THE | AUTONOMOUS

# SAFETY & ARCHITECTURE
## Working Group

Digest Report

*The full report of the Safety & Architecture Working Group will be published at the end of 2023. The present document is a digest summarizing the content and key findings. Please refer to the full report for the detailed discussion.*

# THE AUTONOMOUS

The Autonomous is the global community shaping the future of safe autonomous mobility. Initiated by TTTech Auto, The Autonomous is an open platform bringing together the leading executives and experts of the autonomous mobility ecosystem to align on relevant safety subjects. The primary objective of The Autonomous is to cultivate fresh insights and technological breakthroughs within the realm of autonomous mobility. To accomplish this, The Autonomous has established two strategic streams of action:

## 1. EVENT STREAM

facilitates insightful discussions and networking opportunities for leading executives and experts from the autonomous mobility ecosystem.

## 2. INNOVATION STREAM

fosters cooperation across the industry, nurturing the development of globally recognized reference solutions tailored to address safety challenges. These reference solutions align with pertinent standards, serving as a catalyst for the widespread adoption of safe autonomous mobility on a global scale. The Innovation Stream entails the launch and facilitation of **Working Groups** and **Expert Circles** serving as collaborative platforms for the creation of recommended practices and tangible advancements.

# THE SAFETY & ARCHITECTURE WORKING GROUP

## OUR MEMBERS

BASELABS

CoreAVI
INNOVATING A SAFER TOMORROW

KTH
VETENSKAP
OCH KONST

TTTechAuto

Infineon

DENSO
DENSO AUTOMOTIVE Deutschland GmbH

It is commonly understood and accepted that the development and implementation of a failure-free automated driving system for complex driving tasks is a huge challenge. Even when developed to the highest standards, complex HW and SW elements will exhibit faults that can materialize in an arbitrary way. Still, the overall autonomous driving system needs to tolerate such faults and keep up operation at least for a minimum time frame – i.e., it needs to be fail-operational.

In the Safety & Architecture Working Group, members of international research institutes and industrial companies came together to discuss what the conceptual system architecture of an automated vehicle (SAE Level 4 and higher) could look like, in order to address the functional and safety challenges of automated driving.

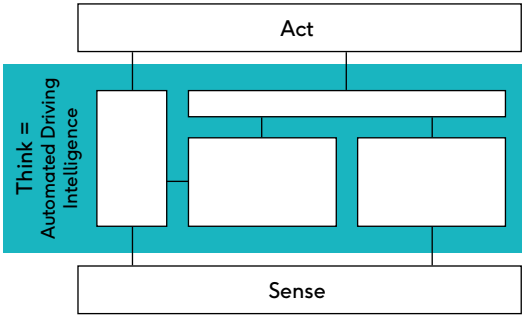
The group first convened in June 2021 and is now, two years later, ready to deliver its report. In this time, we have outlined our reference use case of an SAE L4 Highway Pilot, researched the market and the literature for publicly available information about AD architectures, and tried to derive and document the key properties of these published architectures. Finally, we have compared the architectures with respect to a set of key criteria we consider crucial (such as availability, robustness, and security).

Our work was structured in three major report increments that were accompanied by industry and academic experts as external reviewers. Contributions were partly created offline by individual members and reviewed and discussed by the whole team; numerous topics were also first intensely discussed in informal meetings and later put in writing. Weekly remote meetings were held to track progress, clarify open points, and align on next steps. When needed and possible, this was supplemented by regular in-person workshops. Despite the general travel situation around the pandemic, we worked together with remarkable coherency and team spirit, ultimately achieving what we believe is the first survey of its kind.

