Smart Monitoring for Safety-Assurance in Autonomous Driving

Georg Stettinger*†
Infineon Technologies AG
Am Campeon 1-15, 85579 Neubiberg, Germany
Georg.Stettinger@infineon.com

Franz Wotawa
Institute for Software Technology
Graz University of Technology
Inffeldgasse 16b/2, Graz, Austria
wotawa@ist.tugraz.at

Abstract

Monitoring the functionality of systems during operation is vital for detecting faults and preventing their consequences. In autonomous driving, monitoring is even more critical because of hardly being able to verify all implemented functionality. Today, systems comprise many interacting components making centralized monitoring less feasible and hard to handle. Hence, we suggest a distributed but connected monitoring system that reflects the system's conceptual structure. In this paper, we outline the foundations of a monitoring system, present some applications and show how we use concepts like the operational design domain and requirements for obtaining the required monitoring knowledge in the application area of autonomous driving.

1 Introduction

Autonomous driving has gained much attention in the past years. Driving factors are reduced costs, the opening of new opportunities for services, and the reduction of fatalities, among others. When focusing on decreasing fatalities in driving, we have the underlying assumption that autonomous driving is safe or at least safer than human driving. Checking that a concrete implementation of autonomous driving reaches this goal requires sophisticated verification and validation before deployment. Many researchers have been working on this topic, including Koopman and Wagner [6], Wotawa [11], Schuldt and colleagues [9], or Wotawa and colleagues [13].

The underlying challenge is to find a way of justifying the quality of verification and validation. For this purpose, Kalra and Paddock [5] stated that an autonomous vehicle has to operate for 275 million miles for verification purposes. In their calculation, the authors took the fatality rate of driving in the USA and assumed that autonomous vehicles should be far safer. Unfortunately, testing for 275 million miles is not feasible in practice. Therefore, other researchers have proposed using ontologies for testing automated, autonomous vehicles, e.g., [14, 2, 7]. These contributions focus on testing potential unsafe interactions between the autonomous car and its environment. Note that there is a massive number of potential interactions between a car and its environment, indicating the need for additional means for verifying the behavior of autonomous vehicles.

One way to reduce the search space for verification and validation is to restrict the use domain. And indeed future electric connected and automated (ECA) vehicles will be deployed in specific targeted operational design domains (ODD) across Europe [4], [8] able to safely and robust operate in the target ODD making use of their embedded behavior competences [10]. The recently published and continuously amended EU ADS regulation [1] outlines some general safety regulations L4+ vehicles to be deployed within Europe have to comply with. Especially the outlined pillar III, referred to as in-service monitoring, highlights the importance of monitoring activities during the operation of future ECA vehicles after they enter the market and get deployed on public roads. In-service monitoring and reporting targets to learn from in-service data as a central component to the safety potential of future ECA vehicles enabling key objectives referred to safety confirmation, scenario generation, and safety recommendation representing a major source of general safety requirements of future ECA vehicles.

Figure 1 outlines the general architecture of a monitoring system. First, the monitoring system receives internal values from the system under supervision using a monitoring interface. The internal values comprise sensor inputs, internal states, error signals, and actuator commands over time. Next, these values are checked for consistency with expectations using a knowledge base. In case of inconsistencies, the monitoring system raises an error or warning,

*Authors are listed in alphabetical order.
†DOI reference number: 10.18293/SEKE2023-144

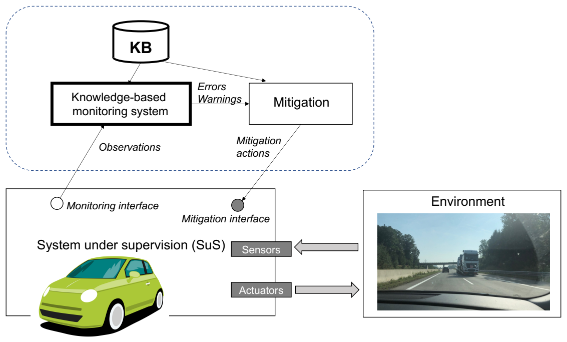


Figure 1: The proposed monitoring architecture.

depending on the severity, which the mitigation mechanism uses for suggesting corrective or compensating actions. Finally, mitigation transfers these actions to the system using the mitigation interface. It is worth stating that the monitoring system is independent of the system under supervision and only communicates via the proposed interfaces. The system itself communicates with its environment using actuators and sensors.

