# Architecture Exploration for Distributed Embedded Systems: A Gap Analysis in Automotive Domain

Xinhai Zhang[1], Naveen Mohan[1], Martin Törngren[1], Jakob Axelsson[2], De-Jiu Chen[1]

Department of Machine Design[1]
KTH - Royal Institute of Technology, Stockholm, Sweden
Email: (xinhai,naveenm,martint,chendj)@kth.se

Software and Systems Engineering Laboratory[2]
Swedish Institute of Computer Science (SICS)
Email: jakob.axelsson@ri.se

***Abstract*—A large body of work can be found in literature on Design Space Exploration (DSE) methods for distributed embedded system architecting (DESA). However, almost none of these methods is successfully adopted in automotive industry. To clarify the reasons, this paper 1) analyzes the current state of the art (SOTA) on DSE methods for DESA through a systematic literature study, focusing on the assumed architecting process and concerns; 2) investigates the state of practice (SOP) on DESA in the automotive industry through a literature study and interviews with experienced system architects from five different automotive manufacturers; and 3) analyzes the gap between SOTA and SOP, and thereby discusses potential improvements of DSE methods.**

## I. Introduction

With the rapid evolution of automotive embedded systems, the architecture design is becoming increasingly complex. Examples of the key architectural decisions include (1) how to allocate software functions to distributed hardware platform during evolutionary development; and (2) how to allocate software functions to, while simultaneously forming, a distributed hardware platform during revolutionary development; [1]. Such decisions can be critical since an improper architectural change might introduce unnecessary cost, deteriorate system performance, prevent further system evolution and even increase the time to market.

Academia also realized this problem and research has investigated ways to provide automated support for distributed embedded system architecting (DESA) for more than 10 years. A large amount of methods for Design Space Exploration (DSE) have been published to solve this problem. However, these methods are still not well adopted in the automotive industry, with the reasons remaining unclear.

In order to clarify the reasons, this paper proposes the following research question. **How well do the current DSE methods reflect the practical architecting context?** The **architecting context** is defined by the decision making process and the architectural concerns from all the stakeholders. To answer this research question, the major contribution of this paper is a gap analysis between the current state of the art (SOTA) on DSE methods for DESA and the corresponding state of practice (SOP) in the automotive industry. The gaps were investigated through the following two studies:

- SOTA study on DSE methods for DESA, with special focus on the assumptions made to the architecting context and the potential DESA methodology in the future. (Section III)
- SOP study on DESA in automotive industry, along with gap analysis between SOTA and SOP as shown in Fig. 1. (Sections IV)

The methodologies to conduct these studies are described in Section II. The overall findings and the threats to validity of the gap analysis are discussed in Section V. Section VI concludes this work and suggests future works.

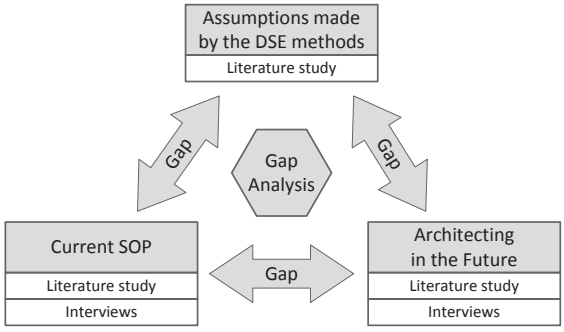


Fig. 1. Outline of this paper

## II. Methodologies

This section describe the research methodologies conducted in the SOTA and SOP studies for automotive DESA.

### A. SOTA Study Methodology

To answer the research question, a systematic literature study on the SOTA of DSE methods for DESA was conducted with special focus on their assumptions about the architecting context. Other concerns such as the computation efficiency of the exploration algorithm, the availability of the DSE tools and the training of the architects are not explicitly covered in this literature study.

*1) Inclusion and Exclusion Criteria:* This literature study was only interested in the DSE methods that provide automated decision support to the complex architecting problem. Therefore, studied work should have a mathematical formulation of the architecting problem so that it can be automatically solved.

The system under design should correspond to a distributed embedded system as the ones adopted in modern vehicles, where multiple computation nodes are connected by one or more communication buses, such as CAN, FlexRay or TTEthernet. Even though optimizations for system-on-chip or multi-core systems [2] are making similar architectural decisions, they were considered out of the scope since such decisions are very likely made in a totally different context (e.g., different scale of the design space and different concerns). Studied work should also consider both software and hardware. Pure software optimization such as the DSE methods for control and scheduling co-design [3] were not covered. For hardware, we are only looking at how to allocate software to general purpose processors, but not how to deploy algorithms on FPGA. Papers that only optimize on ECU level (e.g., only optimize the task scheduling or function to task mapping within one ECU) or only optimize the communication (e.g., only decide signal packing or message scheduling) were also excluded.

In addition, this study only considers the design time optimization for closed systems whose behavior is considered to be predictable. Run-time task allocation methods for generic distributed systems as surveyed by Jiang [4] are out of the scope of this work.

*2) Related Survey:* The latest relevant literature review paper [5] in the area of software architecture optimization systematically surveyed 188 papers published between 1992 and 2011. Out of these 188 papers, 85 were in the embedded system architecting domain and 59 considered the software to hardware allocation problem. Therefore, [5] is considered as a relevant survey and its taxonomy of search-based architecture optimization problems is reused in this paper.

*3) Paper Selection:* As a complement to [5] on distributed embedded systems criteria, relevant papers published between 2012 and 2016 are in addition studied in this paper. The Google Scholar database was used as the only source for this complementary study, wherein papers were searched according to all the different combinations of the selected keywords automotive / CAN-based / distributed embedded systems, design space exploration / architecture exploration / architecture optimization, system-level synthesis and function / software component allocation / mapping. Backward and forward citations were also considered for the papers selected, to achieve greater coverage of the area.

Due to the overlap between the papers, not all the studied papers were used to present the SOTA. Representative ones were selected from all the studied papers according to the following principles.

- Papers that considered more architectural aspects (e.g., more quality attributes considered or more architectural decisions made) were preferred over others that specialized in fewer aspects.
- For incremental work from similar authors with similar architectural concerns, the paper that has more comprehensive description of these architectural concerns was selected.
- For work with similar considerations regarding architectural concerns, the ones that encompassed automotive usecases were prioritized. Amongst the prioritized papers, the latest one was selected.
- Papers studied in [5] with unique architectural concerns were also selected.