# PURPOSE, ABSTRACTION LEVEL, AND REFERENCE AD USE CASE

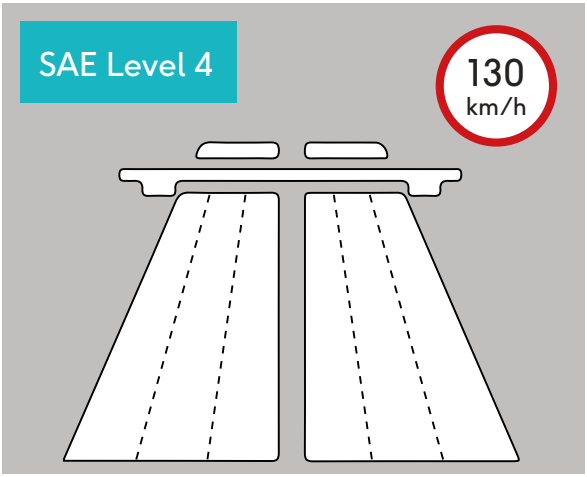
The intended readers of the report are so-called system owners, who make architectural decisions and ensure consistency on many different abstraction levels, from high-level conceptual architectures to low-level physical implementations. Our intention is to support them in making such decisions and building up a safety argumentation.

For the Safety & Architecture Working Group, we have chosen what we call the **conceptual** abstraction level (see illustration on the left). Here, the system is composed of a small set of well-encapsulated subsystems that fail independently (so-called "Fault Containment Units" or FCUs), which can comprise an entire processing channel (from sensors to actuators).



- Conceptual architectures are **sufficiently generic** to be applicable to most system owners, irrespective of commercial or implementation considerations.

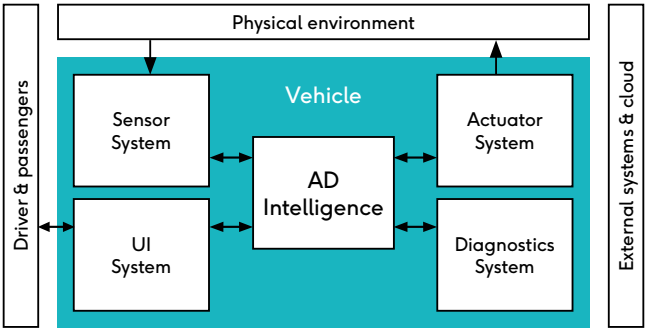
- They are also **sufficiently non-trivial** to bring a benefit to system owners, providing solutions to the question of how to manage sufficiently independent redundancy.




We have outlined a reference AD use case to act as a backdrop for deriving system requirements, assumptions, and design principles applicable to conceptual system architectures. For this, we have chosen an assumed SAE L4 Highway Pilot feature. Not only are such features expected within the next few years and need to deal with complex traffic situations, but more importantly they also imply high availability requirements. These necessitate non-trivial system architectures and therefore pose a new challenge.

# SYSTEM REQUIREMENTS, ASSUMPTIONS, AND DESIGN PRINCIPLES

The Safety & Architecture Working Group primarily considers a system providing AD functionality, which we call the **Automated Driving Intelligence (ADI)**. This system covers all cognitive tasks previously performed by the driver. A simplified representation is shown on the left, illustrating the four other systems the ADI is connected to, as well as the elements that "close the loop" with the physical environment.




There are several **system requirements** applying to the ADI that ensure the safety of commands to control the vehicle. In addition to the correctness of commands, their availability is now also safety-relevant. These are summarized in the table below.

| | |
|---|---|
| S1 | ADI output timeliness |
| S2 | ADI output availability |
| S3 | ADI output correctness |
| S4 | ADI output consistency |
| S5 | Perception malfunction detection |
| S6 | Driver monitoring |
| S7 | ADI diagnostics |

When coming up with conceptual system architectures intended to satisfy these system requirements, several aspects should be considered:

- Certain basic technological limitations constrain how ultra-high reliability systems can be designed, built using realistic HW and SW components, and tested. We have identified 10 such **general constraints**. Examples are limitations to achieving safety via testing or to avoiding design faults in large and complex monolithic systems.

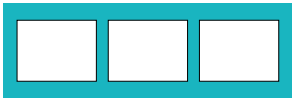
- In addition, there is a set of empirical best practices that should be respected in a sound conceptual system architecture. We have identified 7 such **design principles**. These include, for example, using well-encapsulated subsystems (the FCUs mentioned before), preventing emergent behavior by limiting interactions between subsystems, and mitigating common-cause hazards by adapting the Swiss cheese model.