Because of the hierarchical nature of systems comprising subsystems down to basic components (at least from a conceptual point of view), Lewitschnig and Wotawa [12] introduced a hierarchical monitoring concept. In this paper, we extend this concept to a general monitoring paradigm comprising separate monitoring nodes that are interconnected and which belong to parts of a system. Each monitoring node checks the behavior of its corresponding parts, also considering results from other monitoring nodes. The monitoring nodes perform the checks utilizing formalized knowledge of the expected behavior. In case of severe violations, we assume the existence of mitigation actions like safety maneuvers. However, we do not discuss how mitigation will be carried out in this paper and leave this for future research. In addition, to knowledge-based monitoring, we further discuss how to obtain knowledge by considering ODDs and hierarchically structured monitoring systems. Finally, we illustrate the process using examples from the automotive industry.

We structure this article as follows. We start motivating research in knowledge-based monitoring using the case of an accident of a Tesla car driving in autonomous mode (Sect. 2). Afterward, Section 3 introduces the foundations of monitoring and its specialization into hierarchical monitoring devices. Then, in Section 4, we introduce a process for obtaining the required knowledge using ODD and illustrate its use. Finally, we conclude the paper.

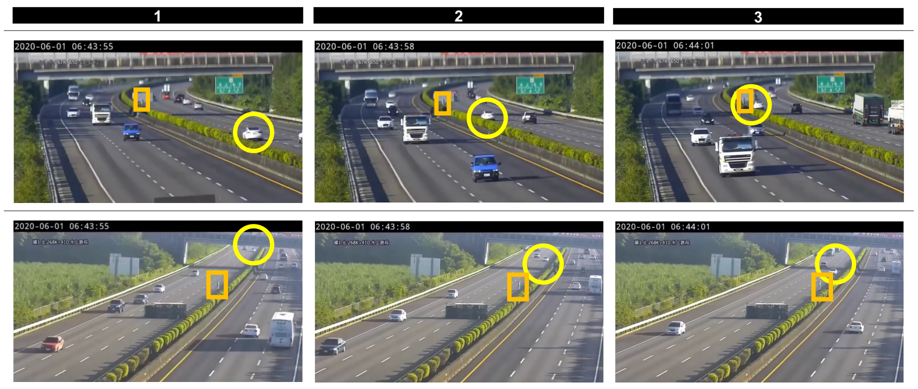
2 Motivation

In this section, we motivate the need for monitoring autonomous driving, considering different views. First, let us discuss an accident of a Tesla car driving in autonomous mode on a highway in Taiwan in 2020, where we depict the most important parts of the accident's chronology in Figure 2. The Tesla car was not reducing speed until braking came too late. As a result, the Tesla crashed into an overturned truck that blocked the two left lanes of the highway. It seems that Tesla's vision systems did not perceive the truck. Hence, there might be an issue with the vision system that an autonomous car can hardly identify without perception redundancies. However, even worst, the Tesla was not even reacting to the truck driver on the highway, who warned drivers. Such behavior does not belong to any road traffic regulation.

The Convention on Road Traffic emerged from the United Nations Conference on Road and Motor Transport in September 1949 in Geneva[1]. In the document Article, 7 and 10 are of particular interest to us. Article 7 states: "*Every driver, pedestrian or other road user shall conduct himself in such a way as not to endanger or obstruct traffic; he shall avoid all behaviour that might cause damage to persons, or public or private property.*". Especially the last part is of interest to us when considering the Tesla accident, where the car is passing by the truck driver at high speed, not considering potential damage to persons. This behavior also conflicts with Article 10: "*The driver of a vehicle shall at all times have its speed under control and shall drive in a reasonable and prudent manner. He shall slow down or stop whenever circumstances so require, and particularly when visibility is not good.*". Of course, in the mentioned case, the Tesla driver is responsible for violating the traffic convention. However, the car in autonomous driving mode was not implementing the regulations too.

We can derive from the Tesla accident example: There is a need for thoroughly verifying the vision system, considering different and even unexpected scenarios. In order to detect faulty behavior of an autonomous car while driving, it is essential to have redundancies in perception. Otherwise, we are not able to detect perception faults. In addition, we need to formalize traffic rules and check them during operation. For example, in the Tesla accident, the control system might have ignored the truck driver due to other regulations preventing persons from walking on highways. An independent monitoring system relying on formalized traffic regulations would detect such a fault and be able to initiate countermeasures like braking or changing lanes. In any case, a monitoring system utilizes perceived informa-

[1]See UN Treaties Chapter XI.B. TRANSPORT AND COMMUNICATIONS `https://treaties.un.org/doc/Treaties/1952/03/19520326%2003-36%20PM/Ch_XI_B_1_2_3.pdf`



A Tesla car in automated driving mode is approaching an overthrown truck on the highway in Taiwan, which blocks two lanes. The truck driver stands on the left lane to warn other drivers.