Accordingly, 20 papers [6]–[26] were selected to depict the SOTA of DSE methods for DESA in Section III.

### B. SOP Study Methodologies

To answer what practical architecting context is in automotive industry, a literature study and interviews with practitioners were conducted in parallel.

*1) Literature Study:* Google Scholar was used as the database to search the papers published between 2005 to 2016 and describing the SOP of DESA in automotive industry. Key words used for the search include Architecting, software Engineering, automotive, distributed embedded systems and industrial practice. Nine articles [1], [27]–[34] were selected to study the SOP.

*2) Interviews with Experienced Architects:* To supplement the SOP depicted by the literature, five semi-structured interviews were carried out with experienced (i.e., more than five years relevant working experience) system architects from different automotive Original Equipment Manufacturers (OEMs) in Sweden and China. The interviewer was the first author of this paper. The length of each interview was between one hour to two hours. The list of questions used for the interviews was designed according to the results of the literature studies for SOTA and SOP. It contains questions from five categories: automotive embedded system architecture, architecting process (including methodology, scenarios and stakeholders), architecting concerns, expected tool support and outlook of AUTOSAR.

## III. SOTA ON DSE METHODS FOR DESA

Based on the methodology introduced in Section II-A, the architecting context assumed by the studied DSE methods is analyzed in this section.

According to the taxonomy from [5], the SOTA of the DSE methods is analyzed in terms of 1) the degrees of freedom (i.e., the architectural decisions that can be made by the DSE methods), 2) the considered quality attributes (objectives and constraints of the DSE methods) and 3) the required information (i.e., the inputs of the DSE methods) to support the decision making and the evaluation of the considered quality attributes. As mentioned before, architecting context includes the decision making process and architectural concerns. It is assumed that the degrees of freedom and the required information of the DSE methods reflects the assumptions made about the decision making process, and that the models of the quality attributes reflects the assumptions made about the architectural concerns.

To distinguish between generic architecting context for distributed systems and the context specifically for automotive use case, the rest of this section is divided into two parts.

Section III-A describes the generic architecting context which is a common subset of the architecting context assumed by most of the studied DSE methods. Section III-B categorizes the automotive specific concerns covered by different DSE methods.

### A. Generic Context

Most of the studied DSE methods can be represented by Fig. 2. On the top of the DSE block, there are two elements that mainly controls the DSE process. The design of the DSE algorithm only controls the trade-off between the computation efficiency of the exploration and the optimality of the result. Quality attributes are normally formulated as objectives or constraints in DSE methods. Detailed descriptions of the inputs, outputs and quality attributes of the DSE methods are provided in the rest of this section.

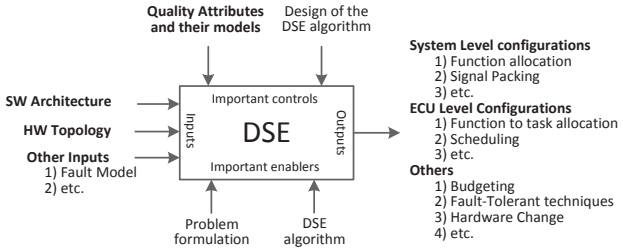


Fig. 2. The input-output view [28] of the DSE process

*1) Degrees of freedom:* Most of the studied DSE methods make multiple architectural decisions together, on both system level and ECU level. System level decisions include function to ECU allocation (All the selected papers considered this.), ECU to sub-network allocation [13], selection and addition of ECUs [17], [18], signal to message packing [11], [12], [26] and message scheduling [11], [12], [26]. ECU level decisions include function to task allocation [11], [12], task scheduling [8]–[12], [24]–[26] and task partitioning for the sake of reliability [8], [23] or energy saving [6], [7], [24].

*2) Quality Attributes:* The most commonly considered quality attributes in the SOTA include feasibility, performance, cost and reliability. Concerns regarding feasibility include resource constraints (almost all the methods consider memory and CPU utilization), schedulability of tasks and messages [6]–[15], [23]–[26] and physical constraints (e.g., some functions can only be allocated on particular ECUs) [11]–[14], [17], [23]. Performance is usually defined in terms of End-to-End (E2E) timing [10]–[14], [26]. Considerations regarding material and development cost can be found in [17], [18], [20], [22], [23].

In the studied literature, reliability is considered from three aspects, namely reliability block diagram [15], fault tree [35] and fault tolerance [8]–[10]. Faragardi et al. [15] defined reliability using the failure rates of hardware, i.e., the processors and the communication links between two ECUs. The reliability of a function chain is determined by the failure rates of all the hardware elements (ECUs or communication links) that carries this function chain. Walker et al. [35] maximized the reliability of the distributed system by introducing an external tool Hip-HOPS[1] into the DSE process for fault tree analysis. They assumed that the failures of the ECU will propagate to the software functions allocated on it. Given predefined fault trees of the software and the hardware platform including power system, their method provides an optimal allocation with minimal causes of system failures.

Another set of DSE methods (e.g., [8]–[10]) address the selection of proper hardening techniques to handle transient faults or soft errors, i.e., the unpredictable hardware failures caused by environmental phenomena, such as radiations or electromagnetic interferences. Pure hardware hardening techniques are normally expensive. Software solutions that rely on ordinary hardware will provide performance and resource penalties, bringing trade-offs to be made. To optimize system performance while guaranteeing the reliability of critical functions, DSE methods were proposed to select proper hardening techniques together with the decisions regarding function allocation. Considered fault detection techniques include explicit output comparison with replication [8]–[10] and concurrent error detection (need external hardware support) [8]–[10]. Examples of fault tolerance techniques are re-execution [8]–[10], rollback recovery (need external hardware support) [8], [9] and voting [8].

*3) Required Information:* All the studied DSE methods request the models of predefined software architecture (i.e., a graph of interactive software components (SW-Cs)) and hardware topology (i.e., the set of ECUs and their connections) as input. Most of the DSE methods assumed that the software architecture and the hardware topology are designed separately. The former is designed based on the required functionalities, and the latter is designed based on the shape of the vehicle, the positions of the sensors and actuators, the length of cables and successful legacy hardware topologies [34]. With the help of the method introduced in [36], customized (formally described) quality attributes can also be one of the inputs of the DSE process.

### B. Automotive Domain Specific Context

Beside these generic architectural decisions and quality attributes, a large amount of problem formulations of the studied DSE methods also consider the particular characteristics of automotive industry. These automotive domain specific concerns are categorized as follows.

