

THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Accelerating DevSafeOps for Autonomous Driving via Generative-AI and Synthetic Data

ALI NOURI



Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden, 2026

**Accelerating DevSafeOps for
Autonomous Driving via
Generative-AI and Synthetic Data**

ALI NOURI

© Ali Nouri, 2026
except where otherwise stated.
All rights reserved.

ISBN 978-91-8103-470-7

Doktorsavhandlingar vid Chalmers tekniska högskola, Ny serie nr 5927.

ISSN 0346-718X

Department of Computer Science and Engineering
Division of Interaction Design and Software Engineering
Chalmers University of Technology | University of Gothenburg
SE-412 96 Göteborg,
Sweden
Phone: +46(0)31 772 1000

Cover image: DevSafeOps cycles for autonomous driving software development, accelerated by Generative AI and synthetic data generation of rare events (black swans) using 3D Gaussian Splatting. The overall cover design and composition were created by the author. The illustration depicting the black swan scenario was generated using ChatGPT.

Printed by Chalmers Digitaltryck,
Gothenburg, Sweden 2026.

“Law 1: A robot may not injure a human being or, through inaction, allow a human being to come to harm.”
- Handbook of Robotics, 56th Edition, 2058 A.D.

Abstract

Background: The safety of Autonomous Driving (AD) remains a barrier to its widespread adoption, as evidenced by recent incidents. Factors such as a complex environment, evolving technologies, and shifting regulatory and customer requirements necessitate continuous monitoring and improvement of AD software. This is a process that may favor software and system engineering supported by DevOps. The iterative nature of the DevOps process is crucial, serving two purposes: satisfying customer demands through continuous improvement of the function and providing a framework for timely responses to unknown bugs or incidents. However, any update to the software must follow rigorous safety processes prescribed by standards, regulations, and the state of the art in industry. Incorporating these safety activities into the DevOps forms an iterative process called DevSafeOps. These necessary activities are vital for safety assurance, and may inherently lead to a compromise in rapidity.

Research Goal: In this work, we identify the challenges of rapid DevSafeOps in AD development and explore existing solutions. Subsequently, we propose multiple approaches for accelerating safety analysis, requirements engineering, code generation, and synthetic data generation in DevSafeOps cycles.

Methods: Diverse research methods are utilized to address each research objective. Interview studies and a systematic literature review are conducted to identify the challenges, research gaps, and existing approaches. Then, design science, interview study, case study, and experimentation are employed to design and evaluate new approaches to address our research goal.

Results: Initially, the challenges and research gaps related to each essential activity for the safety of automated driving are identified (Papers A and B), together with the proposed solutions presented in the literature (Paper B). Two approaches are proposed to accelerate the design phase (i.e., analysis and requirements engineering) as an initial step in DevSafeOps. We adapt System Theoretic Process Analysis (STPA) to enable distributed development within automotive system engineering (Paper C). As an alternative approach, a Large Language Model (LLM)-based multi-agent Hazard Analysis and Risk Assessment (HARA) prototype is proposed and evaluated to enable automation (Papers D and E). The rule-based software-implementation phase is accelerated through LLM-based code generation conducted through a conversation in a simulation environment (Papers F and G). To connect the design phase to operation, a vision-language model (VLM) is employed to enable rapid closed-loop DevSafeOps (Paper H). In parallel, a complementary solution is introduced to address the specific needs of Machine Learning (ML)-based software development. As data act as requirements for ML, it is crucial to generate data in a controlled manner to obtain a sufficiently sized population of critical scenarios for training the expected behavior. Hence, through synthetic data generation using three-dimensional Gaussian Splatting (3DGS), ML-based software development in the DevSafeOps cycle is covered (Paper I).

Conclusions: This thesis first identifies multiple challenges in achieving rapid DevSafeOps in AD development and then proposes several approaches for addressing these challenges across different phases of the DevSafeOps

cycle. To accelerate the design phase, we introduce an adaptation of STPA for multiparty distributed development and employ multi-agent LLMs as a parallel approach for HARA. We further examine how LLMs and VLMs can support safety concept design, code generation, and monitoring activities with reduced engineer involvement, while defining necessary safeguarding measures. Finally, we investigate 3DGS as an effective and rapid DataOps technique within DevSafeOps, enabling improved data generation and augmentation for ML-based software development.