The Tesla car is not reducing speed or changing lanes. The driver of the Tesla is not taking any actions to overrule automated driving.

The Tesla is continuing driving. About 4 seconds later, the Tesla crashed into the Truck. The Tesla driver started braking too late.

Figure 2: Chronology of the 2020 Tesla crash in Taiwan where a Tesla in autonomous driving mode crashed into an overthrown truck on the highway. The Tesla driver was overruling autonomous driving too late to prevent the crash. Pictures are taken from a published YouTube video `https://youtu.be/LfmAG4dk-rU`

tion and checks conformance with formalized regulations and rules.

In addition, a monitoring system has to deal with information coming from various parts of the system under observation. For example, there might be a fault reported in the battery subsystem of an autonomous and electrified vehicle that might limit the operational range of the vehicle. However, when knowing that the vehicle can complete the particular journey having enough electricity to arrive at a workshop afterward safely, the reported fault can be ignored during the journey. In this example, the conclusion requires information from different parts located at different levels of a conceptual hierarchy. For example, the battery belongs to the power train, and knowledge regarding the journey comes from a planning module that uses the power train component to move the car from its current location to its destination. Hence, it might be appropriate to consider conceptual hierarchies when monitoring systems. Furthermore, utilizing conceptual hierarchies during monitoring allows for specifying the knowledge for checking the current status at each level of the hierarchy. In this way, developers of monitoring functionalities focus only on one hierarchy and do not need to know all details of parts. The following sections will outline more details regarding this monitoring design process.

3 Monitoring Foundations

In the following, we outline the foundations behind knowledge-based monitoring. We extend previous work [12] to handle monitoring systems of all kinds of structures, not just hierarchical ones, i.e., tree-structured monitoring systems.

Definition 1 (Monitoring System) *A tuple $(COMP, CONN)$ is a monitoring system where $COMP$ is a set of components, and $CONN$ is a set of connections between components, i.e., a set of pairs from $COMP \times COMP$.*

Because of simplicity, we further introduce the functions for obtaining the *predecessor* and *successor* of a component $C \in COMP$, i.e., $pred(C) = \{C'|(C', C) \in CONN\}$, and $succ(C) = \{C'|(C, C') \in CONN\}$ respectively, and the definition of *basic nodes*, i.e., nodes that do not have successors, $basic = \{C|(C', C) \in CONN\} \wedge succ(C) = \emptyset$.

What we need to add is the monitoring and checking of the behavior of a system. For this purpose, we introduce three additional functions we must define for each monitoring system component. We assume, without restricting generality, to use first-order logic (FOL) for representing the information delivered by these functions.

Output: The $out : COMP \mapsto FOL$ function delivers a set of predicates to hold when executing the given component.

Constraints: The $constr : COMP \mapsto FOL$ function delivers first-order logic sentences that must hold in the context of a given component.

Behavior: The $behav : COMP \mapsto FOL$ function returns a set of first-order logic sentences that are used with the inputs of a component (i.e., the outputs of the successor nodes) to derive the outputs.

We use these functions to express the behavior of the system. For each component C_i, we can derive its output from

Algorithm 1 $monitoring(O, (COMP, CONN))$

```
Input: A set of observations $O$ obtained from the system under the supervision and a monitoring system.
Output: $\top$ if all constraints are fulfilled, and $\bot$ otherwise
 1: Convert each observation $o \in O$ to its corresponding predicate $p$ and assign it to a corresponding monitoring system component $C \in COMP$.
 2: while There is no change in $out(C)$ of any $C \in COMP$ do
 3:     for all $C \in COMP$ do
 4:         for all Fact $P$ that can be derived from $behav(C) \cup (\bigcup_{C' \in pred(C)} out(C'))$ do
 5:             Add $P$ to $out(C)$.
 6:         end for
 7:         if $constr(C) \cup out(C) \not\models \bot$ then
 8:             Add $violation()$ to $out(C)$
 9:         end if
10:     end for
11: end while
12: if $\exists C \in COMP : violation() \in out(C)$ then
13:     return $\bot$
14: else
15:     return $\top$
16: end if
```