*1) AUTOSAR:* According to AUTOSAR specifications, software functions are implemented by a set of software components (SW-Cs) in the application layer of AUTOSAR architecture. Typically, many of the application level SW-Cs are relatively ECU-independent, meaning that they can be freely moved from one ECU to another. Each SW-C comprises one or multiple runnables which are the atomic execution units. Within each ECU, runnables from different SW-Cs can be partitioned into one or more tasks, which are scheduled

[1]Hip-HOPS: http://hip-hops.eu/

by the real-time operating system. On the communication bus, signals transmitted between ECUs can be partitioned into messages, which are scheduled according to the communication protocol. In [12], DSE support for system level and ECU level design decisions is defined as a PPS (Placement, Partitioning and Scheduling) problem, i.e., to place SW-Cs to ECUs, to partition runnables and signals into tasks and messages, and to schedule tasks and messages. Studied papers whose problem formulations comply with AUTOSAR specifications include [11], [12], [14], [16], [23].

*2) Hardware Topology:* A modern vehicle normally has more than 70 interconnected ECUs physically distributed to different parts of the vehicle, and this number is still increasing [31]. Some of the studied DSE methods considered complicated hardware topologies (i.e., how ECUs are connected with each other), for example, [11]–[13]. In their formulations, multiple CAN buses can be gatewayed by one coordinator and sub-buses can be used to directly connect two ECUs from different CAN buses [13].

*3) Architecting Scenarios:* Most of the DSE methods were proposed for a revolutionary architecting process where a brand new architecture needs to be designed (e.g., [12], [13]). Zhang et al. [14] proposed a DSE method for evolutionary architecting where new functions need to be added to an existing architecture.

Architectural decisions are normally made at an early stage of development, where not too much concrete information is available such as the worst case execution time (WCET) of a runnable. When allocating functions to ECUs, Wozniak et al. [11] assumed that detailed timing properties are only known for the reused SW-Cs. For the SW-Cs belonging to new functions that need to be integrated into an existing architecture, their method provides maximum time budgeting while guaranteeing schedulability.

*4) Suppliers:* Automotive DESA also largely involves the interactions with suppliers. Deng et al. [16] considered the collaboration with TIER 1 suppliers, who provide entire subsystems or only SW-Cs to the OEMs. Besides the architecting work that need to be conducted by the OEMs, their method also helps the suppliers to generate runnables from Simulink models, optimizing modularity and reusability.

*5) Flexibility:* The rapid evolution of automotive embedded systems entails the system architecture to be flexible for future changes. Given a set of estimated changing scenarios, the flexibility of the baseline architecture can be optimized by minimizing the architectural modifications required to meet the estimated scenarios [37]. Without a clear prediction for the future, flexibility can also be optimized from the scheduling perspective, i.e., to maximize the slack time between the worst case response times and the deadlines of all the tasks [13] or messages [12].

*6) Product Line and Variability:* To satisfy customers with different requirements, OEMs normally define a series of slightly different vehicle models (each vehicle model is called a variant) under one product line. Therefore, instead of optimizing one particular system architecture, architects usually need to consider all the variants of the whole product line. One key concern regarding variability is to control the architectural differences over all the variants so as to reduce the effort for verification, manufacture and maintenance. Graf et al. [18]–[20] proposed a DSE method to maximize the number of ECUs that can be reused over multiple variants. To increase the reusability of one ECU, they assumed that when the ECU is used for different variants, different subsets of its SW-Cs are activated. This SW-C activity configuration is called one manifestation of that ECU. The number of manifestations of one ECU should also be minimized during the optimization to increase the utilization of that ECU [19]. The schedule of tasks on ECUs with multiple manifestations can be designed by the method introduced in [38]. In addition, when optimizing the whole product line, estimated sales volumes of different variants should also be considered together with the uncertainty of the estimation [20].

*7) Operational Modes:* Vehicle operational modes (e.g., parking or driving) are in the found literature linked to energy saving in DSE methods [6], [7]. Not all the functions need to be active in all operational modes. For example, cruise control is not needed in parking mode. Katoen et al. [7] assumed for run-time operation that the current mode configuration (e.g., driving & entertainment on) determines the activities of different function chains (i.e., a chain of dependent SW-Cs). In terms of energy consumption, they assumed that 1) different SW-Cs have different energy consumption on different ECUs; and 2) if no SW-C is active on one ECU in current mode configuration, this ECU can be turned off to save energy. Given a predetermined usage profile (i.e., a Markov chain model of the mode transition), their DSE method provides an energy-optimal function allocation. Walla et al. [6] also considered the energy consumption of inter ECU communication. Besides, they also assumed that an ECU can be partly turned off by only turning off some partitions of that ECU.

*8) Functional Safety:* Safety is a key quality in the automotive industry. However, guaranteeing safety is becoming complicated due to the increasing complexity of the automotive embedded system. Standards like ISO 26262 bring best practices in the form of a development methodology, with safety concerns covering the whole life cycle. These standards have also been considered in the DSE methods for DESA, for example, [21]–[23]. Schätz et al. [21] described the necessity and possibility to consider ISO 26262 in the DSE methods. Safety critical functions and their implementations (including SW-Cs and ECUs) are assigned with different automotive safety integrity levels (ASILs) according to their safety criticality. ASIL D refers to the highest criticality level and QM refers to non-critical functions. Temporal and spacial partitioning techniques are also suggested by the standard to isolate mix criticality tasks on the same ECU. In this way, given the ASILs of all the SW-Cs, DSE methods can also help to decide the allocation of SW-Cs to partitions [22], [23]. Further more, same functions with higher ASIL request much more costly development. To reduce development cost, functions assigned a high ASIL can be decomposed into

multiple redundant ones, assuming they are independent [21]. For example, an ASIL D function can be decomposed into two ASIL B functions or an ASIL C function plus an ASIL A function. Given all the alternative ASIL decomposition strategies and estimated development costs for all the functions with different ASILs, Tămaş-Selicean et al. [22] proposed a DSE method to also suggest the optimal (in terms of development cost) decomposition of high-ASIL SW-Cs into redundant lower-ASIL SW-Cs.