Keywords

Autonomous Vehicles, DevOps, DevSafeOps, Safety, Requirements Engineering, Hazard Analysis and Risk Assessment, Large Language Model, STPA, Code Generation, 3D Gaussian Splatting, Synthetic Data Generation, Vision Language Model

Acknowledgment

First and foremost, I am sincerely grateful to my supervisor, Prof. Christian Berger, for his continuous support, guidance, and insightful feedback in shaping my research. Your patience, encouragement, and availability during times of stress and difficulty have been invaluable to me.

I extend my heartfelt thanks to my industrial supervisors, Dr. Håkan Sivencrona (formerly at Zenseact, now at Volvo Cars) and Dr. Fredrik Törner. Håkan, you have shown me the bigger picture and have been a supportive mentor, as well as sharing so many brilliant ideas. Your expertise and experience have provided me with a unique perspective. Fredrik, thank you for trusting me and guiding me from the very start; without your support, effort, and guidance, this project would not have been initiated or sustained.

I am also grateful to Behnam Safakhah, Matthias Eng, and Lena Westervall for their excellent support throughout my research. You always trusted me and enabled me by providing everything I needed for my work.

I am very grateful to Prof. Philip Koopman for his insightful feedback on my licentiate thesis, which served as a foundation for this thesis. I also extend my sincere thanks to my examiner, Prof. Jan Bosch, for his valuable feedback, evaluation, and support throughout my doctoral studies.

I am also deeply grateful to Volvo Cars for trusting me and providing the opportunities that made this research possible. I gratefully acknowledge the financial support from Sweden's Innovation Agency and WASP.

My deepest appreciation goes to my parents, Forouzandeh and Masoud, whose selfless support gave me the freedom to explore and pursue my dreams. Your belief in me has been a constant source of motivation. Your love, sacrifices, and encouragement have been the foundation upon which I stand today. To my sister, Sara, thank you for being my first friend and for your encouragement.

Most importantly, I would like to express my deepest gratitude to my beloved wife, Ghazal. Your patience, understanding, and unwavering support have been my greatest source of strength throughout this journey. Thank you for being by my side every step of the way.

Ali Nouri

Gothenburg, August 2026

List of Papers

Appended Papers

This thesis is based on the following papers:

- [Paper A] **A. Nouri**, C. Berger, F. Törner, *An Industrial Experience Report about Challenges from Continuous Monitoring, Improvement, and Deployment for Autonomous Driving Features* *Proceedings of the 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Gran Canaria, Spain, 2022, <https://ieeexplore.ieee.org/abstract/document/10011483>.
- [Paper B] **A. Nouri**, B. Cabrero-Daniel, F. Törner, C. Berger, *The DevSafeOps Dilemma: A Systematic Literature Review on Rapidity in Safe Autonomous Driving Development and Operation* *Journal of Systems and Software*, <https://www.sciencedirect.com/science/article/pii/S0164121225002249>.
- [Paper C] **A. Nouri**, C. Berger, F. Törner, *On STPA for Distributed Development of Safe Autonomous Driving: An Interview Study* *Proceedings of the 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Durres, Albania, 2023, <https://ieeexplore.ieee.org/abstract/document/10371692>.
- [Paper D] **A. Nouri**, B. Cabrero-Daniel, F. Törner, H. Sivencrona, C. Berger, *Welcome Your New AI Teammate: On Safety Analysis by Leashing Large Language Models* *CAIN '24, Association for Computing Machinery*, 2024, <https://dl.acm.org/doi/abs/10.1145/3644815.3644953>.
- [Paper E] **A. Nouri**, B. Cabrero-Daniel, F. Törner, H. Sivencrona, C. Berger, *Engineering Safety Requirements for Autonomous Driving with Large Language Models* *RE 2024, IEEE 32nd International Requirements Engineering Conference*, <https://ieeexplore.ieee.org/abstract/document/10628478>.
- [Paper F] **A. Nouri**, B. Cabrero-Daniel, Z. Fei, K. Ronanki, H. Sivencrona, C. Berger, *Large Language Models in Code Co-generation for Safe*

Autonomous Vehicles

SAFECOMP 2025, 44th International Conference on Computer Safety, Reliability, and Security, Stockholm, Sweden,

https://link.springer.com/chapter/10.1007/978-3-032-01241-8_13.