the inputs considering the defined behavior of the component. Moreover, if each component C_i works correctly, all constraints must be fulfilled considering given inputs and computed output. We use both rules in the monitoring algorithm (Algorithm 1), called every predefined time step t. In the first line of the algorithm, there is a mapping of current observations at time t to corresponding predicates of the monitoring system components. Afterward, we apply derivations until no new facts can be derived anymore. Next, the algorithm derives new facts in lines 4–6, where we use the inputs, i.e., the outputs of predecessors of the current component, and the given behavior of the component to derive new facts P. Afterward, the algorithm checks the fulfillment of given constraints using the current facts in Line 7.

Algorithm $monitoring$ terminates because, in each step, it adds new facts to $out(C)$ for the current component C. Adding facts is done until no new facts can be derived. Because we consider only a finite number of facts, the algorithm must terminate. It is also worth mentioning that the algorithm transfers all facts of any node to its successor. If this is not wanted, we need to add a filtering rule. Further note that the algorithm generalizes a previous one [12].

We illustrate the monitoring foundations using a small example where we formalize a monitoring system that would have at least indicated a wrong driving behavior for the Tesla accident from Figure 2. We assume three monitoring components: C_{gps} for gaining information about speed and the location, i.e., whether the car is on a highway or not accordingly to given maps, C_{vision} for indicating other vehicles or persons, and C_{car} for the overall Tesla car. Obviously, C_{gps} and C_{vision} are predecessors of C_{car}. Hence, we represent the monitoring system using the tuple $(\{C_{gps}, C_{vision}, C_{car}\}, \{(C_{gps}, C_{car}), (C_{vision}, C_{car}\})$.

For all components, we do not add behavior in this example. For C_{gps}, we assume that predicates $onHighway()$, $speed(v)$, and $braking()$ indicate the operation on the highway, the current speed v and information regarding braking, respectively. Note that the predicates can also be available in their negated form using the operator $\neg$. For C_{vision}, we have a predicate $personOnLane()$ indicate whether a person is on the driving lane within reach. For C_{car}, we do not introduce other predicates. Furthermore, for C_{gps}, we add a constraint $\neg(onHighway() \wedge speed(v) \wedge v > 130)$ stating that overspeeding on the highway is not allowed considering the speed limit in Austria, which is 130km/h. For component C_{car}, we assume that we have to brake once a person is on the lane within reach, i.e., $\neg(personOnLane() \wedge \neg braking())$.

Let us now apply the monitoring system to the Tesla accident, considering the third time step. In this case $onHighway()$, $personOnLane()$, and $\neg braking()$ must be true. We assume that the speed is less than 130km/h, e.g., 100km/h, leading to the predicate $speed(100)$. After mapping all observations to predicates, we obtain the following set of predicates for each component:

C	$out(C)$
C_{gps}	$onHighway()$, $speed(100)$, $\neg braking()$
C_{vision}	$personOnLane()$
C_{car}	

After applying algorithm $monitoring$, all predicates are also available in C_{car} because of lines 4–6.

C	$out(C)$
C_{gps}	$onHighway()$, $speed(100)$, $\neg braking()$
C_{vision}	$personOnLane()$
C_{car}	$onHighway()$, $speed(100)$, $\neg braking()$ $personOnLane()$

Obviously, the observations $personOnLane()$ and $\neg braking()$ violate the constraint $\neg(personOnLane() \wedge \neg braking())$ of C_{car}. When considering that mitigation would stop the car because of the constraint violation, the accident would have been prevented.

4 Autonomous driving use case

To maximize the learning from in-service data, we developed a hierarchical monitoring device concept [12] including 4 defined ECA vehicle layers (starting from subcomponents, via components, through sub-systems up to the system, the entire ECA vehicle itself). The main purpose of the hierarchical monitoring device approach is to monitor the health status of the individual elements across the 4 specified layers to guarantee the safe operation of the vehicle within its specified ODD. The nominal behavior of the 4 specified layers is directly specified via the system requirements and specification of the entire ECA vehicle. Compare requirements flow is highlighted in blue in Figure 3. The ECA vehicle requirements originate from the target ODD and related behavior competencies needed to safely operate within the target ODD completed by certification, homologation, legislation, and standardization of relevant aspects. Out of those 4 requirements sources, the system requirements are specified and pushed down along the 4 ECA vehicle layers (ECS value chain).