*9) Security:* Techniques to improve security can also be considered during the architecting DSE process, such as cryptography [24], [25], run-time intrusion detection [24] and Message authentication codes (MACs) [26]. Jiang et al. [25] assumed that each message transmitted online need to be encrypted (by the sender) and decrypted (by the receivers) by individual tasks which introduce computation overhead. They further assumed that stronger security protection requests longer execution time of the encryption and the decryption. Given resource and energy constraints, their DSE method optimize the average security strength of all the messages by deciding the security level of each encryption together with function allocation. Jiang et al. [24] also added one intrusion detection task to each ECU to monitor the communication, they also assumed that higher detection accuracy requests higher execution time. Lin et al. [26] applied MACs (sender identification + counter) to all the messages to protect against the masquerade attack, where a hacked ECU sends a message that claims to be sent from another ECU, and replay attack, where a hacked ECU echos the message it has just received. They assumed that longer MACs in a message guarantee higher security but also reduces the number of signals that the message can carry and introduce higher transmission latency. They also assumed that different ECUs had different risks to be attacked. Subject to the timing and resource constraints, their method minimizes security risk.

## IV. SOP and Gap Analysis

This section presents a short introduction of the architecting process in the automotive industry based on the methodology introduced in Section II-B, before detailing the gaps between the SOTA and SOP across the chosen dimensions.

In practice, architecting is an iterative negotiation process between the architects and different stakeholders such as the marketing department, financial department, the function owners and the system owners. As an example of a common organization of roles in the automotive industry, function owners manage the high level specification of the functions, whereas system owners are responsible for the development or the procurement of a subsystem or a single ECU. Detailed descriptions of function owners and system owners can be found in [39].

Two interviewed architects stated that the design of subsystems or particular ECUs normally comes after function allocation. When a preliminary function allocation is decided by the architects, system owners are responsible for the design, implementation and configuration of the ECUs, such as the decisions on which functions should be implemented by hardware, the selection of the processor and the scheduling of the tasks on the operating system level.

To facilitate the comparison between the practical architecting process in automotive industry and the DSE methods, the industrial process is also abstracted and represented in a input-output view (inspired by [28]), as shown in Fig. 3. Budget constraints normally include monetary budget and time to market budget. The experience of the architects includes the considered architectural quality attributes and the way to leverage them. The rest of this section explains Fig 3 and points out the gaps between SOTA and SOP.

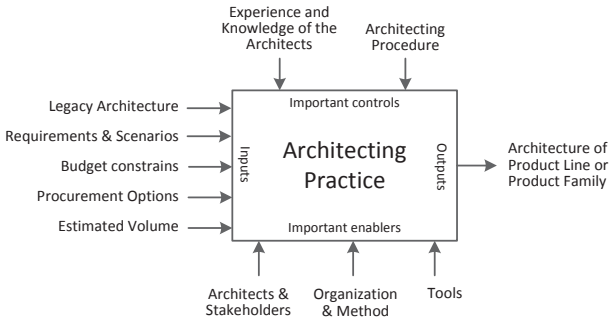


Fig. 3. The input-output view of the industrial architecting process

### A. Architecting Scenarios

In the automotive domain, architecting scenarios are classified as 1) evolutionary architecting (EA) where incremental changes are made to an existing architecture, and 2) revolutionary architecting (RA) where a brand new architecture is designed almost from scratch [1].

An evolution oriented DSE method can be found in [14], however it can only decide function allocation. A real evolutionary architecting design case is given in [29] as shown in Fig. 4, where two design alternatives have been identified after a pre-study. The key decision left is in which ECU the new SW-C sould be allocated. Design alternative (a) allocates the new SW-C into the cabin gateway ECU (CG ECU) and directly connects the external communication link to the CG ECU through the only available communication interface left on the CG ECU. The major advantage of this design is the low material and development cost. Design alternative (b) allocates the new SW-C to an added ECU, which communicates with the CG ECU through the CAN bus. Though the design is more expensive, it saves the last communication interface of CG ECU for future flexibility. The tradeoff between unit cost and system flexibility for this case study is comprehensively discussed in [29] with a proposed cost model.

*1) Gaps:* An important finding is that most of the studied DSE methods focus on supporting RA scenarios, which represent the worst case in terms of the largest design space. However in practice, RA scenarios (once in 5-10 years) are far less frequent than EA scenarios (once in 1-2 years) [1]. Making a distinction between EA and RA scenarios is

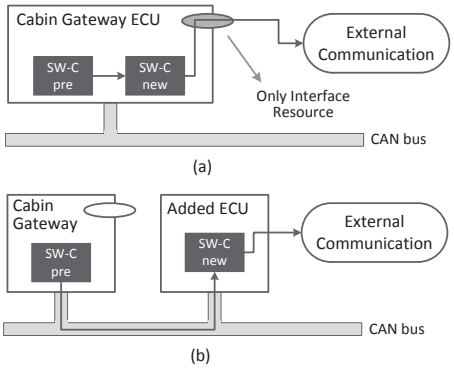


Fig. 4. Two design alternatives of a real case from automotive industry [29]

necessary for the DSE methods since these scenarios imply different architecting contexts [1].

*2) Challenges:* EA scenarios rely to a large extent on already available hardware and software. The challenge here is to be able to capture this existing information and use it for DSE. The existing information may be tacit, informal or represented by unconnected and incomplete pieces of formalized information. Reverse engineering efforts may represent partial solutions [40].

RA scenarios by nature often involve changes to the hardware topology to such a degree that a lot of uncertainty and many options arise, for the architects to deal with. The functional requirements may also naturally be unclear when the RA involves new functionality [41]. Thus detailed information that most DSE methods need, like the execution time of a task, is mostly not available in RA scenarios. Additionally, for RA scenarios in the industry, the system under design is usually not the architecture of a particular vehicle model but the architecture of the product line or the whole product family [27], [30]. Detailed gap analysis regarding product family and system variability is given in Section IV-F.

### B. Degrees of Freedom

According to [1], [27] and the interviews, the architecture refers to the relationships between sensors, actuator, ECUs, software, communication buses, power supply and cabling.

As analyzed in Section III-A1, architectural decisions that can be made by the studied DSE methods contain SW-C to ECU allocation, configurations on ECUs and the communication buses, addition and reduction of ECUs, selection of hardening and security techniques, budgeting for the SW-Cs under development, SW-C partition, and ASIL decomposition.