- [Paper G] **A. Nouri**, J. Andersson, K. D. J. Hornig, Z. Fei, E. Knabe, H. Sivencrona, B. Cabrero-Daniel, C. Berger, *On Simulation-Guided LLM-based Code Generation for Safe Autonomous Driving Software EASE 2025 — 29th International Conference on Evaluation and Assessment in Software Engineering, Istanbul, Türkiye,*

<https://dl.acm.org/doi/10.1145/3756681.3756987>

- [Paper H] **A. Nouri**, *Accelerating DevSafeOps through Large Language Models: Opportunities and Challenges*
Submitted, under review in Journal of Systems and Software.

- [Paper I] **A. Nouri**, Y. Zhang, Y. Zhang, T. Bouraffa, Z. Fei, Z. Han, H. Sivencrona, A. Heyden, *From Concept to Capability: Evaluating 3D Gaussian Splatting for Synthetic Scene Editing in Autonomous Driving SAFECOMP 2026, 45th International Conference on Computer Safety, Reliability and Security, Valencia, Spain,*

https://link.springer.com/chapter/10.1007/978-3-032-34867-8_11.

Other Papers/Patents

The following papers and patents were published during my PhD studies, or are currently in submission/under revision, but not appended to this thesis.

- [a] J. Gu, B. Cabrero-Daniel, **A. Nouri**, L. Armini, C. Berger, *PCICF: A Pedestrian Crossing Identification and Classification Framework 48th IEEE/ACM International Conference on Software Engineering (ICSE), 2026,*
<https://arxiv.org/abs/2509.24386>
- [b] B. Cabrero-Daniel, Y. Fazelidehkordi, **A. Nouri**, *How Can Generative AI Enhance Software Management? Is It Better Done than Perfect? In: Nguyen-Duc, A., Abrahamsson, P., Khomh, F. (eds) Generative AI for Effective Software Development. Springer, Cham.*
https://link.springer.com/chapter/10.1007/978-3-031-55642-5_11.
- [c] T. Bouraffa, E. Kjellberg Carlson, E. Wessman, **A. Nouri**, P. Lamart, C. Berger, *Comparing Optical Flow and Deep Learning to Enable Computationally Efficient Traffic Event Detection with Space-Filling Curves 27th IEEE International Conference on Intelligent Transportation Systems,*
<https://ieeexplore.ieee.org/abstract/document/10919665>.

