Making the Case for Centralized Automotive E/E Architectures

Victor Bandur, Gehan Selim, Vera Pantelic, and Mark Lawford, *Senior Member, IEEE*

***Abstract*—The rapidly increasing complexity of software in modern cars dictates new trends in electrical and/or electronic (E/E) automotive architectures. As a result, many original equipment manufacturers (OEMs) and suppliers have been advocating centralized E/E architectures as the automotive architectures of the future. In this article we make the case for centralized E/E architectures in the automotive industry. We discuss the motivation for centralized architectural schemes by carefully examining challenges and drawbacks of the traditional decentralized automotive E/E architectures, while contrasting with the corresponding benefits offered by centralization. Then, the technologies required to support new centralized architectures are discussed in detail. In particular, we present the state of the art in networking technologies, virtualization, electronic control unit (ECU) hardware and AUTOSAR, and discuss the state of adoption of these technologies in industry. Throughout, functional safety is considered and addressed as an overarching concern in the automotive industry.**

***Index Terms*—Automotive E/E architecture, centralization, domain control, functional safety, ISO 26 262.**

I. Introduction

THE RANGE and complexity of functions required in modern vehicles have been increasing rapidly, driven by industry trends such as automated driving, connectivity, electrification, and smart mobility. Additionally, incentives toward compliance with automotive standards (*e.g.*, AUTOSAR and ISO 26 262) make it increasingly challenging for traditional automotive *decentralized* electrical and/or electronic (E/E) architectures to cope with current and future needs of the industry. In a decentralized E/E architecture, embedded vehicular functions are distributed amongst a large number of interconnected electronic control units (ECUs), where each ECU is capable of processing its own data and communicating it to other ECUs to achieve advanced vehicular functionalities.

While long-standing decentralized architectures have several advantages [1], [2], they also suffer from severe drawbacks [1]–[4], most notably centered around scalability and communications performance. Traditional decentralized architectures mostly have a one-to-one mapping between vehicular functions and ECUs, resulting in an increasing number of cars having more than 100 ECUs [5] executing around 150 million lines of code [6]. In the case where one function is implemented by several co-ordinated ECUs, the communications load placed on the vehicle network increases. The practice of adding individual ECUs for individual functions leads to a significant cost increase due to large numbers of ECUs and bulky wiring harnesses. This practice furthermore results in increased software complexity and a large number of software variants for automotive ECUs. Thus, automotive development is facing significant development and maintenance costs, as well as missed milestones.

To address the limitations of decentralized E/E architectures, automotive E/E architectures have recently begun evolving toward centralized alternatives, such as *domain-centralized* (or *domain-oriented*), *cross-domain centralized* (or *cross-domain oriented*) and *vehicle-centralized* (or *zone-oriented*) architectures. Domain-oriented, cross-domain and zone-oriented architectures will be referred to here as *centralized architectures*. The basic idea of the centralized architectures is to *centralize* the processing of functions at the level of individual domains within a vehicle, groups of domains, or the entire vehicle. This evolution was influenced by a successful transition in the aviation industry from federated (*i.e.*, decentralized) to integrated (*i.e.*, centralized) architectures [2].

Several technologies are required to facilitate the operation of modern, centralized E/E architectures [3], [4], [6]–[8]. ECUs with enhanced processing power and hardware provisions for functional safety and security are needed to support increasingly complex vehicular functions. Further, improved communication networks, with higher bandwidth, real-time and traffic partitioning capabilities, fault tolerance mechanisms, advanced gateways (*i.e.*, gateways that aim to decrease latency and increase throughput using technologies such as hardware-based acceleration), and enhanced security measures are needed to support requirements of increasingly intelligent transportation systems.

While some automotive OEMs (*e.g.*, BMW, Volkswagen and Jaguar Land Rover) and automotive suppliers (*e.g.*, Delphi, NXP and Bosch) have already commercialized centralized solutions and advocated for centralization as the E/E architectural trend of the future, other automotive companies are still in the early phases of experimentation. In this article, we make the case for centralization as the E/E architectural pattern of the vehicle of the future that will address the aforementioned scalability issues that traditional, decentralized architectures are experiencing.

Manuscript received April 3, 2020; revised September 25, 2020 and November 17, 2020; accepted December 27, 2020. Date of publication January 27, 2021; date of current version March 10, 2021. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), Ontario Centres of Excellence (OCE), FCA US LLC, and FCA Canada Inc. The review of this article was coordinated by Dr. M. S. Ahmed. *(Corresponding authors: Victor Bandur; Vera Pantelic.)*

The authors are with the McMaster Centre for Software Certification, Department of Computing and Software, McMaster University, Hamilton, ON L8S 4L7, Canada (e-mail: bandurvp@mcmaster.ca; gehan.mustafa.kamel@gmail.com; pantelv@mcmaster.ca; lawford@mcmaster.ca).

Digital Object Identifier 10.1109/TVT.2021.3054934

We carefully examine key enabling technologies of centralized architectures to demonstrate that there is sufficient technological critical mass to make centralization a viable solution across various domains and applications within E/E architectures. Throughout the article, we reflect upon the implications of centralized architectures and their enabling technologies on the functional safety of vehicles, a major concern in the automotive industry [9].

The rest of this article is organized as follows. Section II motivates the need for centralization and gives an overview of basic concepts in modern E/E architectures as well as their adoption in the automotive industry. Section III introduces the basic functional safety considerations and concepts of the ISO 26 262 functional safety standard which are then used throughout this article to discuss the implications of centralization on functional safety. Section IV details key enabling technologies for centralized E/E architectures. Section V discusses the take-away message of the article, while pointing out possible challenges when centralizing E/E architectures. Section VI concludes the study.

II. The State of E/E Architectures: Motivation, Basic Concepts, and Challenges

This section presents motivation for centralized automotive E/E architectures. Then, basic concepts in the architectures are introduced. Finally, adoption of centralized E/E solutions in the automotive industry is discussed.

A. The Need

A vehicular E/E architecture includes ECUs, sensors, actuators, all their interconnections, as well as the electrical network (*e.g.*, high-voltage network, power electronics) and infotainment systems such as consumer electronics components [10]. Vehicular E/E architectures have traditionally been decentralized.

Decentralized vehicular E/E architectures have several advantages [1], [2]. For instance, decentralized architectures inherently support separation of concerns between ECUs: they allow for one or a few very tightly related functions to be deployed on a single ECU. This makes the verification of conditions for integration of ECUs into a network relatively simple, mainly ensuring the availability of the required bus bandwidth and checking simple timing requirements of communication messages [2]. Additionally, replacing damaged ECUs is straightforward, since ECUs are lightweight and do not implement many functions. For each ECU, CPUs with average processing power can be used since each ECU encapsulates limited functionality. However, decentralized architectures have several limitations [1]–[4].

First, the "one function per ECU" approach has proven infeasible in the time of the rapid and steady rise of software-implemented functionalities in modern cars. Adding individual ECUs for individual functions results in cost increases due to the additional ECUs and their wiring harness stubs. The increased wiring harness length further introduces an unwanted side effect, namely, limited freedom in the physical layout of the architecture and hence tighter constraints in optimizing the final architecture. Further, traditional E/E architectures mostly use CAN (Controller Area Network) networking, but also rely on FlexRay, LIN (Local Interconnect Network) and MOST (Media Oriented System Transport). While these networking technologies have proven sufficient for traditional communication, they are expected to be unsuitable in the near future due to the expected volume of data and number of interconnections between different ECUs. However, automotive Ethernet, albeit not considered a traditional automotive communication technology, has seen a steady rise in use in automotive networks in the past decade. Automotive Ethernet has the capability to support large volumes of data, as well as time-critical communication with functional safety requirements. It is our opinion, shared by many other authors (for example Matheus and Königseder [11]), that automotive Ethernet is the communication technology of future automotive networks, albeit not the sole communication technology to be employed in future cars. The use of automotive Ethernet will be one of the major drivers of the trend of consolidation toward centralized architectures [11].

Building vehicle variants for different market segments today implies a large number of software releases, and might necessitate building several architectures to support the diverse variant features. Therefore, diversity in the current market requirements implies the need for a versatile, as well as cost-effective, E/E architectural solution. This solution needs to handle very large numbers of vehicle variants resulting from customer expectations ranging from, for example, personalization of comfort functions, to reliance upon automated support such as that provided by automated driver assistance systems (ADAS). This increases the number of vehicle variants on offer. Further, automotive product lines today support various markets, technologies, and configurations (for example, different powertrain architectures, different hardware platforms, and components from different suppliers). E/E centralization bears large potential to decrease the number of software releases needed to support large automotive software product lines as will be discussed in the next section.

Also, current E/E architectural solutions were not necessarily designed to be robust to external malicious attacks [3]. Given the increase in vehicle connectivity to the outside world, such security features need to be guaranteed in future vehicular architectures.

Along with these drawbacks, the strategies adopted by OEMs and suppliers also affect trends in the evolution of E/E architectures [3], [12]. For example, besides the need for OEMs and suppliers to be compatible with automotive standards (*e.g.*, AUTOSAR, ISO 26 262), OEMs rely on software and hardware provided by an increasing number of suppliers. They are compelled by demand to integrate improved technologies, such as advanced high definition displays, fast networking and powerful new ECUs performing high-end functions. But current architectures are not flexible enough to allow OEMs to easily integrate new, diverse technologies.

The aforementioned drivers prompted the automotive industry to investigate more centralized solutions for automotive E/E architectures, such as the domain-centralized, cross-domain centralized and vehicle-centralized (or zone-oriented) architectures introduced earlier. Motivation for the industry's shift toward centralized E/E architectures as presented in this section can be further framed by some fundamental examples and results, presented in the following section.

B. Motivating Analyses

1) Variant Handling: Cost-efficient handling of large automotive product lines is a major motivator for the industry to find new E/E architectural solutions. For example, the software for an engine controller of a large automotive OEM supporting three different engines generates 144 software/calibrations releases in a decentralized E/E architecture. In a centralized, domain-oriented architecture, with a powerful (domain) controller hosting the major part of the controller's functionality, and a significantly simpler smart actuator capturing hardware variability only (differences between different engines and the differences between underlying hardware platforms), the number of releases would reduce to 72 for the domain controller, and three for the smart actuator, for a combined total of 75.

As another example of centralized architectures' support for cost-effective development of automotive software product lines, let us consider the case of a vehicle system requirement that varies across geographic regions. In the US, a vehicle is required not to roll away when keyed off, through regulation which states, "*...the starting system must prevent key removal ...unless the transmission or gear selection control is locked in 'Park'*" [13]. Regulations in a country like Indonesia might have a very different requirement, that "*the vehicle can be keyed off and then put into Neutral, provided the driver is notified and explicitly accepts the keyed off transmission state*." This regional requirement allows the vehicle to be rolled when keyed off in automated parking garages or where parking space is at a premium.

In a decentralized architecture of a plug-in hybrid electric vehicle, this simple requirements change could necessitate software modifications to a significant number of ECUs:

1) the Body Control Module (BCM), which monitors the key and power ON/OFF user interface and updates the transmission position display, including accepting user acknowledgement of special gear selection modes;
2) the Transmission Control Module (TCM), which interfaces with the shifter and controls the transmission while possibly interfacing to a separate ePark ECU that controls the park pawl that locks the drivetrain in place;
3) the Hybrid Supervisory Controller (HSC), which coordinates the overall control of the hybrid electric drive train;
4) the Battery Management System (BMS), the battery controller, which, in the US version, might only allow charging while the park pawl is engaged.

All of these components might require an updated software release when the Indonesian vehicle variant is produced. For example, the user interface to the BCM must be modified to allow the car to be keyed off in Neutral, provide an alert to the driver and await explicit driver acknowledgement. In order to allow keying off in Neutral, the HSC and TCM would have to be appropriately modified to account for the requirement change. Further, the BMS in the Indonesian variant might implement the derived requirement that *"If a car is being charged when not in Park, the parking brake is requested while plugged in,"* as the US BMS variant would not allow charging outside of Park for safety reasons.