# CANDIDATE ARCHITECTURES

In the following section, we describe each architecture's structure and behavior, based on our interpretation of the respective source material (scientific paper or patent). We have identified three broad categories of conceptual system architectures:

### 1. MONOLITHIC ARCHITECTURES

present the status quo for SAE L2 ADAS and serve as the baseline for the evaluation.



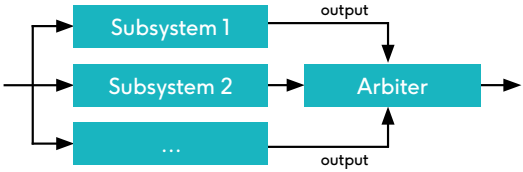
### 2. SYMMETRIC ARCHITECTURES

rely on multiple channels providing the same or similar functions, often with some voting mechanism for arbitration.

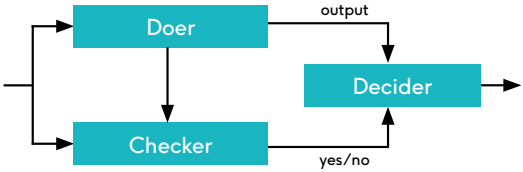
### 3. ASYMMETRIC ARCHITECTURES

employ asymmetric decompositions to reduce the complexity of some subsystems, e.g., via Doer / Checker or Active / Hot Stand-By patterns.

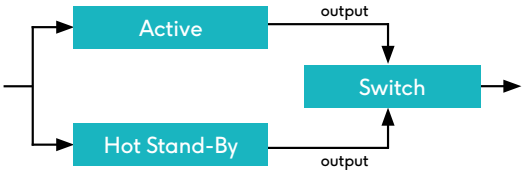
The candidate conceptual system architectures we collected from industry and academia share several underlying patterns:




- The Voting or Arbitration pattern manages redundancy by voting between equal channels. The Agreement pattern is similar, but without an external arbitrator.




- The Doer / Checker pattern asymmetrically decomposes (for correctness) a channel into a Doer performing the function and a Checker approving it.



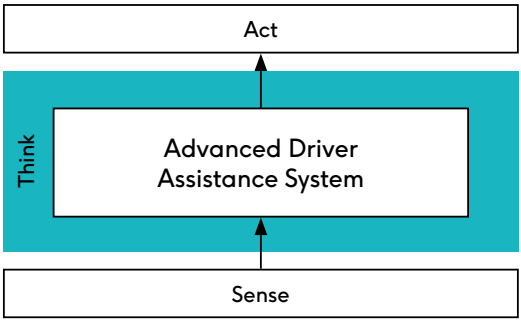

- The Active / Hot Stand-By pattern asymmetrically decomposes (for availability) into a preferred Active channel and – if that is not available – a Fallback channel.

The sole representative of the monolithic architectures is the **Single Channel** architecture. We based our description on the architecture presented by AUDI in 2015, which at the time introduced the first system intended to go beyond SAE L2 into the area of automated driving.

This was done with a single ECU, which can be seen as a single-channel or monolithic architecture. In such an architecture, a single failure of

a component can lead to system-level failure. The functional behavior of such a system can be described as follows:

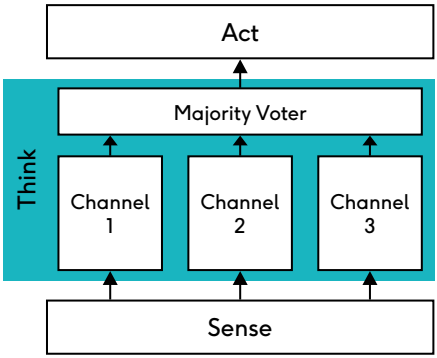
1. It processes received sensor data into a consistent environment model.
2. Then it periodically generates trajectories and corresponding actuator setpoints.
3. These setpoints are then sent to the actuator system.
4. If an internal fault is detected, the system fails silently (with a message to the driver that the system is unavailable).




