmer role, the User-of-Application-Function (UAF). Such an API might be a local or remote interface. In the latter case, it is network protocol based. However, the carried information components (for application programming / control) are identical in both cases. The means how the Northbound Interface is implemented, e.g., an (online) network protocol for ad-hoc requests or a programming API for offline integration into a single planning tool, may vary. However, the semantics of the management objects conveyed via the Northbound Interface must not change to keep a proper way of abstraction, which is e.g., an essential requirement for an intent-based SDV management (see e.g. [27]).

The management objects defined in [12] and extended [28] fulfill the need for abstraction as they are kept independent of any encoding into a certain data format, as well as independent from the actual means of deployment (like a communication protocol) to a (remote) device. However, [28] recommends and provides an encoding of mentioned management objects to YANG .[13] The use of this exemplary format and especially the means of deployment depend on an overall network control and management strategy, which naturally must also be integrated into – or at least aligned with – the larger vehicle context.[14]

5.3 TSN streams and operations

5.3.1 Stream definitions

The identifier of a stream provides inherently also its semantical definition of the underlying information concept "stream". SDN Network Layer functions, which are required to unambiguously identify, process and ultimately relay a particular stream, are defined by the control plane and subsequently deployed by the management plane; this comprises all necessary SDN Network Layer configuration objects supporting and ensuring a certain QoS for the respective stream(s).
Hence, there are at least two plane-specific stream identifiers required:

> One for the associated control and management plane protocols and functions, as reference and inter-plane binding element.

> One for the user plane, carried in the PCI (Protocol Control Information) of each (UP)-PDU (here: Ethernet MAC frames).

13 To note, these (YANG) definitions on the northbound interface are for convenience and clarity only; this is a different goal than the (YANG) models defined for vendor-independent configuration of various devices on the southbound interface. Roughly, the NBI uses YANG for the description of communication services, whereas the SBI uses YANG for the description of communication resources.

14 Thus subject of SDV profile specifications like an Automotive SDN profile specification in protocol profiling all relevant SDN interfaces, data model profiling concerning YANG data required for the intended suite of automotive SDN services, communication service security profiling, etc.

The solution defined by the IEEE 802 for such identities is given in [12] and refined in [28], cl. 46.2. There are two related but different types of stream identifiers (see also Glossary entry "TSN stream"), i.e., a dedicated stream identifier for the user plane, basically used by forwarding and QoS mechanisms, and another stream identifier for usage e.g., at NBI, SBI or control application functions of the SDN service controller. The stream definition on the northbound interface focuses on application control with regards to the communication service requested or in use. The description of such TSN stream based communication services also comprises properties such as communication topology and directionality, and endsystem connectivity besides QoS related information components. Multiple aspects have to be considered, primarily the distinction between Talker- and Listener specifications, as required for the binding of a Tspec and Qspec to source and destination side related UNIs (User-to-Network Interface). The encoding of stream identifiers might support multiple formats, dependent on machine or human-based processing function. Text encoding is typically preferred for human readability. The stream representation model (as YANG data model) as used by the IEEE 802.1 CNC (here for SDN-based SDV: the Centralized Network Controller & Manager) for e.g., configuration purposes is divided in three high-level groups, i.e., Talker-, Listener-, and Status-group.
The common control and management elements (as defined by YANG data models) for Talker- and Listener group are:

Interface-capabilities, enumerate the capabilities of all interfaces (ports) of an end system, most notable VLAN and redundancy. However, due to the absence of other functions, e.g., SCHED (formerly [29]), it is assumed that the realization of behavior is specific to the respective end station (e.g., TSN source shaping [30]).

End-station-interfaces, listing the interfaces of a specific host, identified by MAC addresses

User-to-network-requirements, stating quality specification (Qspec) related properties of the stream in terms of maximum latency and degree of redundancy (Talker only).

Interface-configuration, stating the stream addressing as well as the timely behavior of the Talker in case of time-aware streams (time-aware-offset)[15], after successful planning.

In addition to the common elements, the Talker also contains

Traffic-specification (Tspec), defines the quantitative, traffic description related properties of the stream with respect to frame size and interval (period). In case of time-aware streams, a Talker may also define a certain time window within the interval at which it is able to initiate transmission. In the latter case, the CNC will decide for a specific time-aware offset within this window.

Data-frame-specification, allows a user to specify the addressing of a stream, especially in cases where the addressing is already fixed by the Talker application and -host (or up to CNC decisions).

These definitions offer a sufficient degree of independence between user- and the control plane, omitting specifics of the Ethernet MAC frame forwarding in the user plane; hence in full conformance with SDN principles, the stated definitions are independent to TSN mechanisms provided in the network itself. Consequently, this abstraction allocates the responsibility to realize a certain behavior to the CNC.

15 For experts: some TSN QoS functions could be operated with or without relation to a network-wide common time baseline (time-aware versus time-agnostic). The time-agnostic mode relates to the native ATDM (Asynchronous Time Division Multiplexing) mode of the Ethernet data link layer. The time-aware mode leads essentially to an STDM emulation service (Synchronous Time Division Multiplexing) over ATDM, which would allow to support more stringent QoS objectives (at the expenses of a reduced statistical multiplexing gain and control costs).

5.3.2 Stream operations

The abstract stream definition and the distinction between talker and listener, enables a wide variety of scenarios. Especially the StreamID ([28], cl. 46.2.3.1) enables association of talkers and listeners to a single stream in a weakly coupled manner. The operations provided by the Northbound interface via its SLAg depend primarily on the overall control and management strategy. This strategy will typically comprise various aspects, e.g.

Service request granularity, i.e., if individual streams or even (SDN) clients may be added, or if only the full set of streams can be planned as one. The granularity of the service requests typically also depends on the actual deployment; an (online) server CUC/CNC could cover per-client requests at the cost of reduced solution space. For an (offline) planning tool, a full-network granularity would be the preferred approach. However, also per-stream granularity may be necessary when embedded in the full DevOps cycle due to optimizations and improvements over the lifetime of the system.

Access control to service, i.e., which application layer entities are allowed to request the service provided by the CNC via the Northbound Interface.
While this is most likely not an issue in offline planning scenarios, it is for online scenarios. Especially, in online scenarios this aspect is also closely related to safety and security considerations.

Global properties related to the CNC's planning strategy driven by fundamental considerations regarding the resources of the SDN Network Layer. This may especially relate to converged traffic, i.e., the full integration of all communication services, hence co-existence of service quality critical streams and uncritical ones in the same physical or virtual infrastructure.

5.4 Protocol integration & planning strategies

Due to the properties of TSN, i.e., a wide range of supported and not supported functions on each device, even on each port, require the concept of planning strategies. The actual planning strategy, describe the utilization of means available in the network, algorithmically realized by the CNC.

Naturally, every planning strategy has its pros and cons, as due to mentioned varieties trade-offs between the utilization of the different user plane functions are practically inevitable. One example would be the different traffic aggregate levels of per-streams vs. per-class planning; The former can conceptually minimize the required queue depth but increases the required number of gate-control-entries, while for the latter one, the situation is inverted.

In any case, the protocol-independent, information flow based Northbound interface definition by [12, D2.2] requires a so-called protocol integration, i.e., the embedment of YANG-based information components and service control procedures in a suited application layer protocol. Also, an offline planning tool follows this principle, where the user interface serves as CUC, and the underlying planning entity serves as CNC. Consequently, the CNC can immediately be re-used with other integrations. Other integration options may even include a holistic online scenario by establishing a network control plane protocol between application and CUC, the CUC collecting the requests and forwarding them to the CNC. However, the means how a CUC receives communication requests is explicitly out of scope by IEEE 802 (cf. [28], cl. 46.1.3.3).[16]

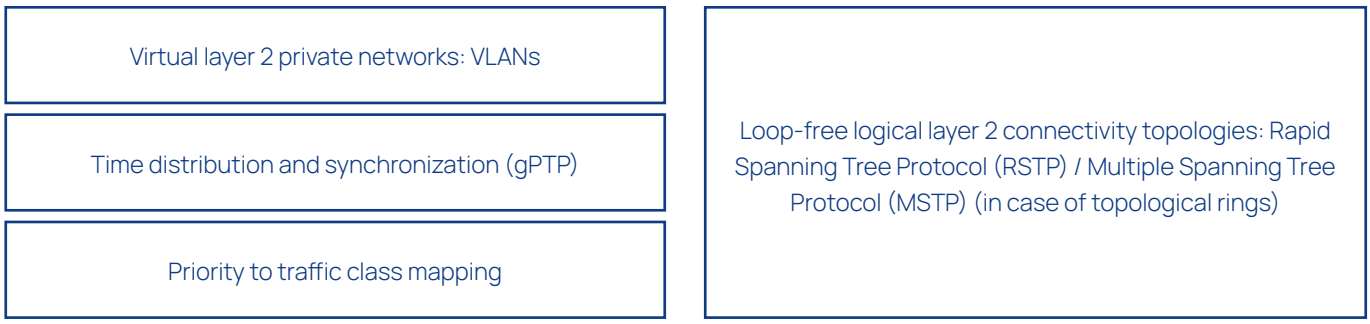
Protocol integration is also related to definitions offloaded to the CNC (as YANG model and protocol independent control procedures). While every property group of a stream definition may be part of the request, it may be beneficial to certain aspects centralized at the CNC, most notably all the basic network management configuration services like address management, virtual network management, or priority service management.

16 For experts: the IEEE 802 does not define entire network solutions like 3GPP or ITU-T but provides already some guidance for usage and integration of IEEE 802 technologies. The automotive SDN NBI does essentially not expose any automotive specific requirements, hence, a common, open, standardized cross-industry domain used control plane protocol should be targeted.