Overall, the number of affected ECUs is relatively high. In a decentralized architecture, the previously mentioned modules are typically located on separate ECUs. With the centralization of the powertrain domain, for example, the TCM, Motor Control Modules (MCMs), BMS, and HSC would have their variable functionality consolidated on one ECU: all the required changes to these modules in the decentralized architecture would now be focused on the one ECU, resulting in a single software release capturing the changes to these four modules. With further centralization by, for example, consolidation of chassis and HMI (Human Machine Interface) in the BCM, and all detailed powertrain requirements in the HSC, with the TCM, BMS, and MCMs functioning as smart actuators, only the BCM for the user interface and HSC for the rest of the functionality would need to be updated. All of the smart actuator code could remain unchanged. Thus, rather than requiring the four or five ECU software releases of the decentralized architecture, the domain controller architecture would only require two releases.

2) A Study: As a further example, an early study by Kanajan *et al.* [14] compares four different E/E architectures, ranging over the spectrum of centralization of both control and I/O. In their work, centralized control is the same notion as discussed in this article, whereas centralized I/O refers to whether a given sensor or actuator is connected directly to the controller ("centralized"), or to the communications network as a smart actuator ("decentralized"). So far in this article we have focused entirely on smart actuators, so "decentralized" in their terminology.

The four architectures considered by Kanajan *et al.* are named CICC (centralized I/O, centralized control), CIDC (centralized I/O, decentralized control), DIDC (decentralized I/O, decentralized control) and MDICC (mixed decentralized I/O, centralized control). Here we are only interested in DIDC and MDICC. DIDC is composed of many individual-function ECUs on a CAN bus, together with only smart sensors/actuators connected directly to the same CAN bus. MDICC is composed of a mix of centralized and decentralized ECUs and sensors/actuators, also with a CAN interconnect. The authors compare the four architectures in terms of a number of desirable architectural metrics: end-to-end control latency; physical attributes, such as wire length and number of ECUs; data volume on the network; architectural flexibility (robustness) to changes, such as relocating, centralizing or decentralizing sensors and actuators in the vehicle, and adding or removing ECUs (architectural changes related to vehicle variants).

The comparison is representative of various options in mixing ECUs, software control and sensors/actuators on a CAN bus, to form an E/E network. MDICC most closely represents a typical traditional, decentralized E/E architecture. However, a centralized architecture as we describe closest resembles DIDC, but with all control centralized onto a single, adequately capable ECU. We can say with confidence that, in the authors' original DIDC architecture, centralizing control onto a single powerful ECU has the following effects with respect to their four criteria.

First, control latencies can only decrease for two reasons. The first reason is that for control actions that require co-ordination between individual ECUs, the co-ordination now takes place on a single ECU with virtually zero latency by comparison. The second reason is that if all ECUs are consolidated, then the number of different messages on the CAN bus which can contend for bus access is reduced, as all these messages are now emitted by one single ECU. Reduced bus contention leads to decreased control latency. *Second*, the total wire length is reduced due to the absence of the wire stubs of each of the now-consolidated ECUs. *Third*, *in the worst case* that the ECUs do not communicate with each other, the total data volume on the bus remains unchanged. However, this is an unrealistic scenario, and we expect total data volume to decrease. *Fourth*, robustness to design changes in terms of cost increases because additional control functionality can be added as software to the one central ECU, whereas removal/insertion/relocation of sensors and actuators requires no architectural changes.

Now we revisit the authors' results in light of this modification to DIDC. *First*, for sensors and actuators directly connected to their corresponding controller, MDICC has superior latency, both in the original study, as well as compared to the modified version of DIDC. However, for smart sensors and actuators on the bus, control latency can be expected to decrease in DIDC with centralized control, for the reasons mentioned earlier. *Second*, the authors' original version of DIDC is only inferior to MDICC in terms of total number of ECUs, but in the modified version of the architecture, with one single master ECU, this is no longer the case. Admittedly, the overall ECU cost difference is a matter of separate investigation. *Third*, as expected, the fully distributed architecture of DIDC in the original study places all signals on the CAN bus and therefore uses the bus more extensively than MDICC. With complete centralization of control onto a single ECU, bus utilization is expected to decrease, as some of the signals will become internal to the central ECU, but it is impossible to state how the modified version of DIDC compares with MDICC in this regard. *Fourth*, in the original study, DIDC is superior to MDICC in terms of architectural flexibility. With all control centralized onto a single ECU, this does not change.

As can be seen, the study of Kanajan *et al.* can be used to understand how a centralized E/E architecture can compare to a classical decentralized architecture along a number of important dimensions. Indeed, assuming a change to the underlying CAN technology to the more modern CAN FD or automotive Ethernet, the latter of which provides orders of magnitude increases in data rate and would positively impact DIDC in reason three above, the authors' results cast centralization in an overwhelmingly favourable light.

3) Lessons From the Aerospace Industry: The aerospace industry has already successfully transitioned from decentralized ("federated" in the aerospace terminology) E/E architectures to centralized ("integrated") architectures. Modern aircraft leverage the Integrated Modular Avionics (IMA) concept, an integrated (centralized) architecture for avionics [15]. IMA is widely accepted as a cost-effective solution for increased avionics complexity with the following benefits: reduction in the number of CPUs, communication channels, I/O modules, and,

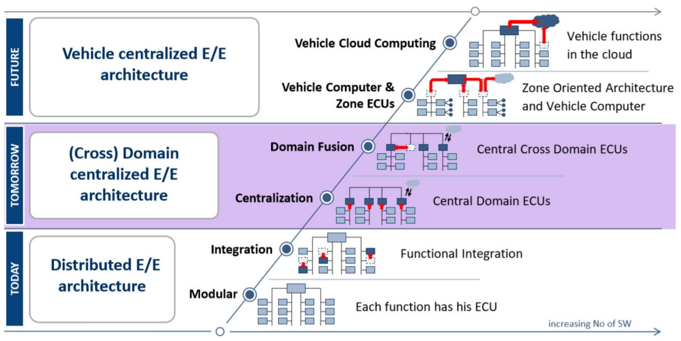
FUTURE
Vehicle centralized E/E architecture
Vehicle Cloud Computing
Vehicle functions in the cloud
Vehicle Computer & Zone ECUs
Zone Oriented Architecture and Vehicle Computer
TOMORROW
(Cross) Domain centralized E/E architecture
Domain Fusion
Central Cross Domain ECUs
Centralization
Central Domain ECUs
TODAY
Distributed E/E architecture
Integration
Functional Integration
Modular
Each function has his ECU
increasing No of SW

Fig. 1. Possible evolution of vehicular E/E architectures [3].

accordingly, size, weight and power consumption of electronics; improvement in modularity, and software and hardware reuse; increase in the efficiency of resources *etc*. [15], [16]. In particular, Boeing's 787 implementation of the IMA concept allowed Boeing to reduce the number of LRUs (Line-Replaceable Units) by more than 100, cutting down approximately 900 kg of the aircraft's weight [15]. Similarly, Airbus A380 cut in half the number of processor units by using the IMA concept [15]. Reductions in power consumption, space, and weight have been reported in the study of 35 projects (both commercial and fighter aircraft) [17]. Further, Mairaj reported on reductions in total number of on-board electronic systems, and improvements in reliability (Mean Time Between Failures) and resource utilization [17]. In the automotive industry only a few similar results have been published. For example, Delphi reported that the centralization of ADAS features on Audi A3 achieved a 30% savings in wiring harness mass and cost [5], while Bosch reported on a 15–20% reduction in wiring harness weight when going from one centralized architecture (domain-oriented) to a more centralized architecture (zone-oriented architecture) [7]. But we can generalize, by observing that automotive E/E architectures are similar to avionic ones in nature, only far less complex. Consider that, as stated earlier, a high-end automobile has over 100 ECUs, whereas the same number is encountered above as *reductions* in the number of electronic systems with the adoption of IMA by Boeing—this comparison provides a sense of scale. It is, therefore, not unreasonable to expect that benefits similar to IMA's will be observed in the automotive domain with the transition to centralization.

C. Basic Concepts

Several modern architectures based on the general idea of centralization have been gradually replacing decentralized architectures in an attempt to improve the scalability of E/E architectures. Fig. 1 by Bosch [3] demonstrates the history and envisioned future of E/E architectures. Automotive OEMs are currently experimenting with the three types of centralized architecture described above, domain-, cross-domain and zone-oriented, all based on the idea of integrating several related functions onto a single, powerful ECU. In this section we present the basic concepts behind these architectures.

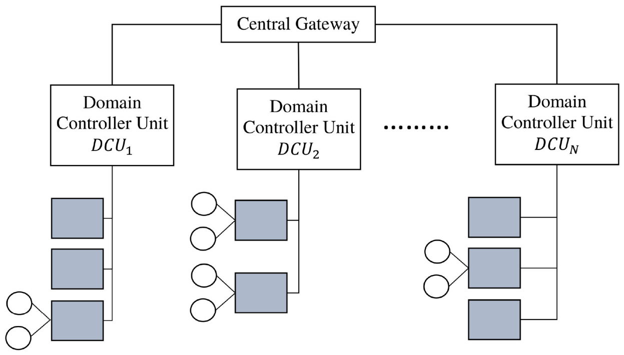
Central Gateway
Domain Controller Unit DCU_1
Domain Controller Unit DCU_2
Domain Controller Unit DCU_N

Fig. 2. Typical domain-oriented E/E architecture.

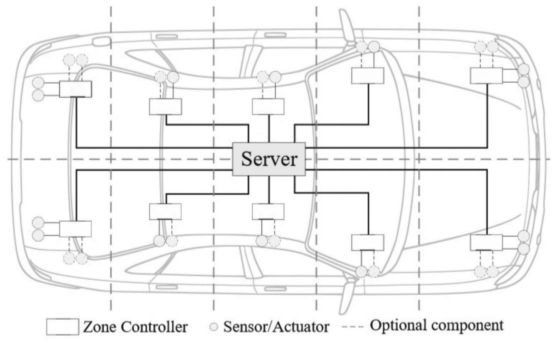

Fig. 3. Typical zone-oriented E/E architecture [18].

In a *domain-oriented* architecture [1], [3], [18]–[21], software components (SWCs) are clustered according to vehicle function domain, as shown in Fig. 2. Although vehicular domains differ slightly between organizations, four main domains are widely agreed upon in the automotive industry: Body and Cabin (comfort and lighting), Infotainment (displays, entertainment and information systems), Vehicle Motion and Safety (chassis, active/passive safety and driver assistance functions), and Powertrain (propulsion and exhaust gas treatment) [21]. In a domain-oriented architecture, each domain has a corresponding *domain cluster*, composed of one *domain control unit (DCU)* and zero or more *sub-domain ECUs*. The DCU controls its sub-domain ECUs, which are connected to the DCU only. A typical DCU has a powerful, multi-core CPU and hosts major functionalities of the domain, while offering an additional abstraction level that simplifies the interface between the domain and other domains. Sub-domain ECUs encapsulate auxiliary domain functions and require less powerful CPUs. Typically, together with their corresponding vehicle hardware, they form *smart actuators*. Different DCUs are interconnected with a high speed connection, *e.g.*, an Ethernet data spine. Sub-domain ECUs are connected to their DCU using typical field buses (*e.g.*, CAN, LIN, FlexRay), but also using automotive Ethernet, if the capability is required.

While domain-oriented architectures address many of the issues of decentralized architectures, they face two main challenges [12]. First, certain high-end functions such as ADAS require heavy interaction among DCUs, making the boundaries between the DCUs unclear. Second, due to the increasing interaction among the DCUs, it is suspected that current communication bandwidths will not be sufficient for future E/E architectures, as more such high-end functions arise. Cross-domain centralized architectures address these issues.

In *cross-domain centralized* architectures, the functions of more than one domain are consolidated onto single ECUs called cross-domain ECUs (CDCUs), as shown in Fig. 1. Thus, a possible realization of a cross-domain centralized architecture looks similar to the architecture in Fig. 2, but with CDCUs replacing DCUs. For example, functions of the chassis and powertrain domains can be consolidated into a single vehicle motion control CDCU [3].