*1) Gaps:* Almost all of DSE methods also assume a fixed network topology and do not consider available physical interfaces (to sensors, actuators or communication buses) as one type of resource of ECUs. For a modern vehicle, this is an oversimplifying assumption as these connections directly affect the allocation of the SW-Cs related to the physical world. Besides, any ECU with multiple communication interfaces can be modified into a gateway to exchange messages among all the connected communication buses. If a subsystem needs to communicate with the CAN bus in the example illustrated in Fig. 4 (b), instead of directly connecting the subsystem to the CAN bus, it could be for example connected to the CG gateway. These solutions are not accessible to current DSE methods.

In addition, procurement strategy is another major decision that needs to be made by the architects during function allocation as discussed in Section IV-E. When the hardware topology need to be changed, cabling is also an important concern since it is directly linked to material cost.

*2) Challenges:* The generality of the DSE methods is thus restricted by the assumption of fixed hardware topology and lack of support for modeling free and occupied communication interfaces. In order to consider the changes on the hardware topology in DSE, different physical interfaces (e.g., GPIO, ADC, DAC and CAN) of ECUs need to be modeled as one dimension of the ECU resource and different connection rules also need to be formulated as constraints.

### C. Considered Quality Attributes

In the interviews and in [27], [29], cost and flexibility were identified as the most important quality attributes during system architecting. The interviewed architects also emphasized backward compatibility, time to market, resource utilization, security and safety. Analysis regarding cost and backward compatibility is provided in Section IV-D. Safety is discussed in Section IV-G. The rest of these quality attributes are analyzed in the following subsection.

*1) Gaps:* There is no DSE method that considers all the quality attributes that are considered by the practitioners. Furthermore, some of the quality attributes have not been considered by any of the studied DSE methods.

Revealed by section III-A, resource constraints considered by the DSE methods only include bandwidth utilization of the communication bus, the CPU and memory of the ECUs and the energy consumption. As mentioned in IV-B, one missing but important dimension to define an ECU is its physical interfaces, implying that the models of flexibility in these methods are largely incomplete. Such interface resources are also ignored by the DSE methods when considering flexibility, as shown in the example in Fig 4.

When discussing timing requirements with the architects, the first concept that came to their mind is the time to market but not the E2E latency. As mentioned by one of the architects in the interviews, the time to market requirement is prioritized, in some cases, even over some of the functional requirements. If the planned addition of some functions cannot satisfy the time to market deadline, the addition might be postponed to the next generation.

The E2E timing requirements are usually addressed with E2E timing budgets allocated to different ECUs. As analyzed in Section III-B, this kind of time budgeting is only considered by the DSE method in [11].

None of the architects mentioned a quantitative way to consider security at the architecture level, but most of them mentioned the heuristic to separate safety critical functions

from external communications and this separation can be easily achieved, owing to the distributed nature of the system.

*2) Challenges:* In practice, quality attributes are normally evaluated qualitatively by the architects based on their experience. Most of these quality attributes are difficult to quantify and prioritize due to the uncertainties from different perspectives. Even though some statistic models can be proposed to estimate for example the time to market, it is also challenging to integrate this model into the current DSE problem formulation. Another way to optimize some quality attributes indirectly is to follow the corresponding architecting heuristics. However, even if these heuristics can be easily formulated, it is still challenging to prioritize different heuristics in the formulation.

### D. Cost

Cost represents an important concern during system architecting, roughly including material cost, development cost, manufacture cost and after-market cost. Cost is even more important for OEMs of commercial vehicles, for which, price is one of the key competences.

Architects need to estimate the cost for different design alternatives based on the historical statistics and their experience. Detailed studies of cost models in the automotive industry can be found in [29], [32], [33].

A more common way in practice to reduce cost is to optimize the cost related quality attributes. Some of these attributes are also considered by the DSE methods. For example, reusability [18]–[20] and flexibility [12], [13], [37]. Cost related heuristics have also been considered by DSE methods, such as to reduce the number of ECUs, [17], [18], [20] and ASIL decomposition to reduce verification cost [22], [23].

*1) Gaps:* Besides a comprehensive cost model, backward compatibility, which is an important quality attribute to limit after-market cost, has also not been explicitly considered in the DSE methods for DESA. Keeping most of the ECUs backward compatible will significantly reduce the effort for maintenance.

*2) Challenges:* It is difficult to estimate the cost precisely. It is also challenging to integrate the not linear cost models into the current DSE algorithms, which commonly request linear constraints and objectives. To consider the cost related quality attributes is easier to implement and also close to the industrial practice since these attributes come from the experience of the architects. Therefore, the DSE designers need to have a deep understanding of the architecting process and formulate these cost-related quality attributes properly.

### E. Procurement Strategy

OEMs based on their business needs may purchase software, hardware, complete ECUs or multi-ECU subsystems designed for a particular function. Different OEMs have different percentage of in-house developed software. [27]. Among the OEMs that we interviewed, the highest has around 50% to 60% of in house software, while the lowest is a small OEM who has almost no software developed in-house. The other OEMs have around 10% to 20% percent software developed in-house.

*1) Gaps:* Unlike the assumption made in [16], it is very uncommon to only purchase SW-Cs. It may become common in the future because of the increasing adoption of AUTOSAR, but nowadays, procurements are normally conducted on the ECU level or subsystem level.

According to [27] and also the interviews, it is possible but expensive to add new functions to a supplied ECU or subsystem. To do this, the OEM needs to pay expensive extra development costs to the supplier, further challenging the assumption of free allocation of SW to HW that most DSE papers make.

Therefore, the freedom to re-allocate SW-Cs from or to a supplied ECU is restricted since realizing this will be expensive (valid for most ECUs). If function allocation is only considered for the in-house developed ECUs, the design space will be largely reduced. However, the procurement strategy brings another degree of freedom for the DSE. That is which supplied subsystems should be purchased, and how they should be modified.

*2) Challenges:* As analyzed above, it is impossible to ignore the interplay with suppliers when designing the DSE method. This procurement strategy brings another degree of freedom for the DSE as the decision on which supplied subsystems/ECUs/functions should be purchased, and how they should be modified.

### F. Product Line and Variability

Most of the DSE methods optimize the architecture of a particular vehicle model, however, in practice, the architects need to manage the variability and to architect the product line and the product family [30].