5.5 Traffic convergence

TSN provides a set of functional extensions to traditional switched Ethernet networks. Due to this property[17], legacy best effort oriented Ethernet networks (like enterprise local area networks) can be integrated seamlessly. However, in order to effectively integrate both types of traffic, legacy QoS-agnostic and TSN QoS-aware Ethernet into a common, single operated Ethernt network, both aspects must be considered and aligned accordingly (the typical subject of QoS engineering activities).

The communication service specific CNC view focuses on services related to streams only, i.e., unidirectional traffic flows of information. This CNC function is provided by the SDN service controller (Fig. 10). Such communication services are embedded in a virtual/physical network infrastructure, which is under responsibility of the network management manager, which also takes care of all non-TSN-based communication services. While it needs to consider existing configuration in the networks (e.g., VLANs) and may utilize infrastructure protocols (e.g., generic Precision Time Protocol (gPTP)), it does not participate in their configuration. Consequently, an additional decision layer entity is required to "prepare" the network infrastructure. Typical aspects for this step comprise setup and initial configuration of



This basic setup of the network, including network operational protocols like gPTP or MSTP, must be provided as input to a CNC, as true Service Level Assurance (SLAs) can only be provided if the SDN Network Layer as a whole- and not only streams - are considered and meaningfully aligned. Consequently, the sum of SDN Decision Layer entities is subject to an overall network control and management framework.

It shall be noted that this framework may be driven by emerging TSN profiles for the respective industry domains (as specified by IEEE). However, it can be foreseen that mentioned TSN profiles cannot cover all aspects of stream and non-stream oriented communication traffic due to the potentially wide variety of application scenarios, even within the automotive domain alone. However, it is considered feasible to establish a minimum viable baseline regarding SDN Network Layer capabilities and available TSN resources as well as their dimensioning.

5.6 SDN Controller "Southbound" Interface

In the proposed Software-Defined Networking (SDN) architecture the Southbound Interface (SBI) provides the communication interface between the SDN Decision Layer and SDN Network Layer. As traditional in-vehicle networks (IVN) typically are static, the SBI introduces dynamic access to the network nodes. The SBI can be used for two main purposes: configuration of the network nodes and retrieval of status data of the network nodes. To ensure interoperability this SBI and the underlying data needs to be clearly defined. A CNC should not depend on vendor-specific behavior of network node implementation, but rather communicate to all nodes similarly and therefore YANG-based (as an abstract model) of node capabilities, configuration data and state data is obligatory.

As the SDN Network Layer in an SDV typically serves distributed, often time-critical applications with high demands to safety and security additional requirements to a southbound protocol arise. A southbound interface protocol therefore needs to meet the current standards of security and allow the safe deployment of configurations to the whole network.[18]

As automotive components are under high cost-pressure, network nodes usually are not equipped with large amounts of memory or performance. Implementations of southbound interfaces in an SDV therefore should ideally be very lightweight and efficient to acknowledge constrained automotive network resources.

17 The capability in defining comprehensive QoS overlay architectures for TSN-enabled Ethernets.

18 There are two security service levels here: the SBI protocol shall support a secured transport option, and the SBI protocol shall be applicable for security management services. Both aspects are supported by NETCONF and CORECONF.

5.7 Leveraging YANG-based protocols for the SBI in an SDN-based SDV

To fulfill these requirements, a standardized way to model configuration and state data is crucial. The IETF standard Yet Another Next Generation (YANG) data modeling language is already widely adopted in ICT as such and as well perfectly matching the model-based SDN paradigm. YANG data is organized in hierarchical structures in a modular manner, reflecting the structure of the functional system architecture. YANG data models are organized in modules, that can be reused, extended, and inherently support versioning. Furthermore, YANG supports a wide range of data types, constraints and validations, allowing a precise modeling of network data, essentially all capabilities as demanded by operation and management of distributed systems. The syntax of YANG is simple and intuitively human readable, resulting in a clear and easy to maintain data model. Still, YANG models can be efficiently parsed and there already exists a wide range of YANG tooling for validation, testing and automation.

5.8 Management protocols

Modern protocol engineering recognizes some fundamental communication service architectural requirements for application protocol layer protocols (such as management protocols): e.g.

Decoupling of application layer protocol and carried payload data in separate specifications.	Decoupling of application layer protocol with underlying transport options (i.e., no entire binding to a lower layer protocol stack).[19]
Decoupling with regards to optional communication security functions.[20]	Application layer protocol management by the network management plane and not the protocol itself.[21]

NETCONF satisfies all these basic protocol engineering requirements. E.g., requirement 1: YANG is specifically designed to work with the IETF Network Configuration Protocol (NETCONF) as a classic management protocol, but fully decoupled from NETCONF, hence usable also by other protocols. NETCONF uses an RPC-mechanism for communication of configuration as well as state data, using the client/server pattern as representation of the hierarchical management roles manager/agent. In the proposed SDN architecture in Figure 8, the network nodes act as servers while the Centralized Network Controller and Manager acts as the client. The application layer protocol NETCONF may use unsecured or secured IP transport services (like SSH or TLS in case of NETCONF-over-TCP). On top of this layer, RPC-messages or notifications are exchanged. Operations like e.g., edit-config operate on YANG based datastores using XML encoding. In NETCONF multiple management datastores for configuration and state data are defined separating between a startup, a candidate, and a running configuration. This decoupled concept supports safe deployments of new configurations at runtime. Furthermore, NETCONF defines rollbacks-on-errors, in case the deployment fails for some reason. These concepts allow robust implementations of southbound interfaces.

19 In order to expand the number of network context options. E.g., there might be multiple beneficial IP transport protocol options in case of Internet-based usage of NETCONF.

20 There are again different security context models, leading to different qualities of communication security support for the upper layer protocol.

21 Example 1: the ASAM-defined automotive XCP violates that requirement. Example 2: the requirement is not existent for the IETF-defined CORECONF proto col due to the assumption of constraint network and/or constraint network devices, resulting in combined user/management plane protocols (here: IETF CoAP).

5.9 CORECONF: A management protocol alternative for constrained devices

As discussed in Section 5.7, in an SDN-based SDV often network nodes are constrained devices (see IETF RFCs 7228, 7547, 7548). A suitable southbound interface protocol for these devices can be the CoAP Management Interface (CORECONF), as designed for the IoT and the management of constrained IoT devices (see ITU-T Y.4115), which currently is an IETF draft. In contrast to NETCONF, CORECONF is a RESTful protocol. It utilizes the Constrained Applications Protocol (CoAP) methods to access structured data defined in YANG. With its much smaller implementation footprint and efficient handling of the REST interface, a CORECONF server can be implemented on constrained devices. To achieve the small footprint, CORECONF reduces the feature set of NETCONF and encodes data efficiently: E.g., In CORECONF there is only one datastore per device, data is encoded in the Concise Binary Object Representation (CBOR) and data exchange is mostly carried out in simple FETCH and iPATCH operations. As the complexity of the small area networks in SDN-based SDVs typically is relatively low and CORECONF allows a fine-granular YANG-based configuration of network devices it can be an efficient southbound interface for constrained devices.

To summarize, YANG based object model representations allow a high degree of control for network management in SDN-based SDVs. While NETCONF is a very mature and powerful protocol in this area, it might not be suitable for constrained devices. These devices can benefit from the simplified protocol CORECONF. As both protocols work with YANG data models, concurrent implementations in a Centralized Network Controller & Manager for both protocols are possible.

6. SDV Architecture Model

6.1 Introduction in SDN-based SDV architecture modeling

The distributed computing and communication system of a vehicle (like an SDV) may be abstracted by a network-centric view, see Fig. 8. The planning, configuration, control, operation, and management of that in-vehicle networks with its communication services requires onboard support functions besides the offboard suite of associated tools.

Fig. 8 outlines the option of a fully centralized network controller/manager function (according to [12]). The applied SDN layering principle (application, decision, and network) facilitates the structuring of the functional architecture, indicates relations to SDN patterns, and is furthermore consistent with typical communication architectural patterns.[22] [17, 31, 32] provide references to the major SDN architecture standards from various Standards Development Organizations (SDO). That SDO-specific SDN pattern emphasize specific aspects and perspectives.[23] Their used SDN terminology isn't fully consistent, but the various SDN architecture models are not in contradiction, they could be rather mapped to each other. The whitepaper here makes an attempt of a harmonized, cross-SDO SDN architecture model, suited for automotive SDN. We will firstly introduce a generic model, acting as technology-neutral basic SDN-based SDV architecture. Then will augment that model by some crucial technologies before finally depicting an architectural model with dedicated SDN entities.

22 It is also consistent with the more technically oriented layering of IEEE 60802.

23 For experts: the ITU-T defined the framework and architecture as generic SDN model [33, 17]. SDN itself originates from the generic Resource and Admission Control system [34], which again were the architectural evolution of the predecessor control plane architecture with centralized elements (known as Bearer-Independent Call Control architectures). [35] provides a mapping of generic SDN to Ethernet-based networks. The generic SDN model by ONF [31] is consistent with [33], but emphasizes additionally business roles and service-oriented aspects. The IETF SDN model [32] focuses on the Internet and connectionless IP forwarding/routing aspects, leading to the IETF IP router and IP gateway specific plane models (like forwarding plane besides control and management planes). SDN itself was subject of network function virtualization: [36] discusses the mapping of SDN models.

That large scale area network SDN models may be downscaled to small in-vehicle area networks with Ethernet anchored QoS support. Such an automotive SDN model could be derived from [33, 17] and correlated with the IEEE 802.1Q proposed TSN control/management plane configuration architecture models (IEEE 802.1Q, cl. 46).

24 The SDV-under-Dev is "unconnected" because represented by technical artefacts, like software components, in context of vehicle offboard development environments.