For domain and cross-domain centralized architectures, the processing of complex functions is carried out in the DCUs and the CDCUs, respectively. Thus, as the complexity of the functions increases, so does the local processing performed in these ECUs, and much larger bandwidths are required to transfer the processed and increasingly complex information between the ECUs [18]. Additionally, more wiring is needed as individual vehicular functions become more complex and more dependent on other functions. These issues can be resolved in a vehicle-centralized or zone-oriented architecture [18].

In a *zone-oriented* architecture, a car is divided into zones and each zone has a zone ECU (ZCU), *e.g.*, zone “front right” and zone “front left”. Similar to sub-domain ECUs in domain-oriented architectures, ZCUs are connected to sensors and actuators. ZCUs, however, differ from sub-domain ECUs in that they do not perform any processing to realize a vehicular function. Instead, they collect and forward information relevant to their respective zone to a much more powerful central server which does the necessary processing of complex functions. In this architecture, complex processing can also be offloaded to the cloud [3]. Fig. 3 demonstrates a realization of a zone-oriented architecture. This architecture can alleviate the wiring length problem mentioned above. As previously mentioned, Bosch reported on achieving a 15–20% reduction in wiring harness weight when using a zone-oriented architecture instead of a domain-oriented architecture [7]. To the best of our knowledge, no supplier or OEM has yet reported on adoption of a zone-oriented architecture augmented with cloud computing; this approach has only been discussed theoretically. Another major advantage of the zone-oriented architecture is that high-bandwidth sensors, such as visual or depth sensors, can be connected directly to the central server for data processing, eliminating a potentially costly consequence to functional safety when adhering to the ISO 26 262 standard.[1] Functional safety and ISO 26 262 are discussed in the next section.

The architectures considered in this article, centralized architectures, include domain-oriented, cross-domain oriented and zone-oriented architectures, as defined in this section.

[1]Briefly, the Automotive Safety Integrity Level (ASIL) is the level of criticality of a safety requirement. Criticality and implementation cost increase as the ASIL increases. Specifically here, the sensors in question may have an ASIL higher than that of the ZCU to which they belong according to the zone partitioning scheme chosen. Adherence to ISO 26 262 mandates that, if the sensors are connected to this ZCU instead of directly to the server, then the ASIL of the ZCU must be increased to that of the sensor with the highest ASIL, increasing the development cost of the software on that ECU. This is discussed in detail in Section III

D. Adoption in the Automotive Industry

Many Tier 1 suppliers have already released DCU solutions. These, in turn, have been integrated in cars. Visteon's domain controller implementation, SmartCore, was launched with Mercedes Benz in March 2018 [22]. The DCU implements an integrated digital cockpit that includes instrument cluster, infotainment, connectivity, heads-up display (HUD) and driver monitoring functions. Another automotive Tier 1 supplier, Aptiv, reported on releasing the Active Safety Domain Controller [23], integrated by BMW, as well as the Cockpit (Domain) Controller [24] integrated by Ferrari. Further, Aptiv developed a centralized controller for Audi's A8 [25] that implements an SAE J3016 level 3 automated driving system. Bosch has developed the driver assistance system domain controller (DASy) [26]. Also, numerous Tier 1 suppliers have offerings of powertrain DCUs ([27]–[29], to name a few). For example, Bosch developed their Vehicle Control Unit (VCU) [27] that can serve as a powertrain domain controller, as a communications interface, and can be used in ADAS applications. Bosch are considering several milestones after a domain-centralized E/E architecture, namely cross-domain and zone-oriented architectures (the latter possibly augmented with cloud computing) [7]. It is Bosch's vision that is captured in Fig. 1. Bosch so far have adopted a cross-domain oriented architecture and a zone-oriented architecture. As previously mentioned, Bosch stated that using a zone-oriented architecture achieved 15–20% reduction in wiring harness weight over the domain-oriented architecture. Delphi announced that the centralization of ADAS features on Audi A3 achieved a 30% savings in wiring harness mass and cost [5]. According to one report, the DCU market is expected to rapidly grow: the expected average annual growth rate of the number of produced DCUs for integrated cockpit and autonomous driving between 2019 and 2025 is 50.7% [30].

Several OEMs have reported on the adoption of centralized E/E architectures. BMW and Audi have been vocal about centralizing their E/E architectures in order to support current and future industry trends [31]. A Body Domain Controller has been present in some of BMW's vehicles since 2017 [32]. While reports on BMW's latest E/E architectures are scarce, BMW are clear that the future of their E/E architecture is centralization. In 2016, Volkswagen stated their plan to develop and deploy a full domain-oriented architecture, with five main domains: powertrain, chassis, safety, driver assistance/autonomy and cockpit [33]. At the time, the company reported on intentions to develop and deploy a full domain architecture from scratch. Since then, the company have developed the MEB platform [34], a modular car platform for electric cars (used in a number of models from Audi, SEAT, Škoda and Volkswagen) based on a centralized E/E architecture. Jaguar Land Rover have been using an Ethernet backbone for their E/E architecture since 2017 and have centralized the infotainment domain by having a DCU control all infotainment functionalities [35]. Further, in [35], Jaguar Land Rover reported on planning to use dynamic network configuration and a service-oriented architecture (SOA) to provide more flexibility in installing new features, to better handle vehicle variants, facilitate hardware and software reuse, and to support increases in bandwidth.

Overall, customer demand, volume and diversity of Tier 1 offerings, and OEM experimentation with very positive results, all under anticipation of full traffic automation in the future, are currently driving OEMs to adopt new approaches to their vehicle E/E architectures. This is necessary and seemingly inevitable in order to future-proof vehicle production for an age of unparalleled levels of information exchange, both at the micro- (in-vehicle) and macro- (regional traffic information exchange) scale.

III. The Role of Functional Safety

As vehicle functions increase in complexity, the potential for unsafe behavior due to faults in the E/E systems increases significantly. This compels OEMs to develop their vehicles increasingly in compliance with safety standards. Currently, the *de facto* functional safety standard for automotive E/E architectures is ISO 26 262 [9]. In this section, we present the main concepts and requirements of the standard. The remainder of the paper uses these concepts to discuss functional safety concerns in the context of centralized architectures.

ISO 26 262, first published in 2011 and revised in 2018, provides guidance on integrating safety engineering with the product development life cycle to achieve functional safety in the E/E systems of road vehicles. The central concept of the standard is a set of four Automotive Safety Integrity Levels (ASILs). These ultimately determine the rigor with which a given component is developed. ASILs are assigned to the top-level safety requirements (safety goals), based on their *criticality*. ASILs propagate through the design process to requirements at all levels, down to the safety-related software and hardware requirements. ASIL A is assigned to relatively low-criticality functions (*e.g.*, instrument cluster display), ASIL D to the most critical (*e.g.*, brake-by-wire).[2] Therefore, requirements with a higher ASIL are implemented subject to stricter guidelines in the standard than those with a lower ASIL. Components which interact and have the potential to affect each other in case of malfunction must all be developed to the highest ASIL found among them. Naturally, the cost of development of a given function increases as its ASIL increases.

To help lower development costs, the standard introduces the method of *ASIL decomposition*, through which a requirement with a specific ASIL can be decomposed into independent but redundant requirements with potentially lower ASILs. Patterns and rules for the decomposition are prescribed in the standard, along with the conditions under which a given decomposition is valid. ASIL decomposition relies on a concept that is central to the standard, that of *sufficient technical independence* between elements of the E/E architecture, including SWCs. This concept itself consists of *absence of common-cause failures*, *absence of cascading failures* and *freedom from interference* [36]. Fig. 4 shows an example of interference, where due to an error (*e.g.*, undiscovered software bug, particle strike, silicon failure) SWC 1 gains the ability to write to the memory partition assigned

[2] An additional level, QM, exists. It is applied to non-safety requirements—these requirements are subject to the development guidelines of the organization's quality management system only, and not those of the standard.

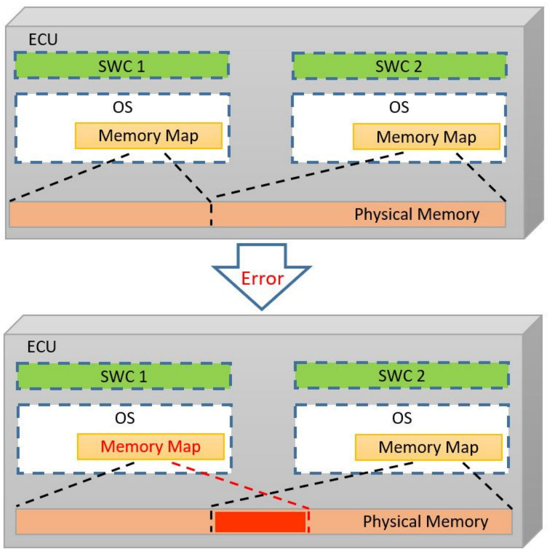

Fig. 4. Memory interference due to undiscovered error.

to SWC 2. ASIL decomposition is only valid when elements to which decomposed requirements are assigned are proven to be sufficiently independent with respect to common-cause and cascading failures.

The ASIL approach of the standard is structured in a way that creates an opportunity for vendors to commoditize high-criticality automotive components. The scheme that supports this ecosystem is the *Safety Element out of Context* (SEooC) [37]. SEooCs are generic components that are developed according to the standard, but which are sold as Commercial off-the-Shelf (COTS) components by automotive suppliers. The most common examples are CPUs. SEooCs are developed under certain assumptions made on their use. When OEMs integrate such components, they validate these assumptions in the context into which the components are integrated. If the assumptions hold, then the components can be assigned low-level technical safety requirements with ASILs at most as high as the ASIL of the components themselves. For instance, an ASIL C encryption key storage hardware peripheral SEooC can be assigned an ASIL B technical safety requirement to securely store cryptographic keys. But it cannot be assigned this requirement if the requirement is designated ASIL D. This allowance in the standard enables suppliers to experiment with their offerings. As long as these products are developed as SEooCs, suppliers can guide how automotive E/E architectures evolve. This can be seen in the market today, where many automotive CPUs are developed as ASIL D SEooCs.

There are several software engineering principles and hardware techniques that are especially important in a centralized context due to the requirements of ISO 26 262 [19]. For example, software designed around loose coupling between SWCs eases the task of demonstrating absence of cascading failures, a requirement of the standard for high ASILs. Memory protection is imperative to centralization, where absence of interference between SWCs co-existing on the same ECU must be demonstrated. Fault containment, the task of isolating faulty SWCs so that other SWCs are not affected, is more difficult in a centralized architecture where memory and peripherals are shared between the SWCs. Mature techniques employed in traditional architectures, such as static recovery mechanisms, parallel/independent execution paths and multi-version software, all of which are enabled by multi-core CPUs, can be re-used.

Dedicated hardware support for functional safety is crucial to the viability of centralization as a solution to scalability bottlenecks. All current DCU offerings from the leading suppliers provide this. Dedicated support includes memory protection units, fault reporting units, redundant and lock-step cores, error checking and correction at memory and communications level, priority-based interrupt mechanisms *etc*. Integrating ASIL D-capable hardware support with increased computational power has led naturally to ECU offerings capable of supporting domain centralization, as we discuss in the following section.

ISO 26 262 received its first update in 2018 [9]. Among the updates is guidance on constructing *fail-operational* systems. Until now, the focus has been on building *fail-safe* systems. But in increasingly situation-aware and autonomous vehicles, it is insufficient for systems to simply fail safe. Often, a high level of functionality is required post-failure to bring the system to a safe state (*e.g.*, pulling the autonomous vehicle out of traffic in the event of an ECU failure). The standard has also been updated to reflect the increasing dependence of functional safety on cybersecurity. However, the standard treats the activity of achieving secure E/E systems as different from the activity of achieving functional safety. As such, only informative guidance is given regarding interactions between the two activities. For example, cyberattacks can lead to violation of safety goals, so cybersecurity and safety analyses can potentially be harmonized [38], [39].