Due to the underlying use case of an SAE L3 Traffic Jam Pilot (low speed and a restrained environment), the safety requirements are noticeably different from most AD use cases with respect to integrity (i.e., complex functionality does not need to reach the highest ASIL) and availability (i.e., the system does not need to provide complex fallback functionality in case of a fault).

We consider Tesla's „Full Self Driving" (FSD) another more recent implementation of a single-channel architecture (as far as can be judged from available documentation).

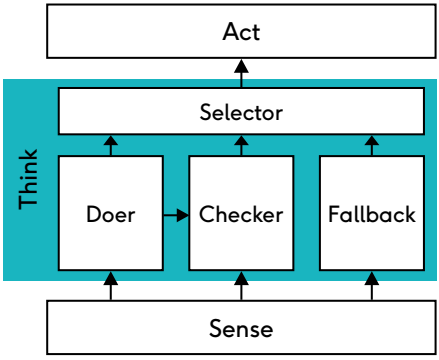
The sole considered representative of the symmetric architectures is the **Majority Voting** architecture. We based our description on [1]. This conceptual system architecture consists of the following principles:




- A multi-channel architecture with a voter and at least three channels.
- Each of the channels is capable of performing the nominal driving function, i.e., can generate trajectories and corresponding actuator setpoints.
- The voter assumes that the majority is correct. It compares (exactly or inexactly) the results from the channels and forwards the majority opinion to the actuators. If all three results differ, no decision can be made.
- To achieve fault tolerance, multiple instances of the voter may be necessary.

The first of the considered asymmetric architectures is the **Channel-Wise Doer/Checker/Fallback** architecture. We based our description on [2].

This conceptual system architecture consists of the following principles:




- The Doer performs the nominal driving function and can resemble an SAE L2+ system.
- The Fallback performs only Minimum Risk Maneuvers and is only in control of the vehicle if the Doer has failed.
- The Checker validates the outputs of the Doer and Fallback and sends the results (safe/unsafe) to the Selector.

- The Selector forwards either the commands from the Doer or those from the Fallback to the actuators, depending on the results from the Checker. It is so simple and low in performance requirements that it can be fully verified to preclude systematic faults. To achieve fault tolerance, it consists of two identical instances. Each subsystem forms an FCU to ensure sufficient independence.

- Doer, Checker, and Fallback can fail arbitrarily and are implemented in a diverse way to minimize common-cause failures. A time-triggered architecture facilitates the redundancy management (cycle-by-cycle) and minimizes interactions between subsystems.

- A faulty Doer is detected by the Checker. The Selector then quickly switches to forwarding the Fallback's commands to the actuators.

- If any other subsystem is found to be faulty, the Doer is sent into a degraded mode, which can involve handing back control to the driver and/or ultimately performing a Minimum Risk Maneuver.

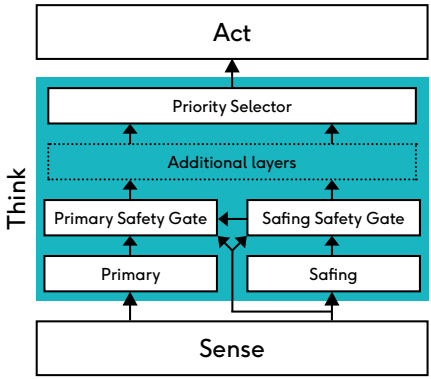
The second considered asymmetric architecture is the **Layer-Wise Doer/Checker/Fallback** architecture. We based its description on our interpretation of the patent [3].

This conceptual system architecture consists of the following principles:

- A multi-channel approach, in which at least a "primary" and a "safing" channel are present.

- The safing channel provides a degraded mode of operation in case the primary channel fails.

- An arbiter "Priority Selector" determines the output to be sent to the actuators, depending on the states of the channels.