6.2 Generic, technology-neutral SDN-based SDV architecture model

The model presented in Fig. 8 emphasizes the explicit separation in south- and northbound network interfaces for control/management purposes. Connectivity between SDV-under-Ops and remote network engineering support is normally realized by V2X (Vehicle-to-Everything) communication services (inclusive legacy remote diagnostic capabilities), whether over the air, cable, near-field communication, etc.[24]

The model uses a service layering scheme at the abstraction level of an entire in-vehicle information system (not to be confused with protocol layering schemes like the OSI BRM (Basic Reference Model)). Fig. 8 is omitting the information planes. Pure SDN architecture models focus normally on the control and user planes only ("due to the SDN control aspect"). However, a more comprehensive and complete architecture model implies also the explicit consideration of the management plane.

Therefore, we termed the central function as "Centralized Network Controller & Manager" (as composition of the SDN controller and Management manager functions) given that engineering goal. The term decision layer shall express that composition: that functional entity essentially generates policy rule decisions for policies applied to network functions or communication services. The outlined network layer summarizes the entire infrastructure of virtual and physical network resources (also known as resource layer in certain standards).

The user application traffic is transported via data transfer services along a routed network path. The indicated user plane interfaces UNI (6) and NNI (7) are necessary for the specification of Non-Functional Requirements (NFR) like QoS-related attributes in case of communication service qualities.

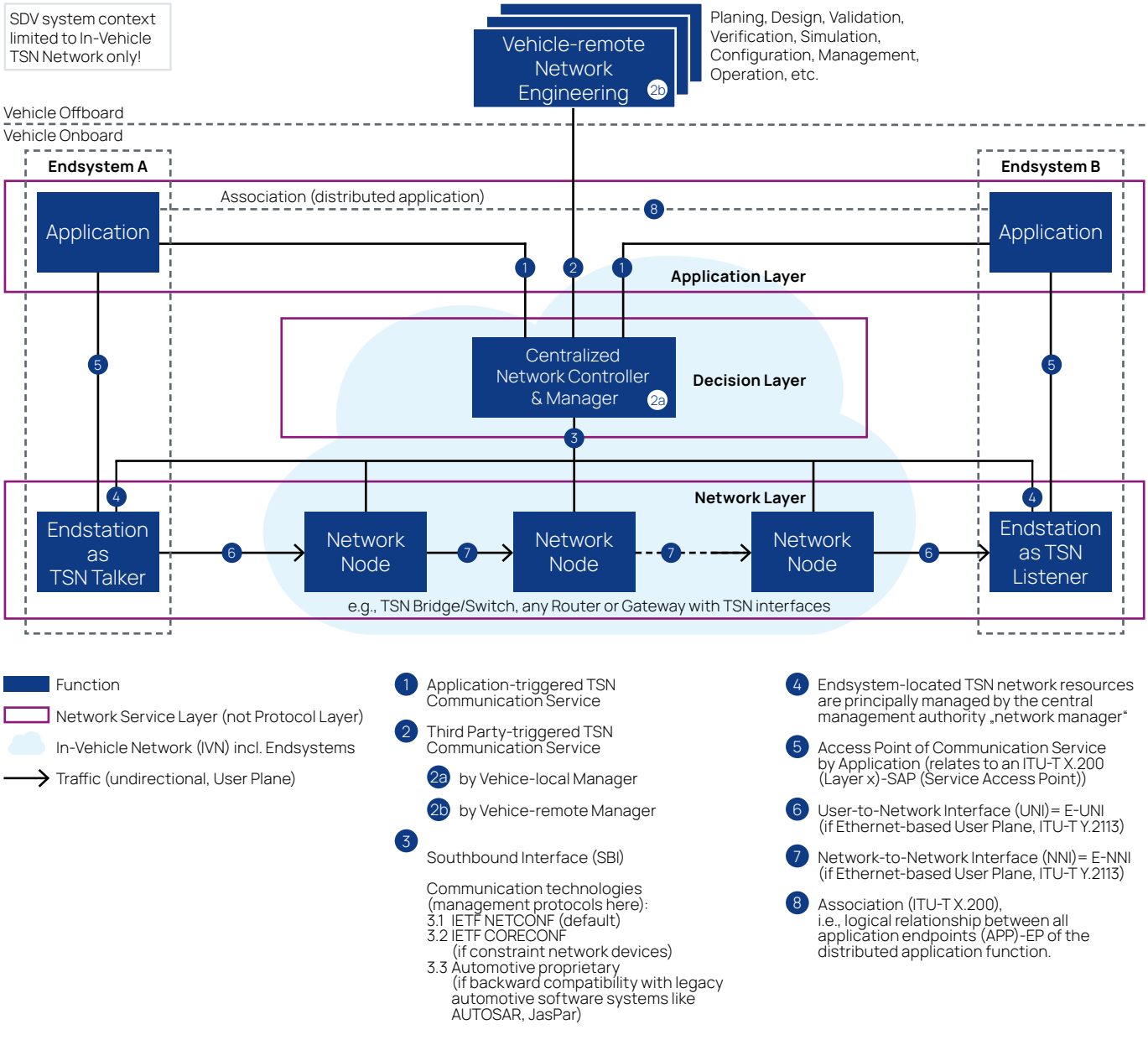
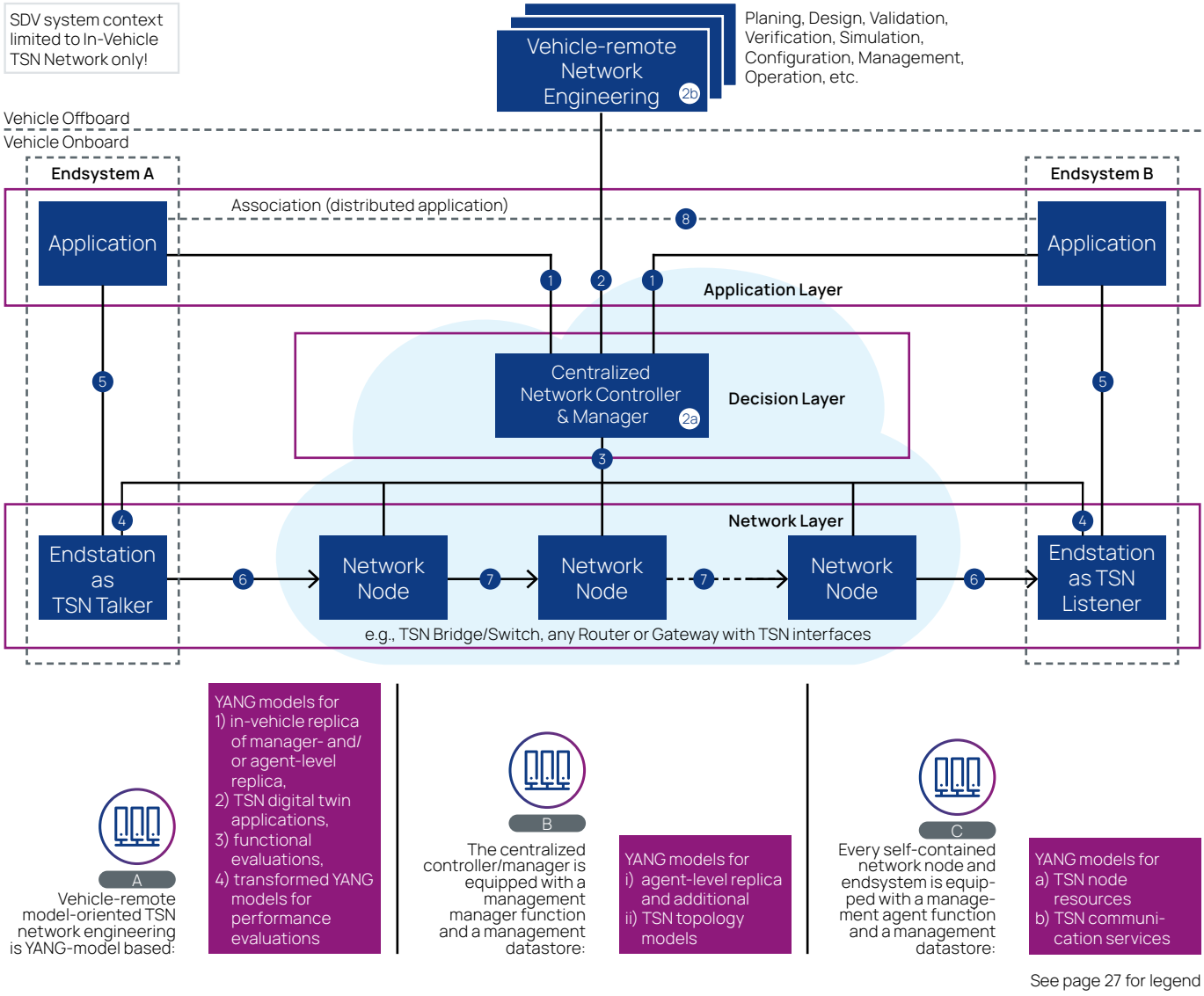


Fig. 8 – Functional, generic SDV system architecture with scope on the vehicle-onboard communication system

6.3 Technology-oriented, TSN-, NETCONF-, YANG-based SDN-based SDV architecture model

The SDN-based SDV uses an augmented Ethernet by TSN building blocks, primarily driven by QoS purposes. The SDV management plane will be operated by standard management protocols: NETCONF as default option and CORECONF in case of constraint network resources. Premise and mandatory is a homogeneous and common representation of all entities under control, operations, and management, leading to YANG as recommended and preferred data modeling language. Fig. 9 highlights the three major levels of YANG data models in use by SDN-based SDVs.