*1) Gaps:* Studied DSE methods regarding variability [18]–[20] only consider the reusability of ECUs among variants under the same product family. However, the consistency of the messages on the communication buses is not emphasized. Like backward compatibility, the consistency on the communication bus among different variants can also reduce after-market cost related to maintenance. Three interviewed architects mentioned the heuristic to keep the bus consistent among variants by adding a gateway ECU between the communication bus and the variant supplied subsystems as exemplified in Fig 5. Three different gearbox management systems (GMSs) are supplied from different suppliers for three variants of the same product line. Different GMSs request different interfaces to the powertrain CAN bus. A gateway ECU is developed through which all the variant GMSs connect to the CAN bus. The gateway ECU has a fixed interface to the CAN bus so as to keep it consistent for all the variants. The extra price of this solution is the cost of the gateway ECU.

Another design alternative is to integrate the software implementation of this gateway into the three supplied GMSs by outsourcing the development to the suppliers. This design will reduce the development effort of the OEM and also improve the E2E timing performance and reliability since the gateway ECU may fail. The architects need to tradeoff between these two alternatives according to the estimated sales volume, the

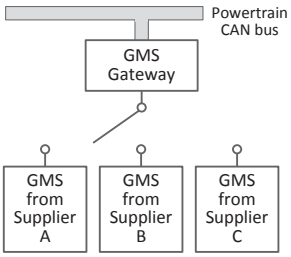


Fig. 5. The gateway ECU keeps the CAN bus consistent for three variants

price of the outsourced development and the results of the impact analysis of both design alternatives.

*2) Challenges:* Only optimizing the architecture for a single vehicle yields a large design space, not to say the whole product family. Meanwhile, exploring the design space for the optimal solution is NP-hard. Therefore, a computationally efficient DSE method is an enabler for DSE methods to architect the product family.

### G. Functional Safety and Verifications

Safety is seen as an emergent vehicle level property and the architects primarily deal with safety at a functional level though they do consider hardware and software implications.

*1) Gaps:* Safety analysis of application design requests complete end-to-end considerations from sensors to actuators. Neglecting the considerations of sensors and actuators, like most of the studied DSE methods do, may lead to suboptimal or invalid solutions. Standards such as ISO 26262 require a hazard at a vehicle level to be addressed across the whole functional chain and not just across the processing units. To illustrate, consider the example topology shown in Fig. 6. ECU 1 and ECU 2 implement two redundant functionalities. The two sensors also provide the same functionality but they have different ASIL assigned. ASIL assignment for the SW-Cs in both ECUs has consider how the sensors are rated by ASIL levels.

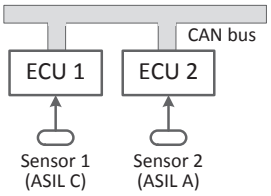


Fig. 6. Example of ASIL decomposition

*2) Challenges:* First, safety attributes have to be added to the models of all the software and hardware components. Second, safety related techniques or architecting heuristics also need to be formulated as part of the DSE problem.

## V. Discussion

Referring to Fig. 1, this section discusses the estimated ideal architecting in the future and summaries the findings about the gaps. The threats to validity of the gap analysis are also discussed at the end of this section.

### A. Outlook of Future Architecting

Discussions of future architecting processes can be found in the vision of AUTOSAR [42] and the published suggestions for architecting methodology [10], [43]–[45].

To provide automated support to the increasingly complex embedded system design, platform based design methodology was proposed in [43], where DSE methods can be embedded into a model-based environment. OEMs should have their own libraries of legacy software and hardware components. Given high level requirements, reusable components will be selected from the libraries and automatically configured by DSE methods. To further formalize this methodology, contract theory is integrated to manage the functional requirements and behavior [44] so as to automate the component selection.

Thanks to AUTOSAR and the adoption of general purpose processing units, software and hardware design will be largely decoupled [10], [45]. The employment of high performance CPUs will significantly reduce the number of ECUs in a vehicle, but the distributed nature of automotive embedded systems will remain due to its advantages [45].

### B. Summary of Findings

According to the interviews and [41], one of the biggest challenges confronted by the architects is uncertainty. However, this is not addressed by almost all the studied DSE methods. On the contrary, most of them request detailed information such as a fixed hardware topology or the WCETs of tasks. In addition, an entire model based design environment is another heavy assumption of the DSE methods. This might be possible in the future, but represents a challenge for current practice.

Evolutionary vs. revolutionary architecting scenarios have not been distinguished by most of the DSE methods. Moreover, in both scenarios, the practitioners normally have more freedom to improve the architecture than the DSE methods assumed, such as to change the hardware topology or to outsource some subsystems to the suppliers.

The quality attributes for architecting can be basically divided into performance-related quality attributes (PRQAs) and cost-related quality attributes (CRQAs). PRQAs are used to evaluate the performance of each individual product architecture. They are normally formulated from extra-functional requirements from other stakeholders such as the function owners. Examples of PRQA are E2E latency, resource utilization, safety and security. Optimizing PRQAs may improve the brand reputation and thereby increase sales volumes. CRQAs refer to the architectural techniques to reduce costs. They normally come from the experience of the architects. As discussed in Section IV-D, examples of CRQA include flexibility, reusability and backward compatibility. Another important quality attribute which belongs to neither PRQA nor CRQA is the time to market. In practice, the architects need to estimate the time to market for all the possible design alternatives based on their experience. However, this is very difficult to be quantified and not covered by any of the studied DSE methods.

In practice, problems regarding E2E latencies and schedulabilities are normally solved by the system owners, when most of the detailed information is available. The DSE methods that consider both system level and ECU level configurations were designed to holistically support both the architects and the system owners. Though this design cannot reflect the current practical architecting process, it enables the possibility to achieve the globally optimal solution. Furthermore, considering the evolution of automotive functionality, including more advanced motion control (e.g., for active safety and autonomous driving), it is also clear that E2E timing analysis will become increasingly important in the future (i.e., the real-time requirements will be increasing).

To summarize, the union set of all the studied DSE methods can largely reflect the practical architecting context. However, different papers present their DSE formulation with different emphasis. A generic platform to integrate all these DSE methods is still missing.

### C. Threats to Validity

The main contribution of this paper is the gap analysis between SOTA and SOP on DESA. Two threats to the validity of the gap analysis are the risk for incompleteness of the understanding of the SOP and the SOTA.

The risk of the incompleteness of the SOP study highly depends on the following three aspects: 1) the comprehensiveness of the SOP literature study, 2) the coverage of the list of questions for the interviews and the 3) representativeness of the interviewees. The risk regarding the first two aspects is mitigated by the semi-structure approach for the interviews. Most of the questions are open so that the interviewees have the freedom to express their experience. The findings from the interview is further validated by the literature study on the same topic. The representativeness of the interviewees is related to the diversity of their affiliated OEMs. The interviewees were from five different OEMs in two countries. These five OEMs are also diverse in terms of sales volumes and percentages of in-house developed software. Some of the interviewees also have experience from various automotive companies.