IV. Supporting Technologies

While some available technologies can be straightforwardly used in centralized architectures, others need to be adapted to facilitate development, operation, and maintenance of centralized vehicular E/E architectures. This section presents technologies that support centralized E/E architectures. We first discuss advances in ECU hardware necessary for the development of powerful centralized controllers, in Section IV-A. Section IV-B then discusses virtualization, a fundamental hardware-supported software technology that allows software reuse and enables effective and safe integration in centralized E/E architecture solutions. Section IV-C discusses networking technologies required to fulfill the needs of centralized E/E architectures, and Section IV-D discusses AUTOSAR's support for modern E/E architectures.

A. *ECU Hardware*

Centralizing the E/E architecture of vehicles around (cross-) domain or zone controllers results in fewer ECUs in a vehicle, but it also increases the processor workload of some ECUs, since these ECUs now have more functionality deployed to them. Thus, the automotive industry is transitioning to ECUs with multi-core CPUs (currently, typical configurations range from two to six cores) to increase the available processing power and allow running several SWCs in parallel on different cores [40].

A number of hardware vendors have released multi-core processors intended for high-performance real-time safety-critical automotive applications. NXP introduced several microprocessors that can be leveraged in domain controllers in modern E/E architectures. For example, their S32S24 [41] is a quad-core ARM R52-based microcontroller, clocked at 800 MHz, advertised for a range of applications including automotive vehicle dynamics, domain control, and safety co-processor applications. Infineon's AURIX [42] microcontroller family is based on Infineon's proprietary TriCore architecture and is offered with up to six cores clocked at 300 MHz. In all cases the circuits are designed to meet ASIL D applications. As expected, the domain controllers already in production leverage existing microcontroller technology. For example, Visteon's SmartCore mentioned in Section II-D is based on Qualcomm's 8155 chip [22], a custom 64-bit eight-core ARM-based microprocessor.

Of critical importance to functional safety is the fact that multi-core CPUs provide an execution environment with potential for increased freedom from interference, since physically distinct cores isolate the concurrent execution of SWCs with different criticality levels. Although the separation is not as complete as in traditional architectures, where each ECU receives one SWC, it is more effective in mitigating the effects of software and hardware failures than using ECUs with powerful single-core CPUs and the same number of SWCs deployed. Nevertheless, in an ISO 26262 compliant context, running mixed-criticality software on the same multi-core CPU requires *partitioning* mechanisms to prevent processing interference among the cores with respect to shared resources, including main memory, communications infrastructure (in the automotive domain, most commonly CAN and LIN buses) and digital and analog I/O [40], [43]. Modern multi-core offerings contain dedicated hardware that helps achieve this separation, which is in direct support of ISO 26262. Aided by dedicated hardware features like core redundancy, fault reporting, memory partitioning and error detection and correction, the final automotive software can run more safely and efficiently as well as more securely.

Adoption of multi-core ECUs involves making a number of important decisions, most importantly on migration and scheduling strategies. The vast majority of cases will see migration of monolithic, verified legacy applications built for single-core ECUs, wherein the options are to either migrate the software unchanged to one of the cores, or to re-engineer the software into (perhaps parallel) tasks to run according to a schedule on a single core or on several cores concurrently [43], [44]. *Partitioned scheduling* can be applied in the former case, where the task is statically allocated to a core and cannot be executed on another core. *Global scheduling* with *task relocation* can be used in the latter case, where under-utilized cores can be dynamically loaded by moving tasks onto them [44]. The first migration strategy is usually viable since individual cores of current multi-core offerings are more capable than earlier single-core ECUs. In the latter case, the relatively high performance capability of modern automotive CPU cores is enough to offset the added computational load of inter-core communication, task synchronization and task scheduling. In the automotive field, as in other computing-reliant fields, the rewards offered by parallel computing can best be gained through proper engineering of parallel software or re-engineering of monolithic software into parallel tasks [1], [40]. In this case, inter-task communication must be minimized, such that cross-core communication is kept to a minimum [45], and scheduling must be such that cores are maximally utilized while ensuring task precedence, mutual exclusion, and time constraints [46]. The automotive industry is tackling these challenges in order to get maximum benefit from parallel computing on ECUs.

Another promising platform for developing powerful domain, cross-domain and zone controllers are the class of *many-core* CPUs. The number of cores in many-core CPUs is large, tens to thousands, and the CPUs are optimized for parallel processing. The cores are typically connected via a Network-on-Chip (NoC) that provides more scalable solutions than traditional bus- or ring-based configurations of multi-core systems [47]. Many-core CPUs offer several advantages over multi-core CPUs [2], [40]; in particular, they are exceptionally well-suited for computationally intense applications.

Domain controllers require significant processing power, often necessitating hardware accelerators, such as accelerators for artificial intelligence applications, cryptography accelerators and linear algebra accelerators. For example, NXP's S32S24 platform has three hardware accelerators, including a Hardware Security Engine (HSE) that provides security support such as cryptographic algorithms, hashing and random number generation [48]. The AURIX TC3xx also leverages cryptography accelerators [49]. In general, hardware suppliers provide security features with dedicated fast hardware implementations of cryptographic operations, secure key storage, secure communications gateways, network bus message authentication, and intrusion detection. All these functions can be performed in software, but hardware acceleration achieves orders of magnitude speedup. Hardware accelerators often leverage graphics processing units (GPUs), digital signal processors (DSPs), and Field Programmable Gate Arrays (FPGAs). The aforementioned cockpit domain controller, Visteon SmartCore, implemented on a single system on a chip (SoC), features the Qualcomm Adreno GPU and the Qualcomm Hexagon DSP for computer graphics. GPUs are also used in existing ADAS solutions for various applications, including pedestrian detection, line following and path planning [50]. FPGAs have seen numerous applications in modern ECUs, including applications in Engine Control Units for parallel acquisition of the engine data and interfacing to other subsystems [51]; LIDAR signal processing and data fusion, security gateways, connectivity and graphics for intertainment [52]; ADAS data aggregation, sensor fusion and object classification [53].

B. *Virtualization*

Virtualization is a crucial technology underlying the E/E centralization trend. Virtualization software and multi-core computing platforms are two technologies that have existed symbiotically for several decades. Factors such as required freedom

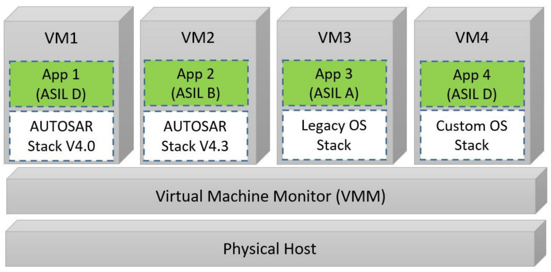

Fig. 5. Four VMs running on a host machine; the VMs are hosting mixed criticality stacks of different operating systems.

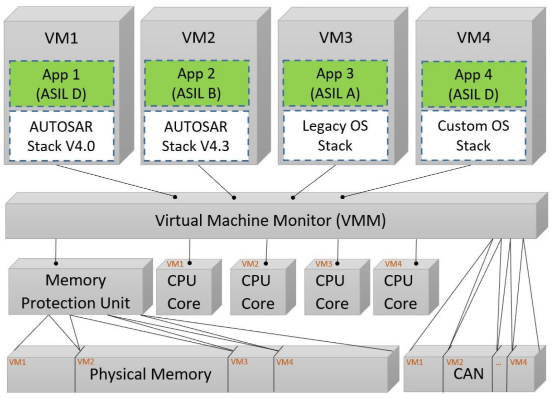

Fig. 6. Virtualization being used to partition memory, CAN hardware and CPU cores among four different SWCs.

from interference and the need for simple software integration have been driving this symbiosis in the embedded automotive domain as well. In this section we describe the factors driving the adoption of virtualization on multi-core automotive ECUs and try to demonstrate why the convenience of virtualization makes it the obvious choice in centralization.

A virtualized system consists of a layer of *virtual machines* (VMs, software), followed by a *virtual machine monitor* (VMM, also software), running on a physical machine host (hardware), as shown in Fig. 5. A VM is software that encapsulates and executes other software by emulating a physical machine [54]. The executed software can be a single program or a full operating system (OS) executing tasks in the usual manner. The VMM is an intermediate software layer that partitions processing, memory, and communication resources amongst VMs, and which schedules and migrates concurrently running VMs onto different resources [43]. VMMs relevant to embedded, automotive applications are known as *hypervisors*. They execute directly on the ECU processor and can be thought of as operating systems for VMs, where the tasks are the VMs themselves.

A major use case for virtualization is consolidation of ECU functions requiring different OSs, as well as different versions of the same OS, as depicted in Fig. 6. For example, it may be necessary to use legacy software (applications and OS) from one ECU side-by-side with the software from another, say AUTOSAR-compliant, ECU. Such integration has several benefits. Reusing legacy ECU software reduces cost and time-to-market, since the legacy software is already verified. But co-existence of different OSs, as well as different versions of the same OS, without the use of a dedicated isolation mechanism is historically a very difficult problem to manage, due to feature interference. Isolating applications requiring different OSs or OS versions inside VMs eliminates this problem completely. Each application receives the OS that it requires and is unaware of the existence of the other OSs by virtue of isolation inside VMs.

Another common scenario in the automotive industry is the consolidation of mixed-criticality ECUs, also captured in Fig. 6. ISO 26262 mandates that freedom from interference among the consolidated functions be guaranteed, such that a fault in one SWC (*e.g.*, a design or implementation flaw, or a fault due to hardware failure) does not propagate to other SWCs of higher ASILs deployed to that same ECU [55], [56]. In the traditional setting, when such a fault manifests itself on an ECU, its effects on other SWCs are minimized by the physical separation of ECUs. For instance, the malfunctioning SWC cannot overwrite the memory of a SWC on a different ECU. The only pathway between SWCs that must be secured against these effects is the communication link between ECUs. Mature methods to achieve this already exist, such as detection of transmission of malformed data over the communications link. But this physical separation disappears in a centralized setting, where the functions of different ECUs are consolidated onto a single ECU. According to ISO 26262, to consolidate two ECUs with functions specified by different ASIL requirements, *e.g.*, ASIL D and ASIL A, the two functions must either be developed to ASIL D requirements, or a mechanism developed to ASIL D requirements must be used to guarantee freedom from interference between the two functions.

Virtualization is a perfect solution to these difficulties. Different legacy SWCs can be deployed side-by-side inside VMs on a single ECU. Since virtualization facilitates the migration of legacy software, heterogeneous software can be consolidated, while isolation inside VMs eliminates concerns surrounding co-existence of different versions of software and operating systems. The most suitable OS can be used on each VM, while the VMM provides the required freedom from interference between the VMs. For example, Visteon's SmartCore DCU, which implements an integrated digital cockpit (introduced in Section II-D), uses a hypervisor to integrate the non-ASIL infotainment domain running on Linux with the ASIL B instrument cluster domain. Hypervisors for embedded applications are normally developed to ASIL D requirements, so they are a COTS solution to ISO 26262-compliant integration of software functions. Finally, virtualization is versatile enough to facilitate the building of flexible and scalable architectures in which moving SWCs between ECUs is feasible.[3]

Virtualization was first demonstrated to be a viable solution to the integration of heterogeneous software on consolidation of business IT infrastructure. Then, the technology was adopted as the centralization-enabling technology for avionic systems in the early 2000s. Now, commercial hypervisors on the market,

[3]It is worth noting that virtualization is generally adopted as a commodity off-the-shelf solution for practical reasons, but is not currently the only way to achieve separation. Bespoke partitioning solutions that do not necessarily resemble virtualization can be implemented to provide the same separation guarantees.

many of which have seen enormous success in safety-critical applications in the aerospace, defense and medical fields, are driving the adoption of virtualization as the integration solution of choice in automotive E/E architectures. Established suppliers to aerospace and defense, such as General Dynamics and Wind River Systems, are all offering mission-critical hypervisors that are well suited for automotive applications.