See page 27 for legend

Fig. 9 – Functional, TSN-based SDV system architecture with indication of management technologies NETCONF / CORECONF and YANG-based management data models

There are typically multiple actors which may cause, initiate, trigger or request the establishment, modification or release of a (TSN or Ethernet) communication service. Fig. 8 depicts two major options, but there might be other use case specific scenarios.

6.4 Major logical SDN elements in an SDN-based SDV architecture model

Fig. 10 expands the model by depiction of SDN and management dedicated entities. The SDN service controller represents that fully centralized control of TSN and non-TSN communication services. The control plane counterparts are the endsystem-hosted SDN clients, as requestors or addressee of application control activities. The notion of SDN client emphasize the business role, not to be confused with a protocol endpoint role.[25]

The network element and network management manager is also fully centralized and cooperates in concert with the SDN service controller concerning TSN stream control and management. Its counterparts are the management agents, equipped in every self-contained network element under management.

Both onboard deployed fully centralized controller and manager entities also have counterparts in the offboard system context, as

Remote entity for online, SDV-under-Ops control and management activities

Offline entity for SDV-under-Dev engineering activities

All these entities work on the same representation models, i.e., common and shared YANG data models.

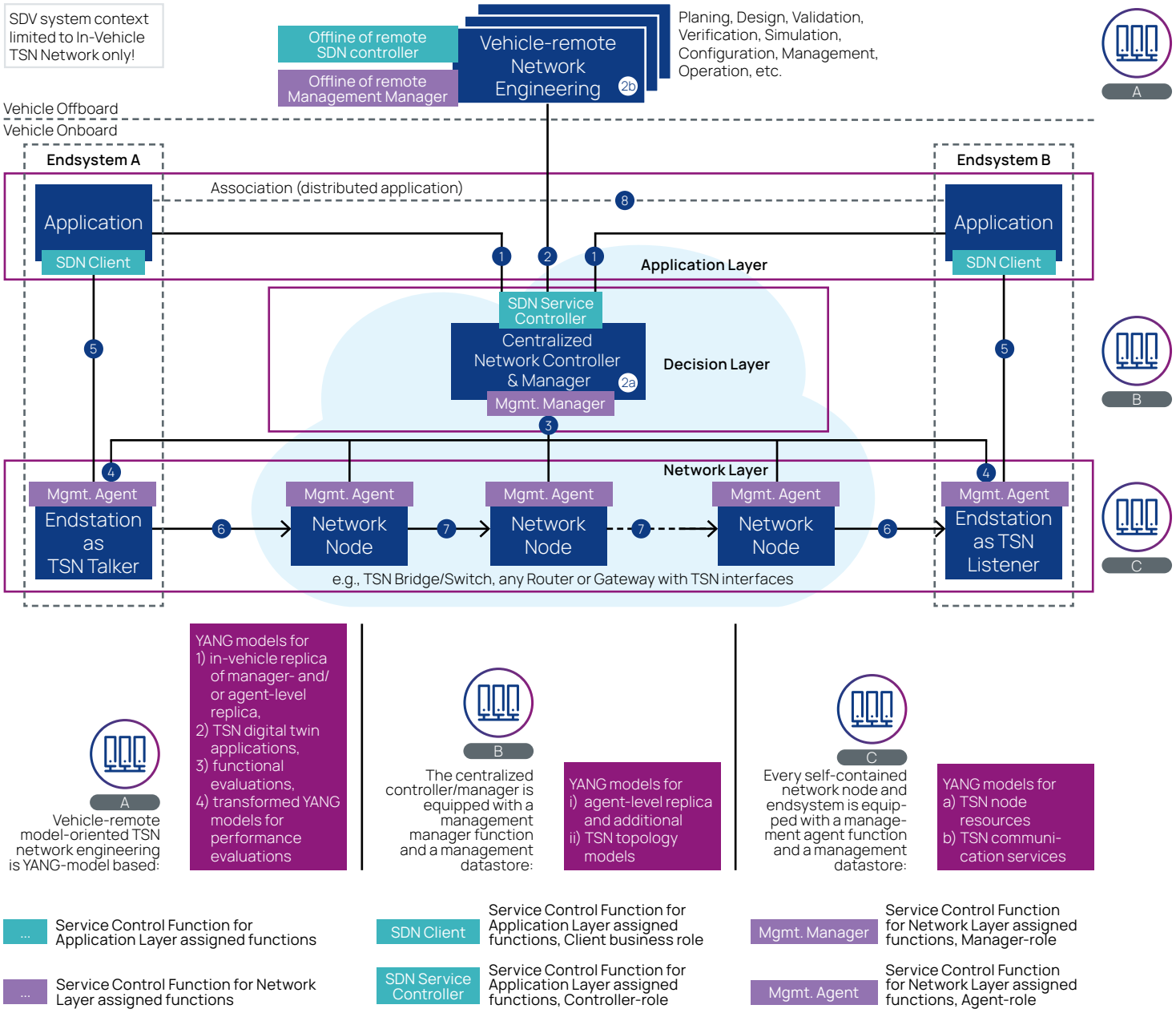


Fig. 10 – Functional, SDN-based SDV system architecture with explicit indication of SDN control and management elements See page 27 for full legend

25 A typical name for such a protocol endpoint role in application control or signaling protocols is e.g., User Agent.

A more detailed automotive SDN model may also indicate to complete set of communication participants, given by e.g., peripheral components, sensors, actuators, or storage nodes.

7. Summary

Following take-aways summarize the major conclusions:

The **SDN pattern,** adapted to the in-vehicle and vehicular system context, provides a **perfect solution for the SDV** control and management plane system architecture.

The **TSN-augmented Ethernet** is embedded in all other still existing legacy in-vehicle communication technologies, acting as **in-vehicle core or backbone** network.

The whitepaper explicates such an **SDN-based SDV** or automotive SDN system model, as an **overall engineering framework.**

TSN is consequently required for support of **high qualitative communication services,** demanding for an in-vehicle functional QoS architecture.

The SDV and its DevOps lifecycle model demand for the **ability of in-service control, operation, and management support.**

Ethernet is selected as **all-services-integrated network technology** for in-vehicle E/E systems.

The approach supports both **off-line and on-line** network scheduling and management.

8. Appendices

8.1 Appendix I – Automotive SDN for AUTOSAR and JASPAR

AUTOSAR (www.autosar.org) and its partner JASPAR (www.jaspar.jp) define frameworks for automotive software architectures, design and build process of distributed in-vehicle software systems. The introduction of control, operations, and management architectural patterns à la SDN is explicitly envisioned. The appendix provides a brief overview. However, any introduction or detailed automotive SDN discussions are beyond the scope of this whitepaper.

81.1 Automotive SDN for AUTOSAR

8.1.1.1 Platform view by AUTOSAR

The SW-centric view dominates AUTOSAR specifications, representing essentially a technical perspective. The notion of "platform" is a technical term and refers to the technical component (TC) of a single compute node in a distributed E/E computing system. It's also called a machine. Some AUTOSAR specifications use a technical system layering scheme like the following (Fig. 11):

Basic technical components for SDN-based SDVs and its deployment on technical nodes Requirement

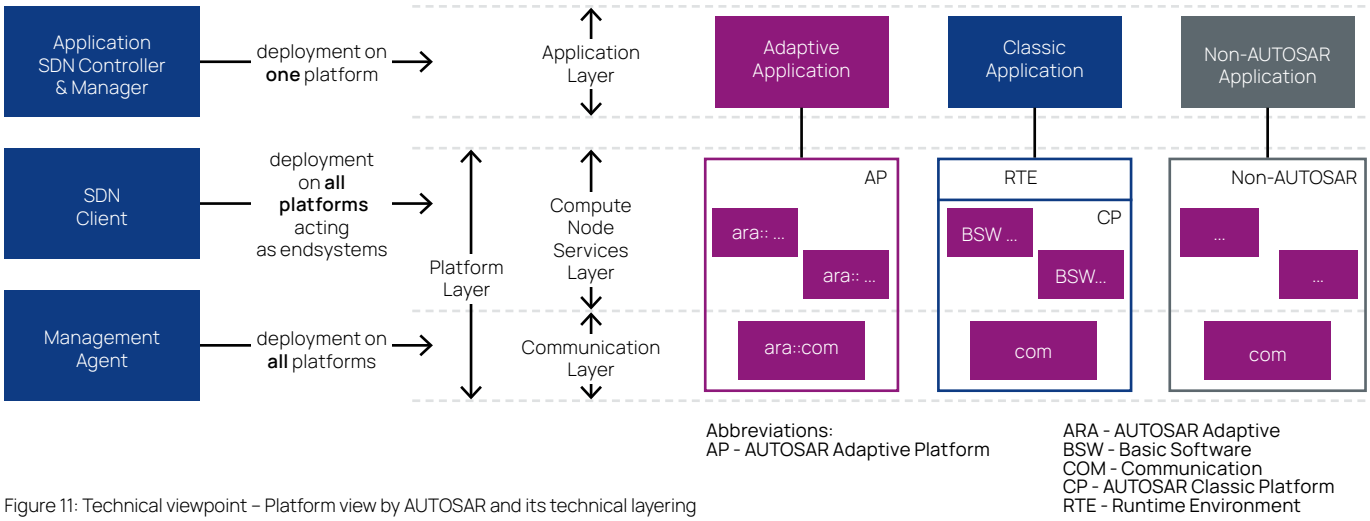
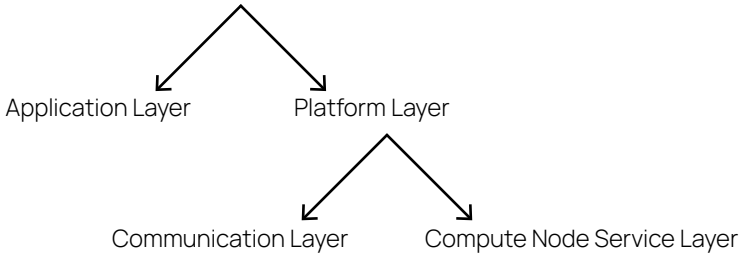


Figure 11: Technical viewpoint – Platform view by AUTOSAR and its technical layering

The applied technical layering scheme provides a coarse granular view (in contrast of the more detailed software layering schemes as used by AUTOSAR Classic and Adaptive Platforms (CP, AP)):



The selected, abstract AUTOSAR layering scheme is required for pointing out the relations to the functional SDN-based SDV reference model.