- Each channel consists of multiple Doer/Checker pairs, one for each layer or stage of the Sense-Plan-Act model.

- The Doers may have low safety integrity levels and may each fail arbitrarily.

- The Checkers are high safety integrity components responsible for checking the outputs of the Doers. A Checker fails silently when it detects unsafe outputs of the corresponding Doer. If a check of the safing channel fails, both primary and safing outputs are inhibited, and a buffered safe trajectory is used. After a time window passes without a correct input from the safing channel, an emergency stop is executed.

- The arbiter is a high safety integrity component, simpler than the Checkers. It must continue to operate in the presence of failures to deliver either the primary, the safing output or to trigger an emergency stop. It may fail silently so long as that failure triggers an emergency stop.

- To strengthen the argumentation for sufficient independence, e.g., with respect to shared information input and exchange of information, we recommend to explicitly encapsulate the channels in FCUs.

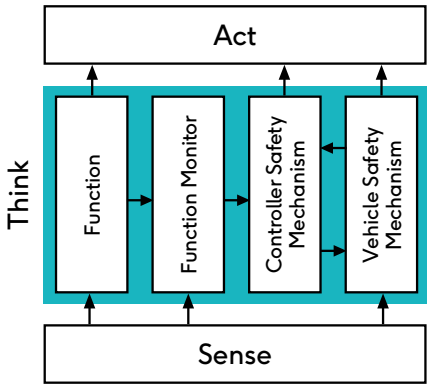
The third considered asymmetric architecture is the **Distributed Safety Mechanism** architecture, as described in [4], which can be seen as a distributed variant of the Doer/Checker/Fallback approach.

This conceptual system architecture consists of the following principles:

- The architecture is composed of three channels, each of them containing safety monitors.

- The nominal channel, consisting of the function itself controlled by a Function Monitor.

- The emergency channel, which is controlled by a Controller Safety Mechanism.



- The safety channel, which is controlled by the Vehicle Safety Mechanism.

- The Function Monitor may have a low safety integrity and is responsible for checking SOTIF issues and monitoring the status of the corresponding function. It fails silently.

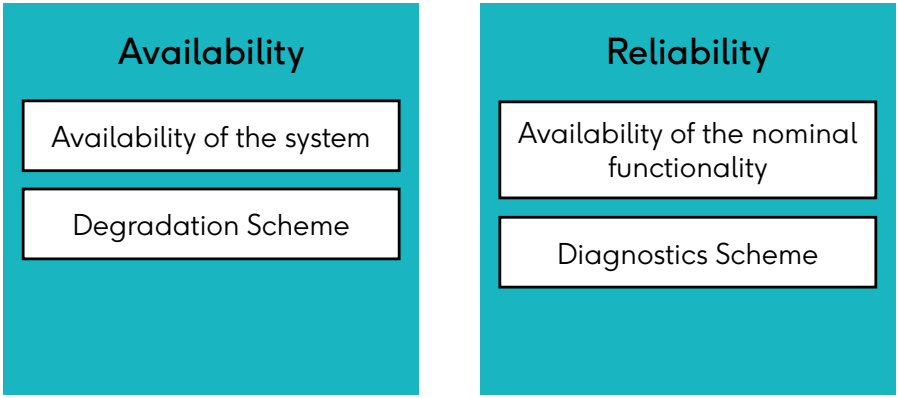
- The Controller Safety Mechanism has a medium safety integrity and is responsible for monitoring all the function controllers (including hardware and software platforms) and the Vehicle Safety Mechanism. It can send control commands to the vehicle actuators in case of a detour or emergency stop.

- The Vehicle Safety Mechanism has a high safety integrity and is responsible for monitoring the communication networks and the Controller Safety Mechanism. It can send control commands to the vehicle actuators in case of comfort or safe stop, by using independent sensor data. It fails silently.

- To strengthen the argumentation for sufficient independence, e.g., with respect to shared HW resources, we recommend to explicitly encapsulate the channels in FCUs.