Virtualization technologies have several limitations [54]. First, real-time performance is not predictable in general, since virtualization can result in scheduling on two levels, at the hypervisor and at the VM level. One way to address this limitation is by static allocation of VMs to cores. Second, VMs utilize considerable memory resources. Third, virtualization of ECU peripherals (*e.g.*, CAN bus) is a complex problem that requires further investigation. Fourth, hypervisors must be kept small to be safe, secure and bug-free so that they can be developed to the highest safety integrity level when used as SEooCs. Finally, in the past, there has been limited hardware support for virtualization in ECUs. However, this situation is currently improving with the introduction of hardware virtualization support in automotive CPUs. These extensions accelerate the execution of virtualized environments. CPUs based on ARM cores, such as offerings from NXP, Infineon and Texas Instruments, all include hardware virtualization support.

Virtualization is a major opportunity for both industry and academia. For industry, it provides an off-the-shelf solution to a difficult problem facing all automotive manufacturers; and for academia it offers new methods for integrating the development methodologies of increasingly complex automotive systems, with the requirements of standards like ISO 26262. In both cases, virtualization is an indispensable enabling technology. Thus, we expect that the aviation story will be repeated and that virtualization will be the technology of choice in the future of the automotive industry for standards compliance.

C. Networking

Traditionally, CAN buses have been used as the *de facto* standard for in-vehicle connectivity [3], [57]. CAN buses are serial buses used to interconnect ECUs within a vehicle. With the increasing complexity of vehicular functions and their resulting data traffic, CAN networks are becoming less suited for future E/E architectures given their data rate limit of 1 megabit per second [57]. CAN FD (flexible data rate) and automotive Ethernet can be used to replace CAN buses in different domains [3].

The features of CAN FD provide the necessary performance upgrade required for advanced functions in the powertrain domain (*e.g.*, hybrid and all-electric drives), the body domain (*e.g.*, advanced comfort-related body components), and the chassis domain (excluding the driver assistance sub-domain) [3], [57]. Such features include a practical maximum data rate of almost 6 Mbit/s [58], packets with a 64 byte data field (as opposed to CAN's 8 byte data field) [59], backward compatibility with legacy CAN systems, and low power consumption [57]. Transitioning an existing network from CAN to CAN FD can reduce bus load significantly, allowing for more data to be transmitted with acceptable delays [60]. However, being identical at the signalling level to CAN, we can expect similar types of limitation with respect to maximum number of nodes and maximum cable length. In fact, though the authors could not locate any supporting experimental evidence, it can be inferred that maximum cable length is significantly reduced over CAN for the higher CAN FD data rates, a factor that may not be acceptable for certain manufacturers and in certain applications. More importantly, CAN FD is not intrinsically versatile enough as a networking technology as required in modern and future software-defined vehicles. In these applications, *service-orientation* becomes the dominant software paradigm. The integration of predefined services, as well as services not known at the time of manufacture of the vehicle – personal gadgets such as phones, music players, fitness trackers, *etc.*, as well as integration with traffic infrastructure – requires data rates and versatility in terms of functional separation, bandwidth reservation, security, *etc.*, that CAN FD cannot provide [61]. Above a certain level of complexity, still faster, more reliable and more versatile communication technologies, *e.g.*, automotive Ethernet [62], will be needed to provide the higher bandwidth for different automotive applications, including the ADAS domain and the infotainment domain.

Automotive Ethernet [11] refers to Ethernet-based communication schemes used within vehicles and which are completely different at the signalling level from original Ethernet. Automotive Ethernet provides a lightweight and cost-effective communication medium which uses single unshielded twisted-pair cabling. Communication data rates of 100 Mbit/s and 1000 Mbit/s are common, with IEEE 802.3ch-2020 specifying maximum data rates of 2.5, 5 and 10 Gbit/s. The IEEE P802.3cy task force is working on specifications at 25, 50 and 100 Gbit/s. The high data rate presents a significant advantage of automotive Ethernet over traditional automotive field buses. Indeed, market experts are confident that full adoption of automotive Ethernet is inevitable due to an estimated 4 TB/h data volume production in future autonomous vehicles [6]. However, as it is a relatively new technology, the literature does not yet seem to reflect an established set of design principles for the physical layer of automotive Ethernet-based in-vehicle networks. Studies are currently revealing the electro-magnetic compatibility and signal quality characteristics of this networking technology [63], [64], the foundations of physical layer design. Also, the Open Alliance provide several test specifications to be used at the physical level of automotive Ethernet applications [65].

In order to provide support for high-bandwidth and low-latency real-time in-vehicle communication, a set of standards, the TSN (Time-Sensitive Networking) standards [66], has been developed. The TSN standards extend the AVB (Audio Video Bridging) standards[4] [67] to provide time-sensitive communication over Ethernet. Clock synchronization is crucial in satisfying system-level real-time requirements. In increasingly complex vehicles, hard real-time requirements, those which can have critical consequences for functional safety if they are violated

[4]The AVB standards [67] were focused on a high quality of service (QoS) when manipulating audio/video (AV) data, *i.e.*, guaranteeing bandwidth and enabling data synchronization for AV communications.

while the E/E system is operational, will necessarily rely on timing guarantees on network performance. Examples include ADAS and "by-wire" systems. TSN's IEEE 802.1AS-2020 [68] standard specifies time synchronization mechanisms across an Ethernet network. It defines a grandmaster which distributes the time base in the network to guarantee that all components follow the same global clock, but also allows the operation of several different time domains in the same network. Further, the standard defines the transport of synchronized time. The standard's protocol for clock synchronization is a variation of the Precision Time Protocol defined by IEEE 1588-2008 [69].

TSN standard IEEE 802.1Q-2018 [70] provides bounded low latencies. Several different traffic classes are defined which are based on priorities. The standard then defines two scheduling mechanisms for transmission of Ethernet frames: strict priority and credit based shaper. A third scheduling algorithm is also defined, TAS (Time Aware Shaper), which deterministically schedules frames using a Time Division Multiple Access (TDMA) scheme that enables exclusive right of transmission of time-critical frames within a window of time in each time period. To achieve even lower latency for the highest-criticality frames, frame pre-emption is proposed, whereby lower-priority frames are interrupted and split up so that high criticality frames can be sent as early as possible.

IEEE 802.1Q-2018 also defines measures that enforce network stream control and policing. For instance, the standard specifies how to avoid overloading the communication network by restricting the allowed ingress bandwidth on certain ports. For fault tolerance, the IEEE 802.1CB-2017 TSN standard [71] specifies redundancy measures to guarantee fault tolerance within the vehicular network. The standard defines methods to identify and replicate packets, as well as methods to identify otherwise duplicated packets and eliminate them. The standard, however, does not identify redundant paths for the packets. Instead, IEEE 802.1Q-2018 defines methods to configure multiple paths on the network to provide redundancy. Together, the IEEE 802.1Q-2018 and IEEE 802.1CB-2017 TSN standards can be leveraged for dynamic network configuration [18]. The capability of dynamic network configuration will be essential to support future vehicular networks to enable software updates via, *e.g.*, FOTA (Firmware Over-The-Air) and SOTA (Software Over-The-Air). This capability, for example, would be required to add a new feature that requires increased bandwidth of an Ethernet link. Further, dynamic network configuration will be employed instead of static, manual network configurations to efficiently handle an increasing number of network configurations across different variants within vehicular product lines [72].

Overall, the TSN standards aim to provide the fundamental guarantees on network performance (bounds on latencies, bandwidth reservation, traffic separation *etc.*) that are prerequisite to implementing mission- and safety-critical requirements. Another protocol, Time-Triggered Ethernet (TTEthernet) [73], is the SAE AS6802 standard that defines mechanisms to support determinism, real-time communication, and functional safety related applications. For example, the standard supports functional safety by partitioning a physical communication channel into independent sub-channels that are free of logical and temporal interference, thus avoiding fault propagation.

Like TSN, TTEthernet requires a statically defined schedule of time slot allocation to each of three types of traffic, including time-critical traffic, in order to guarantee transmission times. In both cases, the schedule must be solved offline, for example by SMT techniques. But unlike TSN, the schedule must be solved for all traffic classes and routes involved. In contrast, the schedule in the case of TSN must only be solved for the time-critical flows, while other flows are achieved best-effort through various mechanisms. However, it is possible to optimize all flows by solving the time-critical flow schedule taking into account non-critical flows [74]. Since not all flows in a vehicle application are time-critical, variants of a vehicle model based entirely on TTEthernet may require more schedules to be solved due to different network configurations than those of a vehicle model based entirely on TSN [75]. Where vehicle models change frequently, and along with them vehicle variants, TSN perhaps affords higher flexibility than TTEthernet.

While next-generation architectures will use a clever mix of CAN, CAN FD, LIN, FlexRay, and Ethernet technologies, many automakers already use Ethernet in production vehicles, and/or are planning to use an Ethernet backbone. In fact, automotive Ethernet, extended appropriately with real-time and safety extensions (such as AVB, TSN and TTEthernet), is emerging as the most fitting solution for inter-domain communication. With real-time extensions, automotive Ethernet is also perfectly suited for intra-domain communications with smart actuators, where timing guarantees must be met, for example for electric motor control and SONAR/RADAR/LIDAR sensing systems. Moreover, we can expect that the vehicle will be increasingly required to integrate with Intelligent Transportation Systems, a role in which real-time communication requirements will become more and more stringent as regulations on how the vehicle communicates with its environment become more demanding. This will place extra demand on the in-vehicle networking technology to scale to meet this demand in the future.

D. AUTOSAR

Centralization of the E/E architecture requires proper support at the level of software architecture. The automotive industry has been gradually moving toward compliance with several standards, including AUTOSAR. The trend of AUTOSAR standardization has several advantages, all of which are aligned with the goals of E/E centralization, such as facilitating the integration of components from different suppliers, reuse of components across different vehicle programs and platforms, and controlling failures in a systematic manner. In this section, we focus on support provided by AUTOSAR for centralized architectures.

AUTOSAR (AUTomotive Open System ARchitecture) [76]–[80] offers a framework for development of automotive embedded systems. The standard was developed by a consortium of OEMs, suppliers and vendors, with the aim of facilitating portability and composability of vehicular software. AUTOSAR's software architecture defines three software layers that execute on an ECU, as shown in Fig. 7. These layers are the *Application*

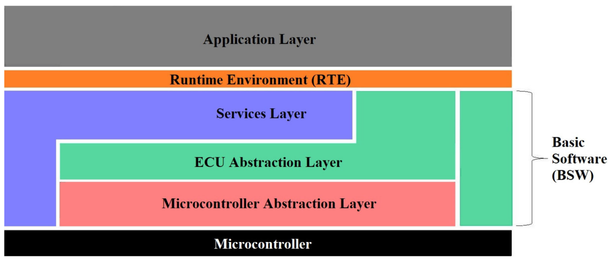

Fig. 7. AUTOSAR's layered software architecture, adapted from [79].

Layer, *Runtime Environment (RTE)*, and *Basic Software (BSW)*. The Application layer implements the functionality of an ECU in the form of hardware-independent SWCs, that either perform vehicular functions or sensor/actuator functions. The RTE is the communication medium of the architecture, *i.e.*, SWCs communicate with other SWCs (on the same or on a different ECU), and with BSW modules, through the RTE. The RTE is the gateway through which vehicle application software accesses ECU functionality to achieve its goals. This functionality is implemented in the BSW, across several layers of abstraction (Fig. 7), into services usable by the application software. The highest, most abstract layer of BSW is the Services Layer, which provides services for the applications, the RTE, and the BSW modules. These services include *system services*, such as the OS, which are functions that can be used by all modules. At the other end of the abstraction spectrum lies the Microcontroller Abstraction Layer (MCAL), the lowest layer of BSW. It is composed of drivers that directly access the microcontroller and the peripherals. This layer is hardware-specific.