-
- [d] M. R. Martínez Rodríguez, **A. Nouri**, Z. Fei, M. Hedblom, *An LLM and Embeddings-based Multi-Agent System for Knowledge Graph Construction and Verification*
26th International Conference on Principles and Practice of Multi-Agent Systems (PRIMA),
https://link.springer.com/chapter/10.1007/978-3-032-13562-9_31.
 - [e] B. Cabrero-Daniel, M. C. Kaya, **A. Nouri**, M. Mohamad, C. Berger, *Fidelity of Simulations for Verification & Validation of Automated Driving: A Practitioners' Perspective*
Journal of Software Engineering for Autonomous Systems, 2026,
<https://doi.org/10.55060/j.jseas.260218.001>.
 - [f] E. Jelačić, R. Gu, C. Secoleanu, N. Xiong, P. Backeman, T. Secoleanu, Z. Fei, **A. Nouri**, *HASCO: A Hybrid AI Simulation Compiler for Semantic Accident Reconstruction*
Proceedings of the 30th Ada-Europe International Conference on Reliable Software Technologies (AEiC 2026).
 - [g] **A. Nouri**, Z. Fei, *Real-time vehicle identification for adaptive behavior in self-driving vehicles*
US20260042460A1, United States Patent and Trademark Office, Volvo Car Corporation, published on 12 February 2026, pending,
<https://patents.google.com/patent/US20260042460A1>.
 - [h] **A. Nouri**, Z. Fei, *Training and operating an object detecting system for a vehicle*
EP4407482A1, European Patent Office, Volvo Car Corporation, published on 31 July 2024, pending,
<https://patents.google.com/patent/EP4407482A1>.
 - [i] **A. Nouri**, B. Safakhah, H. Sivencrona, *Method for managing a user demand-based operational design domain expansion for an automated driving system of a vehicle*
US20250128729A1, United States Patent and Trademark Office, Zenseact AB, published on 24 April 2025, pending,
<https://patents.google.com/patent/US20250128729A1>.
 - [j] **A. Nouri**, H. Sivencrona, *Dynamic adaptation of an operational design domain for an automated driving system of a vehicle*
US20250368226A1, United States Patent and Trademark Office, Zenseact AB, published on 4 December 2025, pending,
<https://patents.google.com/patent/US20250368226A1>.
 - [k] **A. Nouri**, Z. Fei, *Real-time identification of vehicle features for adaptive behavior in self-driving vehicles*
US20260042469A1, United States Patent and Trademark Office, Volvo Car Corporation, published on 12 February 2026, pending,
<https://patents.google.com/patent/US20260042469A1>.

- [l] Z. Fei, **A. Nouri**, *Methods and systems for adapting an automation system in a vehicle*
US20250170975A1, United States Patent and Trademark Office, Volvo Car Corporation, published on 29 May 2025, pending,
<https://patents.google.com/patent/US20250170975A1>.
- [m] Z. Fei, **A. Nouri**, *Crowdsourcing image annotation using grid image user authentication systems*
US20260032006A1, United States Patent and Trademark Office, Volvo Car Corporation, published on 29 January 2026, pending,
<https://patents.google.com/patent/US20250368226A1>.

Research Contribution

I was the main contributor in conceiving the idea, designing and planning the experiments/research methodology, data collection, data analysis, and writing of all appended papers (Papers A to I) and this thesis.

In paper a, I contributed in discussions, writing, and reviewing the paper. In paper b, I was consulted on the idea, and then contributed to writing, and reviewing. In paper c, I was the industrial supervisor of master thesis students, and contributed in writing, and reviewing the paper. In paper d, I was the industrial supervisor of master thesis students, and contributed in idea creation, writing, and reviewing the paper. In paper e, I contributed to designing the interview protocol, conducting the interviews, and writing and reviewing the paper. In paper f, I contributed to the initial idea discussions, its refinement, and to writing and reviewing.

In patents g, h, i, j, and k, I was the main contributor in conceiving the idea, and design. In patents l and m, I contributed with discussion on initial idea and improvement of it.

Relation to Licentiate Thesis

This doctoral thesis builds upon the author’s licentiate thesis:

Ali Nouri, “Accelerating the Design Phase: Towards DevSafeOps for Autonomous Driving Software,” Licentiate Thesis, Chalmers University of Technology, 2024.

The following sections are partially based on or adapted from the licentiate thesis and were revised to complete the overarching narrative:

- Chapter 1.1, Section 1.2.1, Section 1.2.2, Section 1.3.1, Section 1.3.2, Section 1.4.1, Section 1.6.1, Section 1.7, Section 2.1, Section 2.2.1, Section 2.2.2, Section 2.2.3, Section 2.2.5, and Section 2.3
- Papers A, B (version under review), C, D, and E
- Figures 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.10, 1.12 and H.3

The following parts in the doctoral thesis are newly added sections and extend the author’s previous work through:

- Chapter 1.2, Sections 1.2.3, Section 1.2.4, Section 1.2.5, Section 1.2.6, Section 1.2.7, Section 1.3.3, Section 1.4.2, Section 1.4.3, Section 1.4.4, Section 1.6.2, Section 1.6.3, Section 1.9, Section 2.2, Section 2.2.4, Chapter 4, Chapter 5
- Papers B (final version), F, G, H, and I
- Figures 1.8, 1.9, 1.11, 1.13, 2.1, 2.2, 2.3, 3.2 and 3.1

In summary, this doctoral thesis provides a comprehensive investigation of DevSafeOps challenges and proposes approaches to address them, covering the industrial engineering cycle design phase (primarily covered in the author’s Licentiate thesis), implementation, testing, and operation.