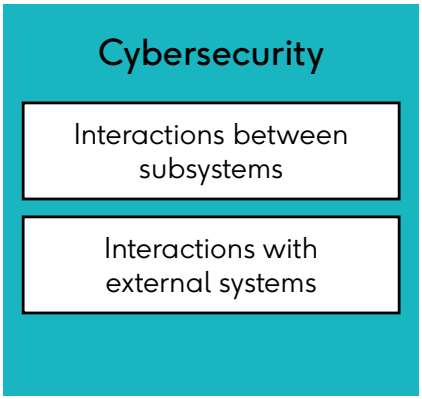
# ARCHITECTURE EVALUATION METHODOLOGY, CRITERIA, AND FINDINGS

The evaluation of candidate architectures was conducted with respect to several key criteria that the team aligned on:

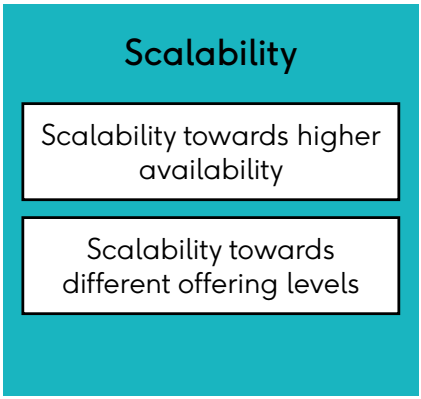



To what extent would the architecture support the fail-operational property, i.e., enable safe operation even in the case of unavoidable electronic or software faults?

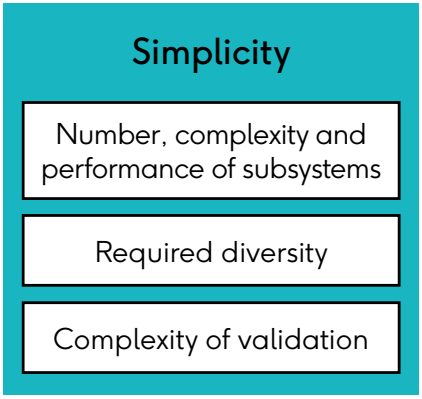
Would continuity of the nominal functionality be well supported, to help ensure a positive user experience, e.g., by avoiding function degradation?



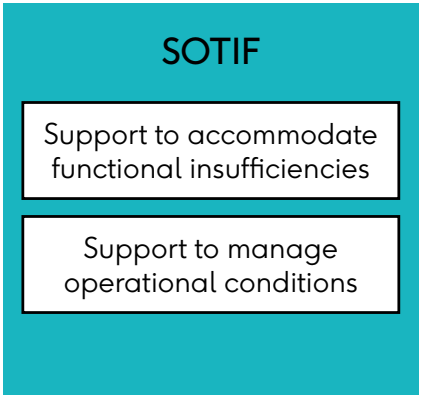

Would the architecture be susceptible to security threats, or would it support resilience measures against attacks?




To what extent would cost-efficient downscaling to lower SAE levels (for vehicle options), or upscaling to higher SAE levels (for future enhancements) be supported?




Would the architecture be conceptually simple, to support modular development, verification, and validation?




Safety of the Intended Functionality: would the architecture help ensure robustness and safe operation in the presence of functional imperfections and unavoidable edge cases?

The evaluation was performed in several steps: To form an unbiased basis for the evaluation, we started with a generic evaluation of each architecture, by listing observations related to each criterion, i.e., properties of each architecture perceived by the team. Next, we evaluated the relative significance of the above criteria for the selected use case of an SAE L4 Highway Pilot. Finally, we directly compared the architectures, considering the observed properties from the generic evaluation and inferring merits or weaknesses with respect to each evaluation criterion, and ranking them under that criterion.