AUTOSAR's layered architecture supports centralization by separating vehicular SWCs from the hardware platform, effectively improving the software's modularity and reuse, and, in turn, facilitating its deployment and relocation to different ECUs within an E/E architecture. Therefore, we view AUTOSAR as an important software architectural pillar for centralization, and as an E/E architectural solution to scalability issues that have been plaguing the automotive industry.

Regarding functional safety, AUTOSAR defines mechanisms that facilitate detecting and handling faults to ensure an interference-free execution environment for SWCs in an AUTOSAR-compliant ECU [81], which is necessary for compliance with ISO 26262. For software, memory partitioning is one such feature where different OS applications are given restricted access to isolated memory partitions, to help enforce functional separation between the applications. Memory access by the applications is then monitored by a memory partitioning mechanism, which is best placed in the OS. The OS can then be either entirely executed in the privileged mode[5] [81], or it can be split such that only the safety-relevant functions run in the privileged mode [82]. Besides memory partitioning, *timing protection* in AUTOSAR allows the OS to react to timing failures and initiate predefined actions, to achieve temporal separation (*i.e.*, avoid propagation of timing faults) between applications. *Logical supervision* mechanisms can also be used to monitor the control flow of software and ensure its correct execution. Finally, *end-to-end communication protection* mechanisms can be leveraged to handle faulty signal transfer between OS applications or ECUs.

[5]Processing mode with unrestricted access to memory and underlying hardware.

AUTOSAR supports functional safety in hardware through diagnostics mechanisms, such as core and RAM tests [81]. Core tests detect faults in the processing unit of an ECU that can result in incorrect outputs, *e.g.*, faults in the arithmetic logic unit. RAM tests detect faults which can corrupt the volatile memory of an ECU. These tests have full hardware support in modern automotive-grade CPUs.

In support of security, AUTOSAR defines the Secure Onboard Communication (SecOC) module, a BSW module that provides authentication mechanisms for sensitive data transmission within automotive networks. Further, AUTOSAR defines the Crypto Abstraction Library (CAL) and Crypto Service Manager (CSM) to support cryptographic services.

Although AUTOSAR was originally developed for single-core ECUs, the standard was later extended to support multi-core ECUs, by addressing the allocation of BSW to different cores (*i.e.*, the parallelization of BSW) [83], [84]. The parallelization of the SWCs in the application layer, however, is not addressed in the standard. Limited research investigating parallelization of SWCs exists discussing either the parallelization process at a high level (for example, the work of Macher *et al.* [45]), or specific approaches without applying them to real-world AUTOSAR systems [44], [85], [86].

The industry's shift to centralized E/E architectures will often necessitate migrating already verified sequential AUTOSAR applications onto modern multi-core platforms. This migration faces several challenges [40], [44], [83]. First, each SWC in AUTOSAR is composed of *runnables*, and runnables must be mapped to tasks which, in turn, will be allocated to cores. The runnable-to-task mapping and the task allocation to cores need to be carried out simultaneously, to avoid excessive cross-core communications while maintaining the system's constraints (*e.g.*, precedence constraints). Second, system services in the Services Layer of AUTOSAR's BSW (Fig. 7) were designed for single-core ECUs. As mentioned earlier, system services are auxiliary functions that can be used by all modules within all AUTOSAR layers (*e.g.*, an operating system). Thus, migrating to multi-core ECUs might require adapting these system services to support different instances of the same BSW module running on different partitions. Also, parallelization at the application level is a rather challenging task. To address these challenges, future work should define parallel design patterns and provide guidelines for the mapping, scheduling, synchronization, and communication [40].

Many automotive OEMs have made a commitment to migrate their software to AUTOSAR, which is an enormous undertaking. But while OEMs have been transitioning to the version of AUTOSAR described here, called AUTOSAR "Classic", the AUTOSAR consortium have been working on another AUTOSAR platform to precisely target the demands placed on vehicular E/E architectures by the ever-increasing complexity and diversity of vehicle functions, in particular high-bandwidth applications such as ADAS and highly automated driving, infotainment, as well as V2X connectivity. The outcome is the AUTOSAR "Adaptive" standard, released in 2017.[6] Effectively, AUTOSAR Adaptive makes a switch to a service-oriented architecture where services and clients are dynamically linked during ECU runtime.

AUTOSAR Adaptive is not a replacement for AUTOSAR Classic—they target different applications and therefore complement each other. While AUTOSAR Classic aims at safety-critical systems with hard real-time requirements and with low-bandwidth communication, implemented on low-resource hardware, AUTOSAR Adaptive targets high performance, low criticality systems, with high-bandwidth and Over-The-Air (OTA) requirements. The two platforms are envisioned to co-exist within a vehicular network, together with non-AUTOSAR platforms and backend systems, such as road infrastructure [87]. The interoperability of the two platforms is enabled by the common Foundation—a set of common requirements and technical specifications [88]. While AUTOSAR Adaptive is expected to take over domains such as autonomous/automated driving, connectivity, and infotainment, AUTOSAR Classic will remain the platform of choice for traditional automotive domains such as powertrain and chassis. With the further trend of consolidation of functionalities on multi/many-core systems crossing the domain boundaries, it is not hard to envision the co-existence of AUTOSAR Classic partitions and AUTOSAR Adaptive partitions on the same platform. Efficient, yet safe, scheduling algorithms for such systems have been proposed (for example, the work of Kritikakou *et al.* [89]).

V. Discussion

Individual solutions to the scalability problem in the automotive industry, such as increasing the capabilities of ECUs or changing the ubiquitous CAN to CAN FD or Ethernet, require proper support at the E/E architectural level. It is not a viable solution to simply upgrade individual ECUs as required to cope with increased processing load, as eventually sophisticated processing creates large volumes of data. Similarly, it is cost-prohibitive to increase the throughput capability of a decentralized architecture by upgrading *all* ECUs with automotive Ethernet technology.

The centralized architecture is a promising approach as it addresses scalability along different vectors: it is possible to incorporate increasingly powerful ECUs and allow versatility in SWC deployment topologies, while reducing the number of software variants and easing product line engineering efforts. Further, the centralized architecture future-proofs the critical high-speed communication backbone by minimizing the number of nodes present on it.

We have witnessed E/E centralization being applied in the industry. Suppliers have been quick to support centralization experiments. Integration of powerful, multi-core automotive CPUs with custom ECU hardware is ubiquitous on the market. At present, DCU offerings include hardware support for the critical functional safety element, hypervisors and, typically, support for advanced automation features such as RADAR and LIDAR data processing, machine vision and learning, and sensor fusion. They also include hardware support for vehicle security with respect to cyberattacks. As we have seen, OEMs and suppliers have been experimenting with these new technologies, which often have very long lead times. This indicates that OEMs and suppliers believe that these experiments are worth the time and financial investments.

As expected, infotainment and ADAS have been the first applications of centralization and will remain hot spots for centralization in E/E architectures. The implementation of centralization within these particular domains demonstrates the availability of the technology needed for these applications (*e.g.*, multi-core CPUs and automotive Ethernet). However, the latest Tier 1 and OEM offerings, and the technological state of the art presented in this paper, indicate that there is enough technological critical mass to make centralization a viable solution in automotive applications with hard real-time requirements, high performance, and high safety criticality levels, such as the powertrain and chassis domains.

In general, developing an automotive E/E architecture is a complex venture that involves solving a multi-objective optimization problem. Historically, in the automotive industry, global optima, or even near optima, have not been typically attained. The E/E architecture development process involves finding a satisfactory allocation of vehicular functions over different ECUs and within ECUs, in order to satisfy a large number of constraints and functional and non-functional requirements, including cost minimization, efficient use of available resources, maintainability, and scalability. For the industry, moving to centralized E/E architectures is a very large undertaking that comes with generic challenges of engineering an E/E architecture, but also creates some distinct problems. For example, on the functional safety side, as functions are consolidated on a powerful ECU, a failure of that ECU might have larger implications on functional safety. In turn, effective and efficient safety architectural patterns are required to provide fail-operational behavior in case of such failures. Further, while multi-core ECUs offer increased processing power, how to engineer, and especially how to *re-engineer*, software to obtain speedup with more cores is still an open question in the automotive industry. Another major challenge of centralization lies in adapting existing automotive software and safety processes to achieve the expected gains from centralization. For example, in the case of domain-centralized architectures, integration testing needs to be adapted and performed at several integration levels, *i.e.*, integration testing at the level of the domain cluster, and then integration testing of all domain clusters. Also, historically, different domains within automotive E/E architectures have leveraged different development tools and methods. Thus, when integrating functions from such domains onto one ECU in a centralized architecture, harmonization of processes and tools would be a major effort for the automotive industry.

The aviation industry transitioned to centralized architectures successfully in a more strict and complex environment, which is testament that the aforementioned challenges in the automotive

[6]https://www.autosar.org/standards/adaptive-platform/

industry can be overcome. On the backdrop of the successful transition in the aviation domain to centralized E/E architectures, it is clear that the elements discussed in this paper, namely industry trends, the current critical state of E/E architectures, quick response from suppliers, support from the AUTOSAR standard, and the increasing need to be standards-compliant, form a critical mass that makes centralization the E/E architectural approach on which the vehicle of the future will be built.

VI. Conclusion

The increased number and sophistication of functions in modern cars is driving the need for the evolution of automotive E/E architectures. The aviation domain has seen a successful transition to centralized E/E architectures, and in the authors' opinion it is evident that automotive E/E architectures need to transition to this type of E/E architecture as an immediate next step in the face of increasing vehicle complexity. Current industry trends also clearly indicate that this is the future of automotive E/E architectures. In this article we report on these trends. Then, we discuss several enabling technologies. Throughout the article we highlight the implications of centralization and its enabling technologies on functional safety.

Further, while this article motivates and makes the case for centralization in the automotive industry, it can also serve as a *catalogue* of the specific technologies that can be used in each of the four presented broad classes of enabling technologies. The article also highlights challenges and gaps in the enabling technologies and standards that still need to be investigated by researchers and practitioners.

Acknowledgment

The authors are particularly grateful to B. Wasacz of FCA US LLC for fruitful discussions on centralized vehicular E/E architectures. The authors also thank Prof. B. Bilgin of McMaster University for his valuable comments on initial drafts of this article, and the various anonymous reviewers for their extremely helpful feedback.

References

[1] D. Reinhardt and M. Kucera, "Domain controlled architecture - a new approach for large scale software integrated automotive systems," *PECCS*, vol. 13, pp. 221–226, 2013. [Online]. Available: https://www.scitepress.org/papers/2013/43407/43407.pdf

[2] M. Di Natale and A. L. Sangiovanni-Vincentelli, "Moving from federated to integrated architectures in automotive: The role of standards, methods and tools," *Proc. IEEE*, vol. 98, no. 4, pp. 603–620, Apr. 2010.

[3] V. M. Navale, K. Williams, A. Lagospiris, M. Schaffert, and M.-A. Schweiker, (R)evolution of E/E architectures," *SAE Int. J. Passenger Cars-Electron. Elect. Syst.*, vol. 8, no. 2, pp. 282–288, 2015.

[4] J. X. Wang, *Will Autonomous Cars Look Like Unmanned Aerial Vehicles (UAVs)?*CRC Press, Jan. 2019. Accessed: Jun. 6, 2019. [Online]. Available: https://www.crcpress.com/authors/i351-john-wang/news/i5114-will-autonomous-cars-look-like-unmanned-aerial-vehicles-uavs

[5] J. M. Amend, "Industry must go digital or die," WardsAuto, Sep. 2017, Accessed: Oct. 29, 2019. [Online]. Available: https://www.wardsauto.com/technology/delphi-industry-must-go-digital-or-die

[6] O. Burkacky, J. Deichmann, G. Doll, and C. Knochenhauer, "Rethinking car software and electronics architecture," McKinsey Center for Future Mobility, Feb. 2018. Accessed: Jun. 6, 2019. [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/rethinking-car-software-and-electronics-architecture

[7] Robert BoschGmbH, "E/E-architecture in a connected world," 2017, Accessed: Jul. 16, 2019. [Online]. Available: https://www.asam.net/fileadmin/documents/Events/Technical_Workshops/2017_Technical_Seminar/08a_EE_Architecture_in_A_Connected_World.pdf

[8] A. McAuslin, "Safe autonomous systems: Centralized or distributed processing?," NXP, Oct. 2016. Accessed: Jun. 6, 2019. [Online]. Available: https://blog.nxp.com/automotive/safe-autonomous-systems-centralized-or-distributed-processing

[9] ISO. *Road Vehicles - Functional Safety*, ISO Standard 26262, 2018.