List of Abbreviations

3DGS	3D Gaussian Splatting
AI	Artificial Intelligence
ADAS	Advanced Driver Assistance Systems
ADS	Autonomous Driving Systems
ASIL	Automotive Safety Integrity Level
AUR	Absence of Unreasonable Risk
CA	Control Action
DevOps	Development and Operations
DDT	Dynamic Driving Task
DMS	Driver Monitoring Systems
DMV	Department of Motor Vehicles
E2E	End-to-End
ECU	Electronic Control Unit
FMEA	Failure Mode and Effect Analysis
FTA	Fault Tree Analysis
Gen-AI	Generative Artificial Intelligence
HARA	Hazard Analysis & Risk Assessment
HiL	Hardware-in-the-loop
HMI	Human-Machine Interface
KPI	Key Performance Indicator
LLM	Large Language Models
ML	Machine Learning
MLOps	Machine Learning Operations
NHTSA	National Highway Traffic Safety Administration

ODD	Operational Design Domain
OEM	Original Equipment Manufacturer
OTA	Over-the-Air
PRB	Positive Risk Balance
SiL	Software-in-the-Loop
SLR	Systematic Literature Review
STPA	System Theoretic Process Analysis
TIM	Traceability Information Model
UNECE	United Nations Economic Commission for Europe
VLA	Vision-Language-Action

Contents

Abstract	iii
Acknowledgement	v
List of Publications	vii
List of Abbreviations	xiii
I Summary	1
1 Introduction & Background	3
1.1 Autonomous Driving System (ADS)	3
1.1.1 Safety of ADS	4
1.1.2 Defining Safety	5
1.1.3 ADS Technology	7
1.2 ADS Development Process	8
1.2.1 Automotive Safety Development Process	8
1.2.2 Abstraction Levels	10
1.2.3 Item (Concept Phase)	10
1.2.4 Function	12
1.2.5 System	13
1.2.6 Software - Modular Approach	13
1.2.7 Software - “End-to-End” Approach	14
1.3 DevSafeOps	15
1.3.1 Software Update and DevOps	15
1.3.2 Safety of DevOps	16
1.3.3 MLOps	16
1.4 Methods and Tools	17
1.4.1 System Theoretic Process Analysis (STPA)	17
1.4.2 Simulation Environment	20
1.4.3 Generative Artificial Intelligence (Gen-AI)	20
1.4.4 Synthetic Data Generation	22
1.5 Motivation and Problem Domain	22
1.6 Potential Approaches	23
1.6.1 Distributed Development	23

1.6.2	Gen-AI for Systems Engineering Automation	23
1.6.3	3DGS	24
1.7	Research Goal and Questions	24
1.8	Scope	25
1.9	Structure of the Thesis	25
2	Research Design	29
2.1	Research Paradigms	29
2.2	Prescribed Methods	30
2.2.1	Systematic Literature Review	32
2.2.2	Interview Study	33
2.2.3	Case Studies	34
2.2.4	Experimentation	34
2.2.5	Design Science	39
2.3	Methodological Considerations	39
2.3.1	Level of Researcher's Involvement	40
2.3.2	Generalizability (Universal vs. Particular)	40
2.3.3	Ethical Considerations	40
3	Summary of Included Papers	41
3.1	Paper A - An Industrial Experience Report about Challenges from Continuous Monitoring, Improvement, and Deployment for Autonomous Driving Features	44
3.2	Paper B - The DevSafeOps Dilemma: A Systematic Literature Review on Rapidity in Safe Autonomous Driving Development and Operation	44
3.3	Paper C - Using STPA for Distributed Development of Safe Autonomous Driving: An Interview Study	45
3.4	Paper D - Welcome Your New AI Teammate: On Safety Analysis by Leashing Large Language Models	46
3.5	Paper E - Engineering Safety Requirements for Autonomous Driving with Large Language Models	46
3.6	Paper F - Large Language Models in Code Co-generation for Safe Autonomous Vehicles	47
3.7	Paper G - On Simulation-Guided LLM-based Code Generation for Safe Autonomous Driving Software	48
3.8	Paper H - Accelerating DevSafeOps through Large Language Models: Opportunities and Challenges	49
3.9	Paper I - From Concept to Capability: Evaluating 3D Gaussian Splatting for Synthetic Scene Editing in Autonomous Driving	51
4	Discussion	53
4.1	Contributions	53
4.2	Threats to Validity	56
4.2.1	Conclusion Validity	56
4.2.2	Construct Validity	56
4.2.3	Internal Validity	57