Where's the in-vehicle network? AUTOSAR doesn't consider self-contained network nodes, leading to the consequence that the entire in-vehicle network infrastructure of system context "AUTOSAR" must be allocated to AUTOSAR nodes. Every resource component out of that network infrastructure layer requires at least one assigned administrative authority, which is given by a AUTOSAR node (as technical platform). AUTOSAR allows to build communication-centric, more, or fully network-oriented node types by defining a platform configuration without any application layer software components, without any middleware (runtime environment), - from perspective of a top system abstraction level.

Consequently, the in-vehicle network infrastructure becomes fully integrated in platform nodes (indicated by the switching symbol in Fig. 12 which covers all platform-embedded communication layer components).

8.1.1.2 SDN-extended AUTOSAR

There are multiple options for an SDN-extended AUTOSAR system, i.e., for SDN-based SDV using AUTOSAR-for-SDV-under-DevsOps (see also architectural discussion in [37]). Fig. 11 indicates already that there are at least three network control/management plane related functions required, here constituted as technical components:

Functions	Logical Component (LC)	Technical Component (TC)	SDN purpose
The entire set of functions as provided by LC ...	LC "SDN controller & network manager"	Application component "SDN controller & network manager"	fully centralized controller / manager instance, in between the SDN NBI and SBI
	LC "Management agent"	Platform component "Management agent"	The instance providing the southern part of the SBI.
	LC "SDN client"	Platform component "SDN client"	The instance providing the northern, endsystem located part of the NBI.
NOTE 1 – The technical components are outlined at that top system abstraction level as monolithic components. However, all these components should be divisible by partitioning dedicated subset of functions in multiple technical components at lower system abstraction level. E.g., a particular technical component of the column here may result in multiple technical components as ASW, BSW, or/end RTE software components in AUTOSAR CP.			

The application component "SDN controller & network manager" should be principally deployable on any in-vehicle SDV platform node because the "SDN controller & network manager" entity represents basically a nomadic function. Nomadic network functions don't provide network topological properties, hence are fundamentally deployable on any in-vehicle network topological node. However, nomadicity and deployability are restricted and constraint in SDN-based SDVs. The set of restriction is defined by the entire set of Non-Functional Requirements (NFR), commercial, economical, and technical system constraints.

Fig. 12 provides an exemplification with AUTOSAR CP as the host of the SDN application component:

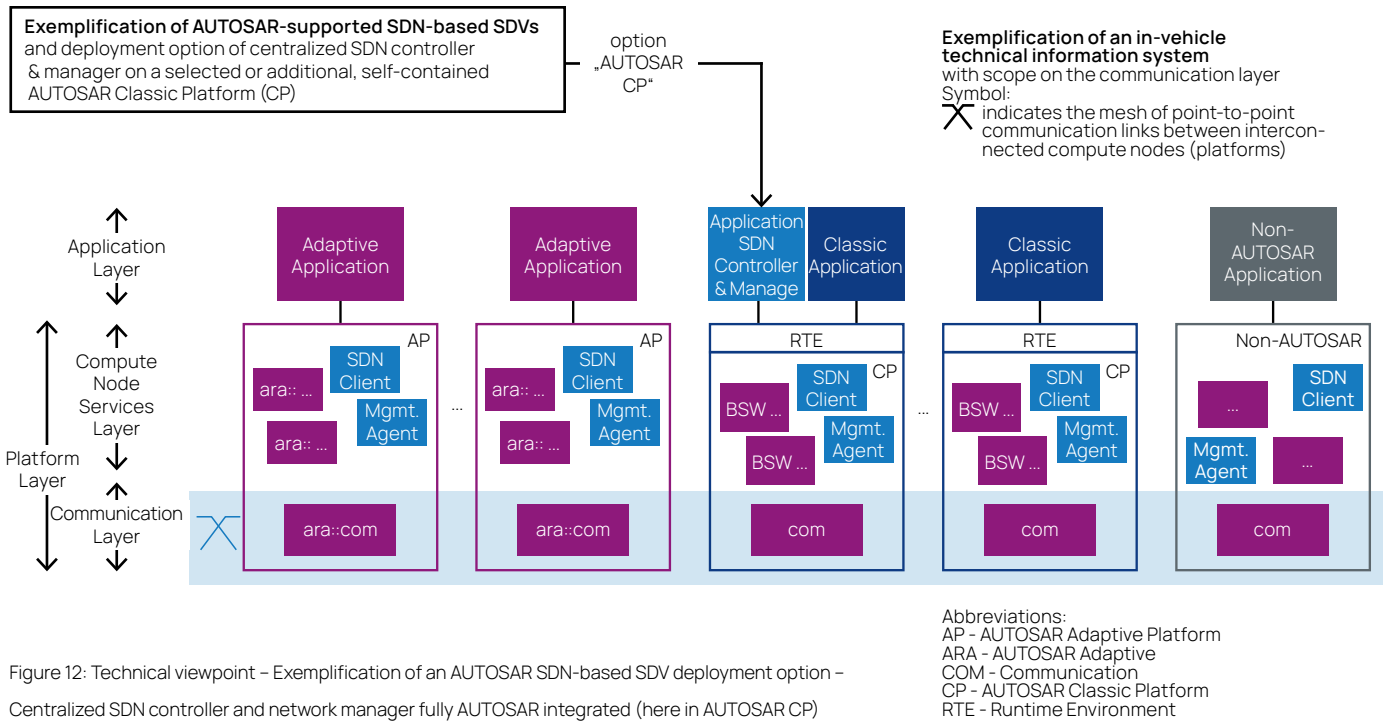


Figure 12: Technical viewpoint – Exemplification of an AUTOSAR SDN-based SDV deployment option – Centralized SDN controller and network manager fully AUTOSAR integrated (here in AUTOSAR CP)

There is the assumption here that all non-functional requirements (NFR) and constraints could be satisfied by either

an integrated approach, i.e., an existing AUTOSAR CP build could additionally be used	or	an additional, self-contained AUTOSAR CP is used

The two other basic technical components must be also equipped:

SDN client on all platform nodes acting as endsystem, i.e., running application software instances of distributed application functions:	and	management agent on all platform nodes

It must be noted that this isn't a complete and comprehensive description of a technical AUTOSAR system. The outlined SDN-capable AUTOSAR model provides just the top-level framework. For instance, the discussion of control, operation, and management of peripheral nodes (like sensors, actuators, I/O, storage) associated to platform nodes is omitted here.[26]

8.1.1.3 Integration of management data models in AUTOSAR: YANG-to-ARXML

Data modeling for AUTOSAR software systems starts usually at the modeling language level of UML (Unified Modeling Language), which are then again transformed (at a lower system abstraction level) in programming context specific data models like ARXML (or FIBEX, etc). Another AUTOSAR architectural paradigm is the full integration of the communication system in the AUTOSAR ecosystem, leading to the consequence that management data models for communication service and network resources are defined in ARXML.

A future AUTOSAR SDN-capable software system is expected in fully integrate IEEE- and IETF-defined YANG data models. Such an integration step requires a so-called M2M (Model-to-Model) transformation function. See e.g., [37] for a YANG-to-ARXML based M2M requirements discussion.

26 Another example is the deployment and integration of the technical automotive SDN components in technical AUTOSAR platform software architectures. For instance, the management agent would be most likely closely positioned to the Operating System given the management data elements in scope and the relation to AUTOSAR diagnostic services [21].

8.1.2 Automotive SDN for JASPAR

We'd like to discuss a possible mapping of the functional SDN-based SDV architecture to JASPAR's automotive SDN. JASPAR provides already explicit specification how TSN shall be network and protocol profiled for in-vehicle communication systems (see [38, 39]). The control, operation, and management of that in-vehicle TSN system shall follow a dedicated SDN pattern. Automotive SDN requirements and a top-level architecture are published in [40, 41]. There are many automotive SDN use cases elaborated, some of them demanding dynamic SDN-based control activities for an SDV-under-Ops (see also [42]).