[10] S. Kugele *et al.*, "Research challenges for a future-proof E/E architecture—A project statement," in *Proc. INFORMATIK 2017*, M. Eibl and M. Gaedke, Eds. Gesellschaft für Informatik, Bonn, 2017, pp. 1463–1474.

[11] K. Matheus and T. Königseder, *Automotive Ethernet*, 1st ed. USA: Cambridge Univ. Press, 2015.

[12] W. Haas and P. Langjahr, "Cross-domain vehicle control units in modern E/E architectures," in *Proc. 16. Internationales Stuttgarter Symp.*, Springer, 2016, pp. 1619–1627.

[13] US Department of Transportation National Highway Traffic Safety Administration, "Theft Protection and Rollaway Prevention," *Code of Federal Regulations Title 49*, Part 571, 2010.

[14] S. Kanajan, C. Pinello, H. Zeng, and A. Sangiovanni-Vincentelli, "Exploring trade-off's between centralized versus decentralized automotive architectures using a virtual integration environment," in *Proc. Design Automation Test Europe Conf.*, 2006, vol. 1, pp. 1–6. [Online]. Available: https://past.date-conference.com/proceedings-archive/2006/DATE06/PDFFILES/05E_1.PDF

[15] G. B. S. Tagawa and M. L. de Oliveira e Souza, "An overview of the integrated modular avionics (IMA) concept," in *Proc. DINCON*, 2011, pp. 277–280.

[16] C. B. Watkins and R. Walter, "Transitioning from federated avionics architectures to integrated modular avionics," in *Proc. IEEE/AIAA 26th Digit. Avionics Syst. Conf.*, 2007, pp. 2.A.1-1–2.A.1-10.

[17] A. Mairaj, "Preferred choice for resource efficiency: Integrated modular avionics versus federated avionics," in *Proc. IEEE Aerosp. Conf.*, 2015, pp. 1–6.

[18] S. Brunner, J. Röder, M. Kucera, and T. Waas, "Automotive E/E-architecture enhancements by usage of ethernet TSN," in *Proc. IEEE 13th Workshop Intell. Solutions Embedded Syst. (WISES)*, 2017, pp. 9–13.

[19] D. Reinhardt, D. Kaule, and M. Kucera, "Achieving a scalable E/E-architecture using AUTOSAR and virtualization," *SAE Int. J. Passenger Cars-Electron. Elect. Syst.*, vol. 6, no. 2, pp. 489–497, Apr. 2013.

[20] J. Bach, S. Otten, and E. Sax, "A taxonomy and systematic approach for automotive system architectures—From functional chains to functional networks," in *Proc. 3rd Int. Conf. Veh. Technol. Intell. Transport Syst.*, VEHITS, INSTICC. SciTePress, 2017, vol. 1, pp. 90–101. [Online]. Available: https://www.scitepress.org/Papers/2017/63076/63076.pdf

[21] W. Stolz, R. Kornhaas, R. Krause, and T. Sommer, "Domain control units - the solution for future E/E architectures," SAE Tech. Paper, Tech. Rep. 2010-01-0686, Apr. 2010.

[22] Visteon Corporation, "Domain controller-ECU consolidation for the cockpit," 2018. Accessed: Jun. 1, 2019. [Online]. Available: https://www.visteon.com/products/domain-controller/

[23] E. Rosman, "Active safety teach in," 2018. Accessed: Jun. 1, 2019. [Online]. Available: http://q4live.s22.clientfiles.s3-website-us-east-1.amazonaws.com/336558720/files/doc_presentations/2018/Aptiv-Active-Safety-Teach-In-June-2018.pdf

[24] PLC Aptiv, "Now entering the supercomputing era," 2018. Accessed: Jun. 1, 2019. [Online]. Available: https://www.aptiv.com/media/article/2018/01/10/now-entering-the-supercomputing-era

[25] PLC Aptiv, "The autonomous driving platform: How will cars actually drive themselves?," 2018. Accessed: Jun. 1, 2019. [Online]. Available: https://www.aptiv.com/media/article/2018/01/07/the-autonomous-driving-platform-how-will-cars-actually-drive-themselves

[26] Robert Bosch GmbH, "Driver assistance system domain controller (DASy)," 2019. Accessed: Jun. 1, 2019. [Online]. Available: https://www.bosch-mobility-solutions.com/en/products-and-services/passenger-cars-and-light-commercial-vehicles/automated-driving/driver-assistance-system-domain-controller/

[27] Robert Bosch GmbH, "Vehicle control unit," 2019. Accessed: Jun. 1, 2019. [Online]. Available: https://www.bosch-mobility-solutions.com/en/products-and-services/passenger-cars-and-light-commercial-vehicles/powertrain-systems/electric-drive/vehicle-control-unit/

[28] STMicroelectronics, "Vehicle control unit (VCU)," 2019. Accessed: Sep. 4, 2019. [Online]. Available: https://www.st.com/en/applications/electro-mobility/vehicle-control-unit-vcu.html

[29] Continental AG, “High performance powertrain master,” 2019. Accessed: Sep. 4, 2019. [Online]. Available: https://www.continental-automotive.com/en-gl/Trucks-Buses/Vehicle-Chassis-Body/Vehicle-Control/High-Performance-Powertrain-Master

[30] PR Newswire, “Global automotive domain control unit (DCU) industry report 2018-2019 with focus on the chinese market,” 2019. Accessed: Jun. 1, 2019. [Online]. Available: https://www.prnewswire.com/news-releases/global-automotive-domain-control-unit-dcu-industry-report-2018-2019-with-focus-on-the-chinese-market-300807740.html

[31] Electronic Design, “BMW and Audi want to separate vehicle hardware from software,” 2019. Accessed: Oct. 1, 2019. [Online]. Available: https://www.electronicdesign.com/automotive/bmw-and-audi-want-separate-vehicle-hardware-software

[32] newTIS.info, “Body domain controller,” 2018. Accessed: Jun. 1, 2019. [Online]. Available: https://www.newtis.info/tisv2/a/en/g30-530i-lim/components-connectors/components/components-with-a/a258-body-domain-controller/1VnYYvxP5Z

[33] Green Car Congress, “Volkswagen’s EB for EVs: Long electric range, open-platform, open-space, pricing for the volume market tablet on wheels,” Oct. 2016. Accessed Jun. 6, 2019. [Online]. Available: http://www.greencarcongress.com/2016/10/20161005-meb.html.

[34] WardsAuto World, “All-electric MEB platform to drive new firsts at VW,” 2018. Accessed: Jun. 1, 2019. [Online]. Available: https://www.wardsauto.com/technology/all-electric-meb-platform-drive-new-firsts-vw

[35] S. Bath and S. Shinde, “Ethernet journey at land rover: Challenges in the development of an ethernet backbone,” presented at *Vector Automot. Ethernet Symp.*, May, 2017. Accessed: Feb. 12, 2021. Available: https://assets.vector.com/cms/content/events/2017/vAES17/vAES17_02_Ethernet-Journey-JLR_Bath_Shinde.pdf

[36] ISO. *Road Vehicles - Functional Safety - Part 9: Automotive Safety Integrity Level (ASIL)-Oriented and Safety-Oriented Analyses*, ISO Standard 26262, 2018.

[37] ISO. *Road Vehicles - Functional Safety - Part 10: Guidelines on ISO 26262*, ISO Standard 26262, 2018.

[38] ISO. *Road Vehicles - Functional Safety - Part 2: Management of Functional Safety*, ISO Standard 26262, 2018.

[39] T. Chowdhury *et al.*, “Safe and secure automotive over-the-air updates,” in *Proc. Int. Conf. Comput. Saf., Rel., Secur.*, Springer, 2018, pp. 172–187.

[40] M. Becker, D. Dasari, V. Nélis, M. Behnam, L. M. Pinho, and T. Nolte, “Investigation on AUTOSAR-compliant solutions for many-core architectures,” in *Proc. IEEE Euromicro Conf. Digit. Syst. Des.*, 2015, pp. 95–103.

[41] NXP Semiconductors N.V., “S32S24: Safety microcontroller for automotive applications,” 2019. Accessed: Jul. 16, 2019. [Online]. Available: https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/s32-automotive-platform/safety-microcontroller-for-automotive-applications:S32S24

[42] Infineon, “32-bit AURIX microcontroller based on TriCore,” 2019. Accessed: Jul. 16, 2019. [Online]. Available: https://www.infineon.com/cms/en/product/microcontroller/32-bit-tricore-microcontroller/

[43] A. Herkersdorf, H.-U. Michel, H. Rauchfuss, and T. Wild, “Multicore enablement for automotive cyber physical systems,” *It-Inf. Technol. Methoden Innov. Anwendungen Informatik Informationstechnik*, vol. 54, no. 6, pp. 280–287, 2012.

[44] S. Widlund and A. Annenkov, “Migrating a single-core AUTOSAR application to a multi-core platform: Challenges, strategies and recommendations,” M.S. thesis, Dept. Comput. Sci. Eng., Chalmers Univ. Technol., Gothenburg, Sweden, 2017.

[45] G. Macher, A. Höller, E. Armengaud, and C. Kreiner, “Automotive embedded software: Migration challenges to multi-core computing platforms,” in *Proc. IEEE 13th Int. Conf. Ind. Inf. (INDIN)*, 2015, pp. 1386–1393.

[46] J. Schneider, M. Bohn, and R. Röβger, “Migration of automotive real-time software to multicore systems: First steps towards an automated solution,” in *Proc. 22nd EUROMICRO Conf. Real-Time Syst.*, 2010, pp. 34–40.

[47] L. Benini and G. De Micheli, “Networks on chips: A new SoC paradigm,” *Computer*, vol. 35, no. 1, pp. 70–78, 2002.

[48] NXP Semiconductors N.V., “Vehicle electrification solutions - semiconductors for the next generation of electric vehicles,” 2019. Accessed: Oct. 21, 2019. [Online]. Available: https://www.nxp.com/docs/en/brochure/VEHICLE-ELECTRIFICATION-BR.pdf

[49] Infineon, “AURIX 32-bit microcontrollers for automotive and industrial applications: Highly integrated and performance optimized,” 2019. Accessed: Oct. 21, 2019. [Online]. Available: https://www.infineon.com/dgdl/Infineon-TriCore_Family_BR-ProductBrochure-v01_00-EN.pdf?fileId=5546d4625d5945ed015dc81f47b436c7

[50] I. S. Olmedo, N. Capodieci, and R. Cavicchioli, “A perspective on safety and real-time issues for GPU accelerated ADAS,” in *Proc. IECON 2018-44th Annu. Conf. IEEE Ind. Electron. Soc.*, 2018, pp. 4071–4077.

[51] E. D. Magazine, “ECUs tap FPGAs for championship performance,” 2013. Accessed: Sep. 2020. [Online]. Available: https://www.electronicdesign.com/technologies/digital-ics/article/21798189/ecus-tap-fpgas-for-championship-performance

[52] Intel, “Automotive applications,” Accessed: Sep. 2020. [Online]. Available: https://www.intel.ca/content/www/ca/en/automotive/products/programmable/applications.html

[53] M. Smolaks, “Xilinx unveils high performance autonomous car brain,” Accessed: Sep. 2020. [Online]. Available: https://aibusiness.com/document.asp?doc_id=761124&site=aibusiness

[54] N. Navet *et al.*, “Virtualization in automotive embedded systems: An outlook,” presented at *RTS Embedded Syst.*, Paris, Mar. 31, 2010. Accessed: Feb. 12, 2021. [Online]. Available: https://www.realtimeatwork.com/wp-content/uploads/RTS10_virtualization_bw.pdf

[55] S. Kramer, “Communication centric design for composability & data consistency in automotive embedded systems,” Robert Bosch GmbH, 2017. Accessed: Nov. 16, 2020. [Online]. Available: http://rtsl-edge.cs.illinois.edu/CMAAS17/media/talk_3.pdf

[56] A. Hamann, D. Dasari, S. Kramer, M. Pressler, and F. Wurst, “Communication centric design in complex automotive embedded systems,” in *LIPIcs-Leibniz International Proceedings Informatics*, vol. 76, Wadern, Germany: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017, pp. 10:1–10:20.