Contrary to the requirements flow (down-down from the system level), the health status flow moves bottom-up, see again Figure 3. Each layer determines the health status of its included elements and communicates them up to the next layer until the system layer (ECA vehicle) is reached.

All monitoring activities within the specific layers are operating according to the same receive-monitor-transmit principle. The receiving element collects all information about the individual health status of the current and previous layer to have a complete database ready which forms the basis for the subsequent monitoring stage. In detail, the monitoring element classifies the health status of the current layer based on the collected data of the receiving element analyzing the current residual risk [3] for continuation. In case some misbehaviors or failures are observed with a certain layer the following strategy will be applied:

- Handle, correct, and mitigate all occurring failures and misbehavior if possible within the current layer making use of e.g. redundant elements, etc.
- In case a correction is not possible within the current layer, classify the health status (optimal, acceptable, critical) based on the available information to be transmitted to the next layer

Finally, the transmitting element transfers the classified health status of the current layer to the subsequent one. In case the system layer is not able to correct eventually occurring faults and misbehavior directly, the ECA vehicle has to reduce its specified manageable target ODD and/or behavior competencies to further guarantee a safe operation and avoid unacceptable residual risks. In other words, the major goal behind that approach is to continue the operation as long as the residual risk in reduced ODD and behavior competences is acceptable again as a safe-stop is not always the best option in case of faults and observed misbehavior. The reduction of the target ODD or behavior competences can be applied several times until the mandatory common basis gets lost, e.g. on the highway the ECA vehicle is just able to drive 30km/h anymore. This operation is against the traffic law which makes a continuation impossible. In that aspect, a so-called minimum risk maneuver has to be trigger that safely terminates the operation of the ECA vehicle e.g. on the emergency lane of the highway.

5 Conclusions

This paper discussed the importance of monitoring focusing on autonomous driving and introduced a general monitoring framework comprising interconnected monitoring nodes. Besides the formal foundations, we outline its use considering a small real-world example. In addition, we discuss how monitoring knowledge can be obtained using the operational design domains and how to handle faults in the monitoring framework. Future work will include implementing the proposed approach in an autonomous driving demonstrator and conceiving an experimental evaluation.