The introduced functional SDN-based SDV model (Fig. 10) and JASPARs SDN vision perfectly match to each other, see the mutual mapping depicted by Fig. 13:

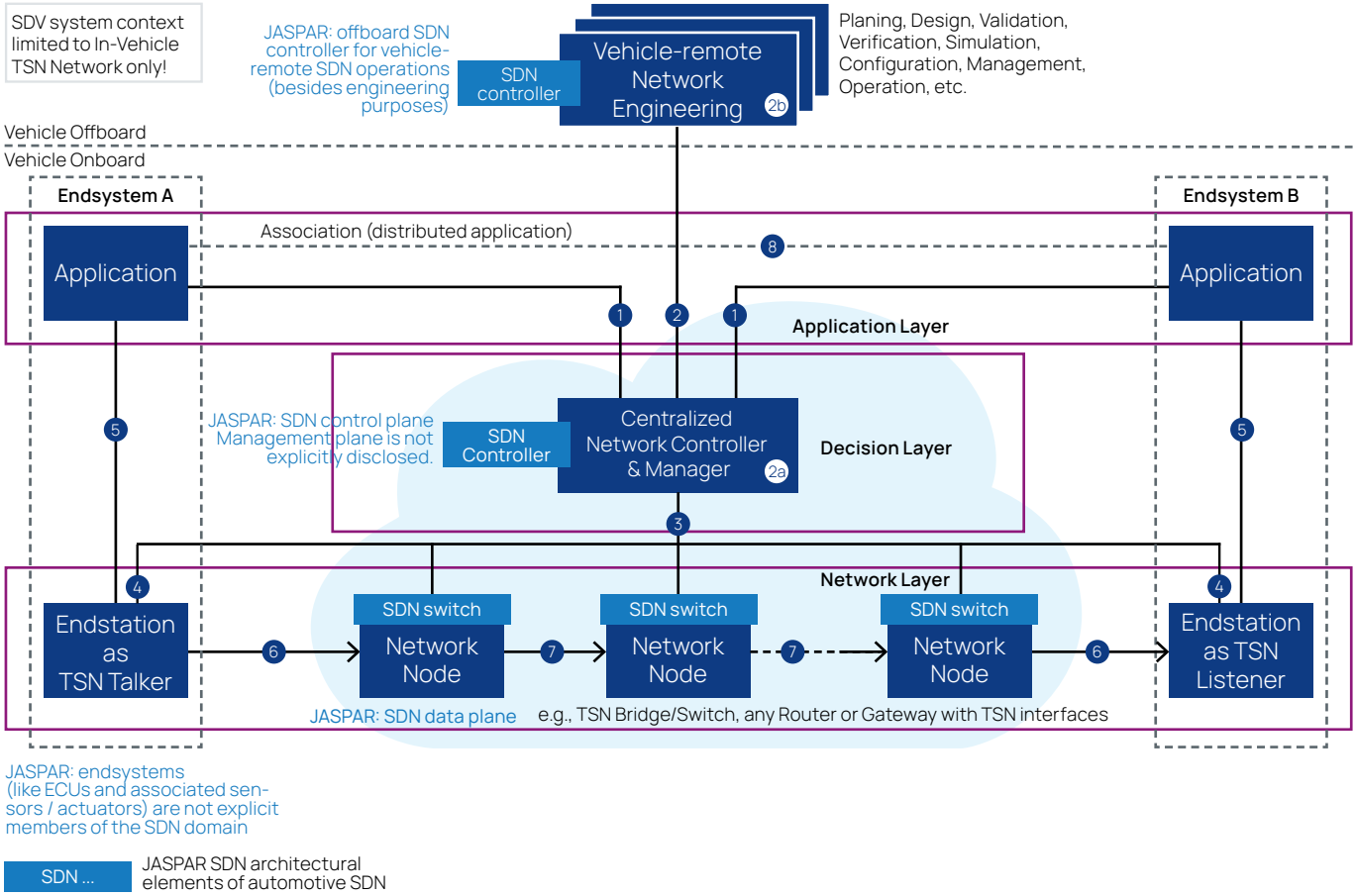


Figure 13: Mapping the functional SDN-based SDV model to the technically oriented JASPAR automotive SDN architecture. See page 27 for full legend

The SDN controller and SDN switch are the main automotive SDN elements in the JASPAR architecture. The automotive SDN specifications focusing on the in-vehicle control and user (data) planes. The management plane is omitted in the SDN model, also not yet explicitly outlined in the published TSN profile versions. Consequently, the management plane interconnection of endsystems (as technical Electronic, Domain, Telematic, etc Control Units (ECU, DCU, TCU)) is not yet outlined (interface 4).

Any more detailed discussion of the JASPAR envisioned and defined automotive SDN is beyond the scope of this whitepaper.

8.2 Appendix II – Fail-safe design and architecture for SDN-based SDVs

Many vehicle application functions are constraint by stringent functional safety objectives, such a applications for vehicle control, assisted or automated driving. There are similar functional safety related non-functional requirements (NFR) for the automotive SDN control and management plane functions, particularly when concentrating SDV control/management capabilities in fully centralized computing entities like the Centralized Network Controller & Manager.

The set of NFRs would typically demand at least for

High service availability of all the services running on that centralized node	High communication availability for the SBI and NBI	Fail-safe operation in case of unavailability of above functions

SDN architectures are generally suited for high quality operational, control and management objectives – that's state of the "SDN art" in carrier-grade public networks.

However, the engineering and discussion of fail-safe automotive SDN architectures would go beyond the scope of this whitepaper.

There is already in-vehicle networking related work concerning safety aspects published: e.g., JASPAR has defined functional-safety related in-vehicle TSN networking requirements and use cases inclusive failure modes for fault tolerant behavior [38]. Furthermore, JASPAR specified functional safety related in-vehicle SDN requirements where the fully centralized SDN architecture model shall be used to control and manage transitioning in fail-operational networking behavior [40].

Similar safety engineering would be also applied on the centralized in-vehicle SDN functions.

The network plane separated automotive SDN architecture implies already an inherent decoupling of functions. For instance, a prompt transitioning from in-service to out-of-service of some or all SDV control and management plane functions, such as the worst-case scenario of a complete outage of the Centralized Network Controller & Manager and its communication connectivity would not immediately impact the SDV user plane functions. Even that existing SDV services couldn't be changed for the moment, they would remain in their latest service configuration state and their operational state "in-service", so the SDV as such keeps going and remains still fully operational.

8.3 Appendix III – Transitioning to automotive SDN, - revolution or evolution?

8.3.1 Starting point

An introduction of SDN capabilities doesn't require necessarily a revolution of E/E architectures. An evolutionary transition towards more and more automotive SDN features needs to be feasible.

Revolution would mean a break of in-vehicle architectural principles as such. Evolution implies the continuous development (CD) and continuous integration (CI) of SDN feature increments in a backward, but also forward compatible manner.

Backward compatibility means that every new SDN feature is backward compatible with previously introduced SDN capabilities. Forward compatibility satisfies the architectural requirement that the applied automotive SDN framework and overall architecture model should be consistent, stable and invariable against every new SDN increment, orthogonal to evolutionary E/E structural changes.

8.3.2 Evolutionary, incremental introduction of automotive SDN

An E/E architectural evolution towards automotive SDN could be partitioned in many feasible incremental roadmaps. The following provides an example with for instance ten SDN increments. Every SDN level adds self-contained added value to the SDN-based SDV. Every SDN level increment comprises a set of automotive SDN use cases. Each use cases represents values in categories of business, services, service qualities, operational aspects, etc.

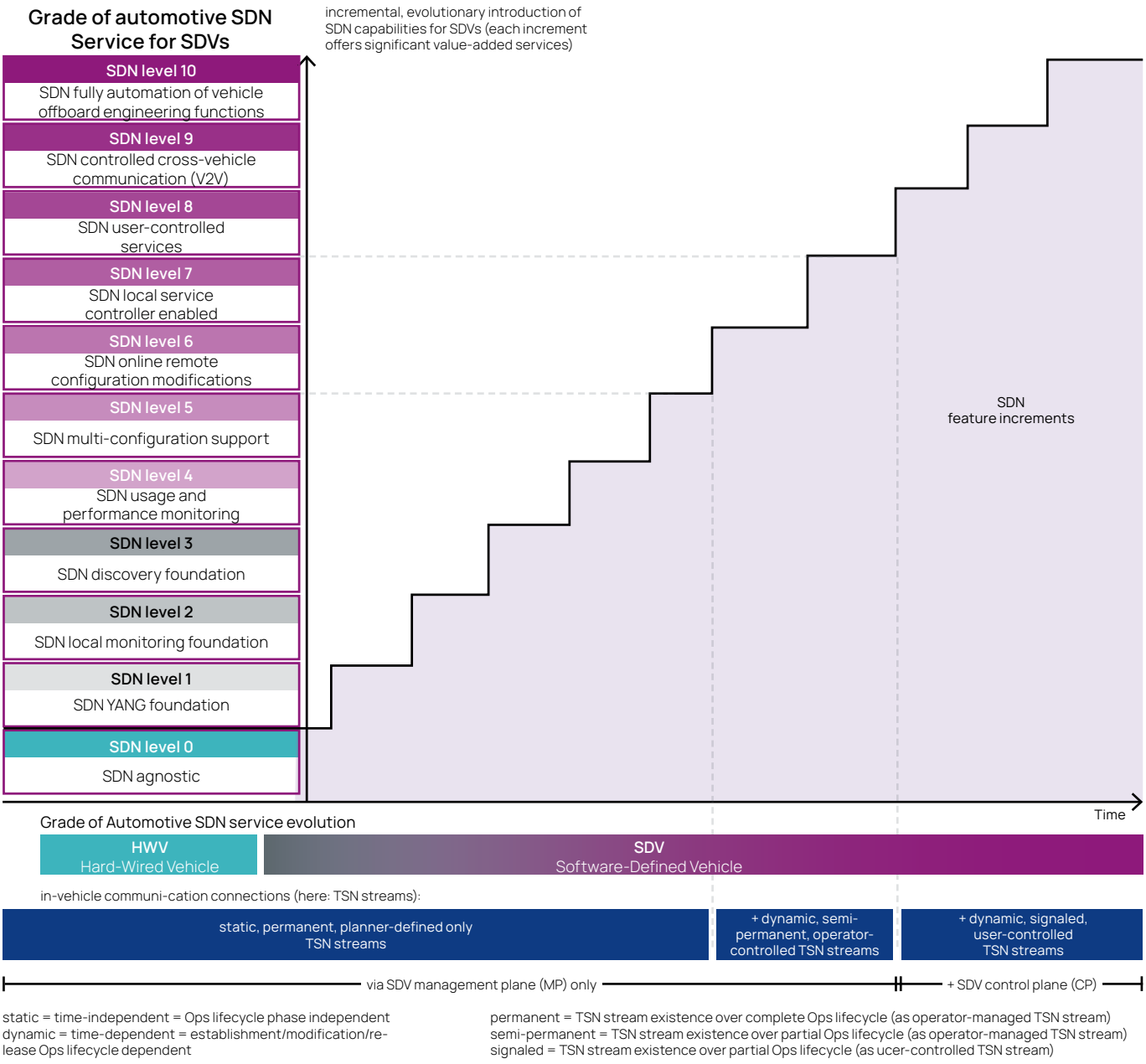


Figure 14: Evolutionary, incremental enhancements of the Grade of automotive SDN Service for SDVs (example of a roadmap partitioned in ten steps).