[57] W. Dubitzky and T. Karacay, “CAN - from its early days to CAN FD,” CAN Newslett., pp. 8–11, 2013.

[58] A. K. Sinha and S. Saurabh, “CAN FD: performance reality,” in *Proc. IEEE 3rd Int. Conf. Comput. Intell. Commun. Technol.*, 2017, pp. 1–6.

[59] *CAN in Automation (CiA)*, “CAN FD - the basic idea,” 2019. Accessed: Oct. 29, 2019. [Online]. Available: https://www.can-cia.org/ru/can-knowledge/can/can-fd/

[60] R. De Andrade, K. N. Hodel, J. F. Justo, A. M. Laganá, M. M. Santos, and Z. Gu, “Analytical and experimental performance evaluations of CAN-FD bus,” *IEEE Access*, vol. 6, pp. 21287–21295, 2018.

[61] M. Eichhorn, M. Pfannenstein, D. Muhra, and E. Steinbach, “A SOA-based middleware concept for in-vehicle service discovery and device integration,” in *Proc. IEEE Intell. Veh. Symp.*, 2010, pp. 663–669.

[62] OPEN Alliance, “OPEN Alliance website,” 2019. Accessed: Oct. 29, 2019. [Online]. Available: http://www.opensig.org/about/about-open/

[63] T. Kawauchi, A. Iwata, H. Urayama, T. Izumi, K. Takayama, and T. Hagihara, “Physical layer simulation technology for automotive Ethernet,” in *Proc. IEEE CPMT Symp. Japan*, 2019, pp. 107–110.

[64] S. Mortazavi, D. Schleicher, F. Schade, C. Gremzow, and F. Gerfers, “Toward investigation of the multi-gig data transmission up to 5 Gbps in vehicle and corresponding EMC interferences,” in *Proc. IEEE Int. Symp. Electromagn. Compat.*, 2018, pp. 60–65.

[65] OPEN Alliance, “Automotive Ethernet specifications,” 2020. Accessed: Nov. 5, 2020. [Online]. Available: http://www.opensig.org/Automotive-Ethernet-Specifications

[66] IEEE, “Time-sensitive networking task group,” 2017. Accessed: Oct. 29, 2019. [Online]. Available: http://www.ieee802.org/1/pages/tsn.html

[67] IEEE, “Audio video bridging (AVB),” 2013. Accessed: Oct. 29, 2019. [Online]. Available: http://ieee802.org/1/pages/avbridges.html

[68] IEEE, “802.1AS-2020 - timing and synchronization for time-sensitive applications,” 2020. Accessed: Sep. 8, 2020. [Online]. Available: https://standards.ieee.org/standard/802_1AS-2020.html

[69] IEEE, “1588-2008 - IEEE standard for a precision clock synchronization protocol for networked measurement and control systems,” 2008. Accessed: Sep. 8, 2020. [Online]. Available: https://standards.ieee.org/standard/1588-2008.html

[70] IEEE, “802.1Q-2018 - IEEE standard for local and metropolitan area networks-bridges and bridged networks,” 2018. Accessed: Sep. 8, 2020. [Online]. Available: https://standards.ieee.org/standard/802_1Q-2018.html

[71] IEEE, “ 802.1CB - frame replication and elimination for reliability,” 2017. Accessed: Oct. 29, 2019. [Online]. Available: https://1.ieee802.org/tsn/802-1cb/

[72] M. Ziehensack and H. Parmar, “Automotive plug & play - scalability and flexibility for E/E networks,” presented at *2016 IEEE Standards Association (IEEE SA) Ethernet IP, Automotive Technology Day*, Paris, Sep. 2016. Accessed: Feb. 12, 2021. [Online]. Available: http://docplayer.net/47791418-Automotive-plug-play.html

[73] SAE International, Time-triggered Ethernet, *SAE Standard AS6802*, 2011.

[74] V. Gavriluţ and P. Pop, "Scheduling in time sensitive networks (TSN) for mixed-criticality industrial applications," in *Proc. 14th IEEE Int. Workshop Factory Commun. Syst.*, 2018, pp. 1–4.
[75] L. Zhao, F. He, E. Li, and J. Lu, "Comparison of time sensitive networking (TSN) and TTEthernet," in *Proc. IEEE/AIAA 37th Digit. Avionics Syst. Conf.*, 2018, pp. 1–7.
[76] AUTOSAR Consortium, "The AUTOSAR consortium website," 2019. Accessed: Oct. 29, 2019. [Online]. Available: https://www.autosar.org/
[77] AUTOSAR Consortium, "AUTOSAR introduction," Jul. 2018. Accessed: Oct. 29, 2019. [Online]. Available: https://www.autosar.org/fileadmin/ABOUT/AUTOSAR_Introduction.pdf
[78] AUTOSAR Consortium, "AUTOSAR Classic platform," 2019. Accessed: Oct. 29, 2019. [Online]. Available: https://www.autosar.org/standards/classic-platform/
[79] AUTOSAR Consortium,, "Technical overview, version 2.0.1," 2006. Accessed: Oct. 29, 2019. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/2-0/Main.zip
[80] AUTOSAR Consortium, "Layered software architecture, version 4.3.1," 2017. Accessed: Feb. 12, 2021. [Online]. Available: https://www.autosar.org/standards/classic-platform/classic-platform-431
[81] AUTOSAR Consortium,, "Overview of functional safety measures in AUTOSAR, version 4.3.0," 2016. Accessed: Feb. 12, 2021. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/4-3/AUTOSAR_EXP_FunctionalSafetyMeasures.pdf
[82] D. Haworth, T. Jordan, A. Mattausch, and A. Much, "Freedom from interference for AUTOSAR-based ECUs: A. partitioned AUTOSAR stack," *Automot.-Saf. Secur.*, vol. 210, pp. 85–98, 2012.
[83] AUTOSAR Consortium, "Guide to BSW distribution, version 4.3.1," 2017. Accessed: Feb. 12, 2021. [Online]. Available: https://www.autosar.org/standards/classic-platform/classic-platform-431
[84] AUTOSAR Consortium, "Guide to multi-core systems, version 4.1.3," 2014. Accessed: Feb. 12, 2021. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/classic/4-1/AUTOSAR_EXP_MultiCoreGuide.pdf
[85] M. Panić, S. Kehr, E. Quiñones, B. Boddecker, J. Abella, and F. J. Cazorla, "RunPar: An allocation algorithm for automotive applications exploiting runnable parallelism in multicores," in *Proc. IEEE Int. Conf. on Hardware/Softw. Codesign Syst. Synth. (CODES ISSS)*, 2014, pp. 1–10.
[86] H. R. Faragardi, B. Lisper, K. Sandström, and T. Nolte, "A communication-aware solution framework for mapping AUTOSAR runnables on multi-core systems," in *Proc. IEEE Emerg. Technol. Factory Automat.*, 2014, pp. 1–9.
[87] AUTOSAR Consortium, "Explanation of Adaptive platform design, AUTOSAR Adaptive platform, release 17–10," 2017. Accessed: Sep. 9, 2020. [Online]. Available: https://www.autosar.org/fileadmin/user_upload/standards/adaptive/17-10/AUTOSAR_EXP_PlatformDesign.pdf
[88] AUTOSAR Consortium, "AUTOSAR Foundation 1.5.1," 2019. Accessed: Sep. 9, 2020. [Online]. Available: https://www.autosar.org/standards/foundation/foundation-151/
[89] A. Kritikakou, T. Marty, C. Pagetti, C. Rochange, M. Lauer, and M. Roy, "Multiplexing adaptive with classic AUTOSAR? Adaptive software control to increase resource utilization in mixed-critical systems," in *Proc. Workshop CARS 2016 - Crit. Automot. Appl.: Robustness Saf.*, Göteborg, Sweden, Sep. 2016. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01375576

Victor Bandur received the B.Sc. and M.A.Sc. degrees in software engineering from McMaster University, Hamilton, ON, Canada, in 2006 and 2008, respectively, and the Ph.D. degree in computer science from the University of York, York, U.K., in 2016. He is currently a Postdoctoral Fellow with McMaster Centre for Software Certification (McSCert). He was a Postdoctoral Fellow with Aarhus University, Aarhus, Denmark, and the University of York, where he worked on the automatic code generation from formal software specifications, and on formal methods for software assurance cases. His primary research interest include the use of formal methods in software specification, development, and assurance. He is currently involved in the functional safety research for automotive software.

Gehan Selim was born in Cairo, Egypt, in 1984. She received the B.S. and M.S. degree in information technology from the Faculty of Computers and Information, Cairo University, Cairo, Egypt, in May 2004 and September 2008, respectively, and the Ph.D. degree in software engineering from the School of Computing, Queen's University, Kingston, ON, Canada, in June 2015. From September 2015 to August 2016, she was a Postdoctoral Fellow and an Instructor with the School of Computing, Queen's University. From September 2016 to December 2019, she was a Postdoctoral Fellow with the Department of Computing and Software, McMaster University, Hamilton, ON, Canada. Since January 2020, she has been a Research Associate with the Department of Computer Science, University of Toronto, Toronto, ON, Canada. Her research interests include model-driven software development (MDSD), and previously included testing and formal verification of model transformations, applications of MDSD in the automotive industry, and development of tools to support MDSD. Her research interests include safety, security, and privacy assurance of safety-critical software.

Vera Pantelic received the B.Eng. degree in electrical engineering from the University of Belgrade, Belgrade, Serbia, in 2001, and the M.A.Sc. and Ph.D. degrees in software engineering from McMaster University, Hamilton, ON, Canada, in 2005 and 2011, respectively. She is currently a Senior Principal Research Engineer with the McMaster Centre for Software Certification and McMaster Institute for Automotive Research and Technology (MacAUTO), McMaster University. She manages industrial collaborations with automotive companies focusing on the embedded software development and functional safety. She is also an Adjunct Assistant Professor with the Department of Computing and Software, McMaster University. Her research interests include the development and certification of safety-critical software systems, model-based design, and supervisory control of discrete event systems.

Mark Lawford (Senior Member, IEEE) received the B.Sc. degree in engineering mathematics from Queen's University, Kingston, ON, Canada, in 1989, and the M.A.Sc. and Ph.D. degrees from the Systems Control Group, Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, Canada, in 1992 and 1997, respectively. He is currently a Professor and the Chair with the Department of Computing and Software, McMaster University, Hamilton, ON, Canada, and was the former Director with the McMaster Centre for Software Certification. He was with Ontario Hydro as a real-time Software Verification Consultant on the Darlington Nuclear Generating Station Shutdown Systems Redesign project. In 1998, he joined the Department of Computing and Software, McMaster University, where he helped to develop the Software Engineering programs and Mechatronics Engineering programs. Since 2012, he has been involved in automotive software research. His research interests include software certification, formal methods for safety critical real-time systems, MBD of automotive software, and functional safety of autonomous vehicles. In 2014, he was the co-recipient of a Chrysler Innovation Award, and was the recipient of the University Medal in Engineering Mathematics from Queen's University and the Ontario Hydro New Technology Award for Automation of Systematic Design Verification of Safety Critical Software in 1999. He is currently the Lead Investigator or Co-Investigator on several projects with OEMs and is the CAVs Faculty Advisor for the McMaster EcoCAR Mobility Challenge team. He is a licensed Professional Engineer in the province of Ontario, Canada.