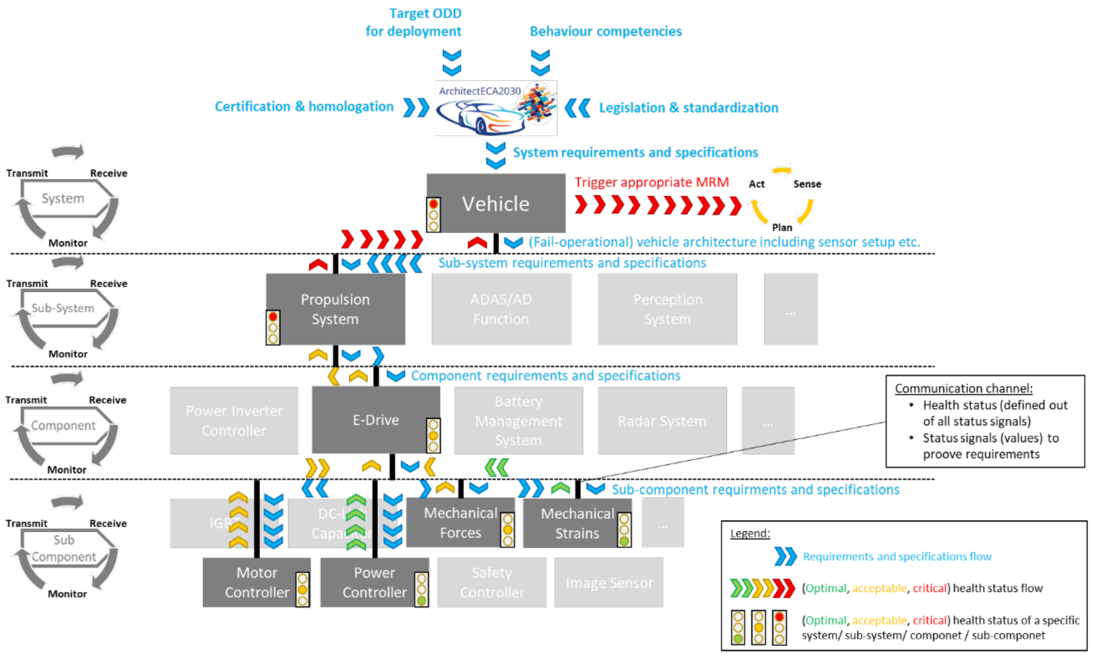
Acknowledgement

ArchitectECA2030 receives funding within the Electronic Components and Systems For European Leadership Joint Undertaking (ESCEL JU) in collaboration with the Euro- pean Union's Horizon2020 Framework Programme and National Authorities, under grant agreement n° 877539. All ArchitectECA2030 related communication reflects only the author's view and the Agency and the Commission are not responsible for any use that may be made of the information it contains. The work was partially funded by the Austrian Federal Ministry of Climate Action, Environment, Energy, Mobility, Innovation and Technology (BMK) under the program ”ICT of the Future” project 877587.

References

[1] *EU ADS Implementing Act*. European Commission, Brussels, Belgium, 2022.

[2] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, T. Weissgerber, K. Bengler, R. Bruder, F. Flemisch, and H. Winner. Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance. *IET Intelligent Transport Systems*, 8(3):183–189, May 2014.



Target ODD for deployment
Behaviour competencies
Certification & homologation
ArchitectECA2030
Legislation & standardization
System requirements and specifications
Vehicle
Trigger appropriate MRM
Act
Sense
Plan
(Fail-operational) vehicle architecture including sensor setup etc.
Sub-system requirements and specifications
Propulsion System
ADAS/AD Function
Perception System
...
Component requirements and specifications
Power Inverter Controller
E-Drive
Battery Management System
Radar System
Communication channel:
Health status (defined out of all status signals)
Status signals (values) to proove requirements
Sub-component requirments and specifications
IGBT
DC-Link Capacitor
Mechanical Forces
Mechanical Strains
Motor Controller
Power Controller
Safety Controller
Image Sensor
Transmit
Receive
System
Sub-System
Component
Sub Component
Monitor
Legend:
Requirements and specifications flow
(Optimal, acceptable, critical) health status flow
(Optimal, acceptable, critical) health status of a specific system/ sub-system/ componet / sub-componet

Figure 3: Requirements vs. health status flow.

[3] ISO. Road vehicles – Functional safety, 2011.

[4] ISO/SAE PAS 22736:2021. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. Technical report, ISO/TC 204 Intelligent transport systems, Geneva, Switzerland, 2021.

[5] Nidhi Kalra and Susan M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94:182 – 193, 2016.

[6] Philip Koopman and Michael Wagner. Challenges in autonomous vehicle testing and validation. *SAE Int. J. Trans. Safety*, 4:15–24, 04 2016.

[7] Till Menzel, Gerrit Bagschik, and Markus Maurer. Scenarios for development, test and validation of automated vehicles. In *arXiv:1801.08598*, 2018. Accepted at the 2018 IEEE Intelligent Vehicles Symposium.

[8] On-Road Automated Driving (ORAD) committee. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. Technical report, SAE International.

[9] Fabian Schuldt, Andreas Reschka, and Markus Maurer. A method for an efficient, systematic test case generation for advanced driver assistance systems in virtual environments. In Hermann Winner, Gunter Prokop, and Markus Maurer, editors, *Automotive Systems Engineering II*. Springer International Publishing AG, 2018.

[10] Eric Thorn, Shawn Kimmel, and Michelle Chaka. A framework for automated driving system testable cases and scenarios. Technical Report DOT HS 812 623, National Highway Traffic Safety Administration, Washington, DC, September 2018.

[11] Franz Wotawa. Testing autonomous and highly configurable systems: Challenges and feasible solutions. In D. Watzenig and M. Horn, editors, *Automated Driving*. Springer International Publishing, 2016. DOI 10.1007/978-3-319-31895-0 22.

[12] Franz Wotawa and Horst Lewitschnig. Monitoring hierarchical systems for safety assurance. In *IDC*, volume 1026 of *Studies in Computational Intelligence*, pages 331–340. Springer, 2021.

[13] Franz Wotawa, Bernhard Peischl, Florian Klück, and Mihai Nica. Quality assurance methodologies for automated driving. *Elektrotechnik & Informationstechnik*, 135(4–5), 2018. https://doi.org/10.1007/s00502-018-0630-7.

[14] Zhitao Xiong, Hamish Jamson, Anthony G. Cohn, and Oliver Carsten. *Ontology for Scenario Orchestration (OSO): A Standardised Scenario Description in Driving Simulation*. ASCE Library, 2013.