8.3.3 Description of automotive SDN levels

Each SDN level is characterized by a bundle of specific capabilities. The following Table 2[27] provides a rough description in the three dimensions of SDV control and management inclusive safety risk assessment, required onboard and offboard SDN functions and finally impact on major SDN interfaces:

27 Only readable in electronic document version. The PDF document version may be found under www.etas.com

SDN-based SDV: automotive SDN attributes							Function								Automotive SDN control and management plane interfaces			
							onboard (in-vehicle)				offboard (SDV engineering functions)							
							SDV management plane (MP)		SDV control plane (CP)		SDV management plane (MP)		SDV control plane (CP)		1 (NBI for ACI)	2 (NBI for remote online access)	3 (SBI in general)	4 (SBI for end-systems)
SDN feature increment	**Automotive SDN level**	**SDV control & management services**	**Safety-criticality: risk assessment**	**SDV configuration management**	**SDV supervision and data acquisition**	**SDV fault management and alarm reporting**	**Management Agent**	**centralized Management Manager**	**centralized SDN Service Controller**	**SDN client**	**Offline (Devs) Management Manager**	**Online (Ops), remote Management Manager**	**Offline (Devs) SDN Service**	**Online (Ops), remote Service Controller**	**Application Control Protocol**	**Management protocol NETCONF**	**Management protocol NETCONF or CORECONF**	**Management protocol NETCONF or CORECONF**
SDN agnostic	0	none (SDV = HWV)	nothing additional to HWV	Static vehicle Configuration-under-Devs only	no	no	no (SW image built-in configuration data)	no	no	no	yes, as part of integrated Dev engineering functions	no	no	no	not in use	not in use	not in use	not in use
SDN YANG foundation	1	introduction of YANG-based data models	nothing additional to HWV	Static vehicle Configuration-under-Devs only	no	no	yes incl. management datastore "startup" (for YANG configuration data)	no (i.e., the management agents running still self-contained, performing initial configuration setup only)	no	no	nothing in addition	no	no	no	not in use	not in use	not in use	not in use
SDN local monitoring foundation	2	introduction of centralized management manager for only vehicle supervision (but without V2X-based remote online reporting capabilities)	nothing additional to HWV (passiv monitoring and data acquisition should non-intrusive)	Static vehicle Configuration-under-Devs only	yes, Monitoring-under-Ops	no	yes incl. management datastores "startup", "running" (+"operational")	yes, incl. management datastores for YANG state data only)	no	no	nothing in addition	no	no	no	not in use	not in use	yes, in notification management mode only	yes, in notification management mode only
SDN discovery foundation	3	cross-check of actual versus planned configuration, which requires the discovery of devices, capabilities, services, topologies inclusive its operational state data	nothing additional to HWV (passiv monitoring and data acquisition should non-intrusive)	Static vehicle Configuration-under-Devs only	yes, Monitoring-under-Ops	yes	yes, incl. neighbour-to-neighbour discovery function	yes, incl. management application function for comparing actual versus intended behaviour incl the notification and logging, which implies to generation of manager-local topological YANG data models	no	no	nothing in addition	no	no	no	not in use	not in use	yes, in notification management mode only	yes, in notification management mode only
SDN usage and performance monitoring	4	extended SDV supervison services by dedicated resource usage and performance parameter monitoring and selected acquisition of correspondent state data	ditto	Static vehicle Configuration-under-Devs only	yes, Monitoring-under-Ops	yes	yes, extended set of YANG state data to be acquired	yes, incl. collection and aggregation of YANG data	no	no	nothing in addition	no	no	no	not in use	not in use	yes, in notification management mode only	yes, in notification management mode only
SDN multi-configuration support	5	SDV onboard support of multiple configuration options inclusive autonomous changeoves between SDV configurations (SDN fail-safe fallback to safety uncritical configurations) NOTE - See also JASPAR [...]	risk mitigation capabilities	Static vehicle Configuration-under-Devs only, but preparation of multiple configuration data sets	ditto	ditto	yes, additionally multiple local candidate datastores, one per configuration data set	yes, incl. the capability in managing conditionally determined switchovers between configurations	yes, in case of service-individual changeovers between configurations (the SDV service controller would be the decisive instance)	no	nothing in addition	no	no	no	not in use	not in use	yes in full operational mode (i.e., incl. configuration management)	yes in full operational mode (i.e., incl. configuration management)
SDN online remote configuration modifications	6	SDV admission control and network resource management performed offboard (thus, not yet delegated to the onboard entities), SDV operator-controlled TSN stream management only	risk subject of offboard functions only	Dynamic vehicle Configuration-under-Ops, but entire vehicle configuration setup prepared vehicle remotely	ditto	ditto	nothing in addition	nothing in addition	yes, dumb control capabilities, just forwarding of remotely prepared configurations to management manager	no	nothing in addition	yes	yes	yes	not in use	yes	nothing in addition	nothing in addition
SDN local service controller enabled	7	SDN online remote configuration requests to in-vehicle SDN service controller, which becomes now fully responsible now for admission control and network resource management	fail-safe operational behaviour required	Dynamic vehicle Configuration-under-Ops, now also vehicle locally initiated by SDN service controller (so-called operator-controlled TSN streams)	ditto	ditto	nothing in addition	nothing in addition	yes, now smart because the SDN service control intelligences in terms of network algorithmics for admission control decisions and network resource management	"yes" (optional, if NBI-ACI will be used in informing the end-point	nothing in addition	yes	yes	yes	not in use	yes	nothing in addition	nothing in addition
SDN user-controlled services	8	SDN-supported in-vehicle applications are allowed to request communication services, i.e., support of both, user- and operator-controlled TSN streams	fail-safe operational behaviour required	Dynamic vehicle Configuration-under-Ops, now also user controlled (by application endpoints) besides operator-controlled (by vehicle local or remote initiation)	ditto	ditto	nothing in addition	nothing in addition	yes, additional consideration of NBI application control	yes	nothing in addition	yes	yes	yes	yes	yes	nothing in addition	nothing in addition
SDN controlled cross-vehicle communication (V2V)	9	SDN-integrated inter-vehicle communication services using established ETSI-defined V2V communication services	fail-safe operational behaviour required	Dynamic V2V Configuration-under-Ops	ditto	ditto	nothing in addition	nothing in addition	yes, additional SDN-integrated V2V connection management	yes	nothing in addition	yes	yes	yes	yes	yes	nothing in addition	nothing in addition
SDN fully automation of vehicle offboard engineering functions	10	continuous replacement of still human activities by machines in varies degrees of automation	ditto	nothing in addition	ditto	ditto	nothing in addition	nothing in addition	no impact	yes	nothing in addition	yes (now fully automated)	yes (now fully automated)	yes (now fully automated)	yes	yes	nothing in addition	nothing in addition

Table 2: Description of incremental automotive SDN service levels

8.4 Glossary

Term	Definition
Automotive Ethernet	The automotive specific selection of native IEEE Ethernet building blocks around the physical and data link protocol layers, network operational protocols (such address resolutions, topology modifications, time synchronization), TSN-based QoS support functions, communication security measures, etc. All such SDV dedicated Ethernet features are motivated by automotive-specific network constraints, quality objectives or/and communication traffic patterns. Thus, it is IEEE compliant Ethernet with dedicated support for automotive networking requirements but not automotive proprietary.
Communication matrix	The topological structure of the entire set of communication services operated concurrently in a closed communication network (domain), abstracted as end-to-end connections. There are many matrix abstraction levels. A basic communication matrix might show application layer protocol connections, and the depiction of source and destination nodes only (i.e., without any interim network nodes).
Static versus dynamic communication matrix	Static: an unmodified communication topology during the entire SDV-und-Ops lifecycle phase. There are consequently only permanent TSN streams. However, traffic and/or quality related properties of existing TSN streams might be subject of change. Dynamic: new TSN streams might be established or/and existing TSN streams might be modified or released during SDV-under-Ops.
Configuration management	Represents a polysemous term in context of SDV information systems under DevOps. It covers at least: 1. workflow or build configuration management (Dev only); 2. structural configuration management (DevOps), with scope on topological and operational structures; 3. content configuration management (DevOps) by setting or modifying of object attribute values (without structural impact).
E/E architecture	Electrical and electronic architecture [5] A coupled, two-plane vehicular architecture, consisting of: 1) an electrical energy or power distribution network plane; and 2) an information processing and communication network architectural plane.
SDV network plane	SDV information planes: user, control, and management plane according to or derived from e.g., ITU-T Y.2011. The whitepaper will consider a dual-plane SDV architecture (with UP and MP) only.
SDV mobility state model	Management services for the SDV information system are typically conditional and constrained by mobility state of a vehicle. A mobility state does further detail the operational state of an SDV-under-Ops. The whitepaper uses following example: mobility states of motion, parking, charging, workshop maintenance. All motionless states might be further qualified by planned or ad-hoc, which impacts the knowledge about the available time window for management purposes.
Service Level Agreement (SLAg)	ITU-T E.860 [25]: A Service Level Agreement is a formal agreement between two or more entities that is reached after a negotiating activity with the scope to assess service characteristics, responsibilities and priorities of every part. The SLAg in SDN-based SDV relates to the agreement about network service qualities (like security, QoS, safety, availability) between the application function and the in-vehicle network communication services.