The incompleteness of the SOTA is highly related to the keyword list adopted for the literature search. To mitigate the risk, the literature search was performed iteratively to evolve the keyword list. In addition, the detected gaps (as described in Section IV) were also added to the keyword list.

## VI. Conclusion and Future Work

In order to figure out the reason why DSE methods for DESA are not well adopted by the automotive industry, this paper investigates the gaps between the architecting context assumed by the DSE methods in literature and the industrial practice. Architecting context is defined by the architecting process and the architectural concerns from relevant stakeholders. For the DSE methods, the inputs and outputs are considered to reflect the assumed architecting process, and the considered quality attributes represents the assumed architectural concerns.

The SOTA of DSE methods was depicted by a systematic literature review. The SOP was achieved through literature study and five semi-structured interviews with experienced architects from different OEMs. Based on the comparison between the SOTA and the SOP, our findings on the suggestions to improve DSE methods were discussed in Section IV.

As future work, the result of this gap analysis should be presented to different experts in this area and also the architects from other automotive OEMs than the interviewed ones so as to achieve a comprehensive requirement analysis of DSE methods for automotive DESA.

## Acknowledgment

The authors would like to thank all the interviewees and their affiliated companies. Support from Vinnova FFI, through ARCHER (proj. No. 2014-06260) and EUREKA TRACE (proj. No. CAT311) is also acknowledged.

## References

[1] J. Axelsson, "Evolutionary architecting of embedded automotive product lines: An industrial case study," in *2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture*. IEEE, sep 2009, pp. 101–110.

[2] M. Belwal and T. S. B. Sudarshan, "A survey on design space exploration for heterogeneous multi-core," in *2014 International Conference on Embedded Systems (ICES)*, no. Ices. IEEE, jul 2014, pp. 80–85.

[3] P. Deng, Q. Zhu, A. Davare, A. Mourikis, X. Liu, and M. D. Natale, "An Efficient Control-Driven Period Optimization Algorithm for Distributed Real-Time Systems," *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3552–3566, dec 2016.

[4] Y. Jiang, "A Survey of Task Allocation and Load Balancing in Distributed Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 585–599, feb 2016.

[5] A. Aleti, B. Buhnova, L. Grunske, A. Koziolek, and I. Meedeniya, "Software architecture optimization methods: A systematic literature review," *IEEE Transactions on Software Engineering*, vol. 39, no. 5, pp. 658–683, may 2013.

[6] G. Walla, A. Herkersdorf, A. S. Enger, A. Barthels, and H.-u. Michel, "An automotive specific MILP model targeting power-aware function partitioning," in *2014 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV)*, no. Samos Xiv. IEEE, jul 2014, pp. 299–306.

[7] J.-P. Katoen, T. Noll, H. Wu, T. Santen, and D. Seifert, "Model-Based Energy Optimization of Automotive Control Systems," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013*. New Jersey: IEEE Conference Publications, 2013, pp. 761–766.

[8] C. Bolchini and A. Miele, "Reliability-Driven System-Level Synthesis for Mixed-Critical Embedded Systems," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2489–2502, dec 2013.

[9] P. Pop, V. Izosimov, P. Eles, and Zebo Peng, "Design Optimization of Time- and Cost-Constrained Fault-Tolerant Embedded Systems With Checkpointing and Replication," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, no. 3, pp. 389–402, mar 2009.

[10] B. Zheng, Y. Gao, Q. Zhu, and S. Gupta, "Analysis and optimization of soft error tolerance strategies for real-time systems," in *2015 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. IEEE, oct 2015, pp. 55–64.

[11] E. Wozniak, M. Di Natale, H. Zeng, C. Mraidha, S. Tucci-Piergiovanni, and S. Gerard, "Assigning time budgets to component functions in the design of time-critical automotive systems," in *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering - ASE '14*. New York, USA: ACM Press, 2014, pp. 235–246.

[12] A. Mehiaoui, E. Wozniak, S. Tucci-Piergiovanni, C. Mraidha, M. Di Natale, H. Zeng, J.-p. Babau, L. Lemarchand, and S. Gerard, "A two-step optimization technique for functions placement, partitioning, and priority assignment in distributed systems," *ACM SIGPLAN Notices*, vol. 48, no. 5, p. 121, may 2013.

[13] Q. Zhu, Y. Yang, M. Natale, E. Scholte, and A. Sangiovanni-Vincentelli, "Optimizing the Software Architecture for Extensibility in Hard Real-Time Distributed Systems," *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 621–636, nov 2010.

[14] X. Zhang, L. Feng, D.-j. Chen, and M. Torngren, "Design-Space Reduction for Architectural Optimization of Automotive Embedded Systems," in *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*. IEEE, aug 2015, pp. 1103–1109.

[15] H. R. Faragardi, R. Shojaee, M. A. Keshtkar, and H. Tabani, "Optimal task allocation for maximizing reliability in distributed real-time systems," in *2013 IEEE/ACIS 12th International Conference on Computer and Information Science (ICIS)*. IEEE, jun 2013, pp. 513–519.

[16] P. Deng, F. Cremona, Q. Zhu, M. Di Natale, and H. Zeng, "A model-based synthesis flow for automotive CPS," *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems - ICCPS '15*, pp. 198–207, 2015.

[17] B. Clark, I. Stierand, and E. Thaden, "Cost-minimal pre-allocation of software tasks under real-time constraints," in *Proceedings of the 2011 ACM Symposium on Research in Applied Computation - RACS '11*. New York, New York, USA: ACM Press, 2011, p. 77.

[18] S. Graf, M. Glas, J. Teich, and C. Lauer, "Multi-variant-based design space exploration for automotive embedded systems," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014*. New Jersey: IEEE Conference Publications, 2014, pp. 1–6.

[19] S. Graf, M. Glass, J. Teich, and C. Lauer, "Design Space Exploration for Automotive E/E Architecture Component Platforms," in *2014 17th Euromicro Conference on Digital System Design*. IEEE, aug 2014, pp. 651–654.