As a result, it turned out that the **asymmetric architectures are generally preferable to symmetric ones.** By virtue of their inherent diversity of computational streams, they exhibit more robustness with respect to availability, cybersecurity, and SOTIF because the channels complement each other and tend to mutually compensate their potential weaknesses. The asymmetric architectures also offer more options with respect to scalability, as omitting channels quite naturally leads to lower SAE level functionality, and higher levels can be reached by adding channels. Superficially, they might appear more complex and less reliable (in the sense of keeping the intended functionality) than symmetric architectures. However, their diversity actually facilitates modular development and independent verification of the channels, which in turn is expected to lead to lower development costs and enhanced reliability.

The symmetric architectures, such as voting approaches, were seen as highly susceptible to common cause deficiencies that might impact all channels at the same time – be it from the functional safety, SOTIF, or even the cybersecurity perspective. If this problem is addressed by heterogeneous channel implementations (e.g., different chipsets), then the feasibility of voting is questionable since channels might come to different but equally valid solutions. Finally, the monolithic single-channel architecture is not seen as a feasible solution: it does not fulfill any of the criteria without additional internal redundancy and supervision mechanisms that are introduced during implementation. This would make it evolve into one of the other architectures.

# IMPLEMENTATION CONSIDERATIONS

For further refinement of the conceptual system architecture into hardware solutions with redundant channels, dependent failures of the hardware elements are an important aspect to consider. In other words, freedom from interference between channels and the absence of common hardware elements for redundant channels needs to be ensured. We discussed dependent failure initiators and provided hints on how to overcome them.

Similarly, we considered selected topics related to the further refinement of the conceptual system architecture into a software safety concept. This included a discussion on different software architectural styles – depending on the use case – as well as common safety measures. Considering the increasing number and complexity of software elements required for automated driving systems, we briefly discussed the relevant aspects of software reuse, software updates, real-time operating systems, machine learning, and software tool qualification.

To achieve a sound safety argumentation for the chosen architectures, we referred to the relevant safety standards, in particular ISO 26262 and ISO 21448. For instance, the partitioning provided by the presented architectures can also be understood as an ASIL decomposition in the sense of ISO 26262. We proposed advanced methods like formal verification on the architecture level and for the logical-to-physical mapping, as well as probabilistic methods to quantify the residual faults of components, to meet an ASIL D target for the system availability.

# NEXT STEPS

The further direction of the Safety & Architecture Working Group is still under discussion among the member companies of The Autonomous, and some of the potential work packages could be as follows:

- Extend the analysis to further use cases, such as SAE L5 or an Urban Pilot function, and see if and to what extent the outcome of the evaluation of each architecture's relative merits and weaknesses changes.

- Extend the analysis to other architectures that seem to evolve in the market, such as self-checking pairs.

- Deepen the analysis to include more implementation aspects, e.g., to propose concrete HW and SW mappings and suitable ECU and networking architectures.

- Extend the report with practical guidelines, such as how to check on and ensure logical completeness and consistency of an architecture, or how to evaluate and quantify the independence of computation channels.

- Work out an "architecture evaluation guideline" from the experiences gained throughout the presented work, to help the industry community apply a similar framework in their concrete development projects.

Whatever the future direction, working on this report has been a hugely gratifying experience for the team, and we hope that it provides good value to the community of industry players and academic institutions working on automated driving systems.

# REFERENCES

[1] M. L. Shooman, „Reliability of computer systems and networks: fault tolerance, analysis and design," in N-Modular Redundancy, Wiley-Interscience, 2002, pp. 145-201.

[2] H. Kopetz, „An Architecture for Safe Driving Automation," in Principles of System Design, Springer, 2023.

[3] M. Wagner, J. Ray, A. Kane and P. Koopman, „A safety architecture for autonomous vehicles". Patent EP3400676B1, 2017.

[4] Y. Fu, A. Terechko, J. Groote and A. Saberi, „A formally verified fail-operational safety concept for automated driving," SAE Intl., pp. 7-21, 17 Jan 2022.

[5] A. Avizienis, J.-C. Laprie, B. Randell and C. Landwehr, „Basic concepts and taxonomy of dependable and secure computing," IEEE Transactions on Dependable and Secure Computing, vol. 1, no. 1, 2004.

THE | AUTONOMOUS