Term	Definition
Service Level Assurance (SLAs)	Assurance is the degree of confidence that the process or deliverable meets defined characteristics or objectives. The SLAs focuses on QoS objectives (e.g., ITU-T Y.3107 [43]), and belongs to the category of QoS assurance management services. The SLAs in SDN-based SDV relates to probe-based checks (e.g., by performance monitoring) of an assured SLAg and the actually delivered QoS.
Service Quality Agreement (SQA)	A SQA is the subset of a SLAg with scope on quality of communication service objectives only (e.g., ITU-T E.860 [26]). The SQA in SDN-based SDV relates to the QoS-specific agreement between the application function (as QoS user) and the in-vehicle network (as QoS provider).
TSN stream	A basic IEEE 802.1Q stream which is further qualified by a particular TSN usage profile. Such a network profile specification determines the relevant stream descriptor elements (i.e., the set of stream identification elements from the IEEE 802.1Q-2018, clause 35.2.2.8.3 DataFrameParameters" descriptor), the so-called TSN user plane stream identifier. NOTE 1 – The IEEE 802.1Q stream concept refines the IEEE 802.3 stream and emphasizes also the anchoring of such stream to the data link layer, i.e., Ethernet MAC frames. NOTE 2 – A TSN stream represents consequently topologically a unidirectional, connectionless, point-to-point or -multipoint end-to-end layer 2 connection (model). Any layer 2 connection is used as aggregate for upper protocol layer traffic. Supported traffic aggregates (associated to a TSN stream) are determined by the profiled stream identifier specification.
(SDV) TSN communication matrix	A communication matrix at the abstraction level of TSN streams. Such a matrix may only provide TSN stream source and destination endpoints (talker and listeners) due to the connectionless nature.[28] The representation of communication matrices are data models describing the topological and service structure of TSN streams, independent of static (time-independent) or dynamic (time-dependent) communication services. YANG topological data models are used for the description of TSN communication matrices at the vehicle onboard and offboard levels (see Fig. 9).[29] The offboard YANG TSN communication matrix is essentially a digital twin of the associated onboard YANG topological model. It has to be emphasized that all type of TSN streams (static, dynamic) are part of the same communication matrix.

28 Id est, we exclude the option of an enforced connection-oriented operation by e.g., managed FDBs (Filtering Database) to route all MAC frames of a particular TSN stream along the same end-to-end network path.

29 For experts: YANG-based TSN communication matrix data models would contain the stream descriptions of all active (enabled), inactive (disabled), planned TSN streams, using a stream data model according to [12, D2.2], clause 48.2.12: e.g., a reduced subtree of branch string of YANG module cnc-config.

8.5 Bibliography

[1] N. Anerousis, P. Chemouil, A. A. Lazar, N. Mihai and S. B. Weinstein, "The Origin and Evolution of Open Programmable Networks and SDN," in IEEE Communications Surveys & Tutorials, vol. 23, no. 3, pp. 1956-1971, 2021.

[2] D. Kreutz et al., "Software-Defined Networking: A Comprehensive Survey," Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, 2015.

[3] "The Software-Defined Vehicle: A Digital-First Approach to Creating Next-Generation Experiences", eBook (access via https://www.digital.auto/sdv-report), 2023.

[4] A. Lock, N. Tracey, and D. Zerfowski, "Entering New Worlds: New E/E Architectures with Vehicle Computers Offer New Opportunities," ETAS GmbH, Tech. Rep., 2020. https://www.etas.com/en/downloadcenter/35398-35399.php

[5] ITU-T Recommendation X.1381, "Security guidelines for Ethernet-based in vehicle networks" (2023)

[6] ETSI EN 302 665, "Intelligent transport systems (ITS); Communications architecture" (2010).

[7] M. Haeberle et al., "Softwarization of Automotive E/E Architectures: A Software-Defined Networking Approach", 2020 IEEE Vehicular Networking Conference (VNC), 2020.

[8] T. Hackel et al., "Software-Defined Networks Supporting Time-Sensitive In-Vehicular Communication", IEEE 89th Vehicular Technology Conference, 2019.

[9] N. Cardona et al., "Software-Defined Vehicular Networking: Opportunities and Challenges", IEEE Access, vol. 8, 2020.

[10] ETAS RealTimes 2019/2020, "New E/E architectures with vehicle computers offer new opportunities" (2020) Download: https://www.etas.com/en/downloadcenter/35398-35399.php#active RealTimes: https://www.etas.com/en/company/realtimes_2019_2020.php

[11] Bosch Whitepaper, "Die nächste Generation an E/E-Architekturen" (2023) Download: https://www.bosch-mobility.com/de/mobility-themen/ee-architektur/

[12] IEEE Std 802.1Q-2022, IEEE Standard for Local and metropolitan area networks — Bridges and Bridged Networks

[13] ITU-T Y.1291, An architectural framework for support of Quality of Service in packet networks.

[14] ITU-T Y.1222, Traffic control and congestion control in Ethernet-based networks.

[15] ITU-T Y.1563, Ethernet frame transfer and availability performance.

[16] MEF 10.3, Ethernet Services Attributes Phase 3.

[17] ITU-T Y.3302, Functional architecture of software-defined networking.

[18] ITU-T E.360.1, Framework for QoS routing and related traffic engineering methods for IP-, ATM-, and TDM-based multiservice networks.

[19] ITU-T E.529, Network dimensioning using end-to-end GOS objectives.

[20] IEEE Draft Standard for Local and metropolitan area networks — Time-Sensitive Networking Profile for Automotive In-Vehicle Ethernet Communications

[21] AUTOSAR, Specifications of Diagnostics (for Classic and Adaptive Platforms)

[22] ISO 14229, Road vehicles – Unified diagnostic services (UDS)

[23] ASAM, Specification of SOVD (Service-Oriented Vehicle Diagnostics)

[24] B. Misic, W. Hauck and A. Schwarz, "Software-Defined Vehicles under Management in continuous DevOps lifecycles: In-Vehicle Variant Management using NETCONF and YANG," Bosch/ETAS GmbH, Tech. Rep., 2023.

[25] ITU-T E.860, Framework of a Service Level Agreement.

[26] ITU-T E.801, Framework for Service Quality Agreement.

[27] Draft IETF draft-jeong-opsawg-intent-based-sdv-framework, "An Intent-Based Management Framework for Software-Defined Vehicles in Intelligent Transportation Systems" (2024, work in progress)

[28] IEEE Standard for Local and Metropolitan Area Networks - Bridges and Bridged Networks Amendment 38: Configuration Enhancements for Time-Sensitive Networking
https://ieeexplore.ieee.org/document/10542670

[29] IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic

[30] IEEE Standard for Local and metropolitan area networks — Quality of Service Provision by Network Systems

[31] ONF TR-521, SDN Architecture 1.1.

[32] IETF RFC 7426, Software-Defined Networking (SDN).

[33] ITU-T Y.3300, Framework of software-defined networking.

[34] ITU-T Y.2111, Resource and admission control functions in next generation networks.

[35] ITU-T Y.2113, Ethernet QoS control for next generation networks.

[36] ETSI GS NFV-EVE 005, Network Functions Virtualisation (NFV); Report on SDN Usage in NFV Architectural Framework.

[37] Architectural thoughts about management technologies NETCONF and YANG for AUTOSAR-based Software-defined Vehicles, Albrecht Schwarz, 2023 IEEE SA Ethernet & IP @ Automotive Technology Day (E&IP@ATD).

[38] JASPAR ST-HN-8, TSN Profiles for In-Vehicle Ethernet, Ver.3.0 (2022)

[39] JASPAR ST-HN-14, TSN Research Report, Ver.1.0 (Draft 2024)

[40] JASPAR ST-HN-11, JASPAR In-vehicle SDN Function Requirements Definition, Ver.1.0 (2023)

[41] JASPAR ST-HN-13, Architecture Definition of Automotive SDN, Ver.1.0 (Draft 2024)

[42] Proposal of Dynamically Configurable In-Vehicle Network as an Enabler of Software Defined Vehicle, Takumi Nomura et al., 2022 IEEE SA Ethernet & IP @ Automotive Technology Day (E&IP@ATD).

[43] ITU-T Y.3107, Functional architecture for QoS assurance management in the IMT-2020 network.

About ETAS

Founded in 1994, ETAS GmbH is a wholly owned subsidiary of Robert Bosch GmbH with a local presence in all major automotive markets in Europe, North and South America, and Asia.

ETAS offers comprehensive solutions for the realization of software-defined vehicles in the areas of software development solutions, vehicle operating system, data acquisition and processing solutions, integrated customer solutions and cybersecurity.

As industry pioneers in cybersecurity, we assist our customers in managing cybersecurity-related complexities, reducing cyber risks, and maximizing their business potentials with a proven on- and offboard portfolio of software products and professional security services.

ETAS automotive security solutions are safeguarding millions of vehicle systems around the world – and are setting standards for the cybersecurity of software-defined vehicles.

Contact

ETAS
Stuttgart (Germany)

Dr. Albrecht Schwarz
albrecht.schwarz@etas.com

Florian Wagner
florian.wagner3@etas.com

BOSCH
Stuttgart (Germany)

Jens Bierschenk
Jens.Bierschenk@de.bosch.com

TTTech
Vienna (Austria)

Dipl.-Ing. Georg Stöger
georg.stoeger@tttech.com

Dipl.-Ing. Jürgen Wohlmuth
juergen.wohlmuth@tttech.com