[20] S. Graf, S. Reinhart, M. Glaß, J. Teich, and D. Platte, "Robust design of E/E architecture component platforms," in *Proceedings of the 52nd Annual Design Automation Conference on - DAC '15*. New York, New York, USA: ACM Press, 2015, pp. 1–6.

[21] B. Schätz, S. Voss, and S. Zverlov, "Automating Design-Space Exploration: Optimal Deployment of Automotive SW-Components in an ISO26262 Context," in *Proceedings of the 52nd Annual Design Automation Conference on - DAC '15*. ACM Press, 2015, pp. 1–6.

[22] D. Tămaş-Selicean and P. Pop, "Design Optimization of Mixed-Criticality Real-Time Embedded Systems," *ACM Transactions on Embedded Computing Systems*, vol. 14, no. 3, pp. 1–29, apr 2015.

[23] F. Maticu, P. Pop, C. Axbrink, and M. Islam, "Automatic Functionality Assignment to AUTOSAR Multicore Distributed Architectures," *SAE Technical Paper*, no. 01-0041, pp. 1–11, apr 2016.

[24] K. Jiang, P. Eles, and Z. Peng, "Power-Aware Design Techniques of Secure Multimode Embedded Systems," *ACM Transactions on Embedded Computing Systems*, vol. 15, no. 1, pp. 1–29, jan 2016.

[25] W. Jiang, X. Zhang, J. Zhan, Y. Ma, and K. Jiang, "Design optimization of secure message communication for energy-constrained distributed real-time systems," *Journal of Parallel and Distributed Computing*, vol. 100, pp. 1–15, feb 2017.

[26] C.-W. Lin, Q. Zhu, and A. Sangiovanni-Vincentelli, "Security-Aware Modeling and Efficient Mapping for CAN-Based Real-Time Distributed Automotive Systems," *IEEE Embedded Systems Letters*, vol. 7, no. 1, pp. 11–14, mar 2015.

[27] U. Eklund and H. Gustavsson, "Architecting automotive product lines: Industrial practice," *Science of Computer Programming*, vol. 78, no. 12, pp. 2347–2359, dec 2013.

[28] H. Gustavsson and J. Axelsson, "Improving the System Architecting Process through the Use of Lean Tools," in *Picmet 2010: Technology Management For Global Economic Growth*, 2010, p. 7.

[29] ——, "Evaluation of design options in embedded automotive product lines," pp. 478–495, January 2010.

[30] L. Wozniak and P. Clements, "How automotive engineering is taking product line engineering to the extreme," in *Proceedings of the 19th International Conference on Software Product Line - SPLC '15*. New York, New York, USA: ACM Press, 2015, pp. 327–336.

[31] M. Broy, I. H. Kruger, A. Pretschner, and C. Salzmann, "Engineering Automotive Software," *Proceedings of the IEEE*, vol. 95, no. 2, pp. 356–373, feb 2007.

[32] R. Roy, P. Souchoroukov, and E. Shehab, "Detailed cost estimating in the automotive industry: Data and information requirements," *International Journal of Production Economics*, vol. 133, no. 2, pp. 694–707, oct 2011.

[33] J. Axelsson, "Cost Models with Explicit Uncertainties for Electronic Architecture Trade-off and Risk Analysis," *INCOSE International Symposium*, vol. 16, no. 1, pp. 1700–1714, jul 2006.

[34] O. Larses, "Architecting and Modeling Automotive Embedded Systems," Ph.D. Thesis, KTH, 2005.

[35] M. Walker, M.-O. Reiser, S. Tucci-Piergiovanni, Y. Papadopoulos, H. Lönn, C. Mraidha, D. Parker, D. Chen, and D. Servat, "Automatic optimisation of system architectures using EAST-ADL," *Journal of Systems and Software*, vol. 86, no. 10, pp. 2467–2487, oct 2013.

[36] X. Zhang, L. Feng, M. Törngren, and D.-j. Chen, "Formulating customized specifications for resource allocation problem of distributed embedded systems," in *Proceedings of the 35th International Conference on Computer-Aided Design - ICCAD '16*. ACM Press, 2016, pp. 1–8.

[37] P. Emberson and I. Bate, "Stressing Search with Scenarios for Flexible Solutions to Real-Time Task Allocation Problems," *IEEE Transactions on Software Engineering*, vol. 36, no. 5, pp. 704–718, sep 2010.

[38] F. Sagstetter, P. Waszecki, S. Steinhorst, M. Lukasiewycz, and S. Chakraborty, "Multischedule Synthesis for Variant Management in Automotive Time-Triggered Systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 4, pp. 637–650, apr 2016.

[39] D. Sundmark, K. Petersen, and S. Larsson, "An exploratory case study of testing in an automotive electrical system release process," in *2011 6th IEEE International Symposium on Industrial and Embedded Systems*. IEEE, jun 2011, pp. 166–175.

[40] X. Zhang, M. Persson, M. Nyberg, B. Mokhtari, A. Einarson, H. Linder, J. Westman, D. Chen, and M. Torngren, "Experience on applying software architecture recovery to automotive embedded systems," in *2014 Software Evolution Week - IEEE Conference on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE)*. IEEE, feb 2014, pp. 379–382.

[41] N. Mohan, P. Roos, J. Svahn, M. Törngren, and S. Behere, "ATRIUM - Architecting Under Uncertainty: for ISO 26262 Compliance," in *IEEE Systems Conference (SysCon'17)*, 2017, pp. 1–8.

[42] T. Rolina, "Past , Present , and Future of Real-Time Embedded Automotive Software : A Close Look at Basic Concepts of AUTOSAR," *SAE Technical Paper Series*, vol. 01, no. 1236, 2006.

[43] M. Di Natale and A. Sangiovanni-Vincentelli, "Moving From Federated to Integrated Architectures in Automotive: The Role of Standards, Methods and Tools," *Proceedings of the IEEE*, vol. 98, no. 4, pp. 603–620, apr 2010.

[44] P. Nuzzo, A. L. Sangiovanni-Vincentelli, D. Bresolin, L. Geretti, and T. Villa, "A Platform-Based Design Methodology With Contracts and Related Tools for the Design of Cyber-Physical Systems," *Proceedings of the IEEE*, vol. 103, no. 11, pp. 2104–2132, 2015.

[45] K. Jo, J. Kim, D. Kim, C. Jang, and M. Sunwoo, "Development of Autonomous Car—Part I: Distributed System Architecture and Development Process," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 12, pp. 7131–7140, dec 2014.