4.2.4 Dependability and Credibility	57
4.2.5 Confirmability	58
4.2.6 External Validity	58
4.2.7 Residual Risks	59
5 Conclusions and Future Research	61
References	65
 II Appended Papers	 75
Paper A - An Industrial Experience Report about Challenges from Continuous Monitoring, Improvement, and Deployment for Autonomous Driving Features	
Paper B - The DevSafeOps Dilemma: A Systematic Literature Review on Rapidity in Safe Autonom- ous Driving Development and Operation	
Paper C - On STPA for Distributed Development of Safe Autonom- ous Driving: An Interview Study	
Paper D - Welcome Your New AI Teammate: On Safety Analysis by Leashing Large Language Models	
Paper E - Engineering Safety Requirements for Autonomous Driving with Large Language Models	
Paper F - Large Language Models in Code Co-generation for Safe Autonomous Vehicles	
Paper G - On Simulation-Guided LLM-based Code Generation for Safe Autonomous Driving Software	
Paper H - Accelerating DevSafeOps through Large Language Models: Opportunities and Challenges	
Paper I - From Concept to Capability: Evaluating 3D Gaussian Splatting for Synthetic Scene Editing in Autonomous Driving	

Part I

Summary

Chapter 1

Introduction & Background

1.1 Autonomous Driving System (ADS)

SAE J3016 [2] defines six levels of automation, ranging from “no driving automation (Level 0)” to unconditional (i.e., not ODD-specific) “full driving automation (Level 5)”. In such a technology-driven approach [3], these levels are determined by the degree of automation and the user’s role and responsibility during ADS [4] activation and in fallback scenarios. The human-centric approach is an alternative that focuses on human needs and abilities and then designs the system based on them [3]. Given the varying readiness and capability for supervision among users, these levels can be categorized into supervised and unsupervised [3], as shown in Fig. 1.1.

In *supervised* modes, the function performs interventions to support the driver (i.e., Advanced Driver Assistance Systems or ADAS) or controls both lateral and longitudinal motion while the driver is responsible for the Dynamic Driving Task (DDT). Driver Monitoring Systems (DMS) monitor the driver to ensure their attentiveness and supervision (e.g., hands-on or eyes-on). ADAS functions offer warnings, steering assistance, or braking interventions to help the driver mitigate or prevent incidents.

These systems play a crucial role in progressing toward Vision Zero [4], which aims for no fatalities or severe injuries within the road traffic system. For instance, driver distraction is the main cause of rear-end collisions occurring at speeds up to 30 km/h [5]. This led the industry to introduce active safety functions, and later some Original Equipment Manufacturers (OEMs) offered them as a standard feature in all passenger car models (e.g., Volvo Cars’ City Safety [5]). A subsequent study, analyzing data from 2010 to 2014 from Sweden, showed a 28% reduction in the rate of rear-end frontal collisions compared with cars not equipped with this system [6]. Similar conclusions were drawn for the reduction of car-to-pedestrian collisions by using other ADAS functions [7].

In *unsupervised* mode, the vehicle fully controls the driving task. If a

¹J3016 [2] uses the term ADS for Automated Driving System, covering Levels 3 to 5. However, this study focuses on Level 4/5 (i.e., unsupervised capability), often referred to as an Autonomous Driving System.

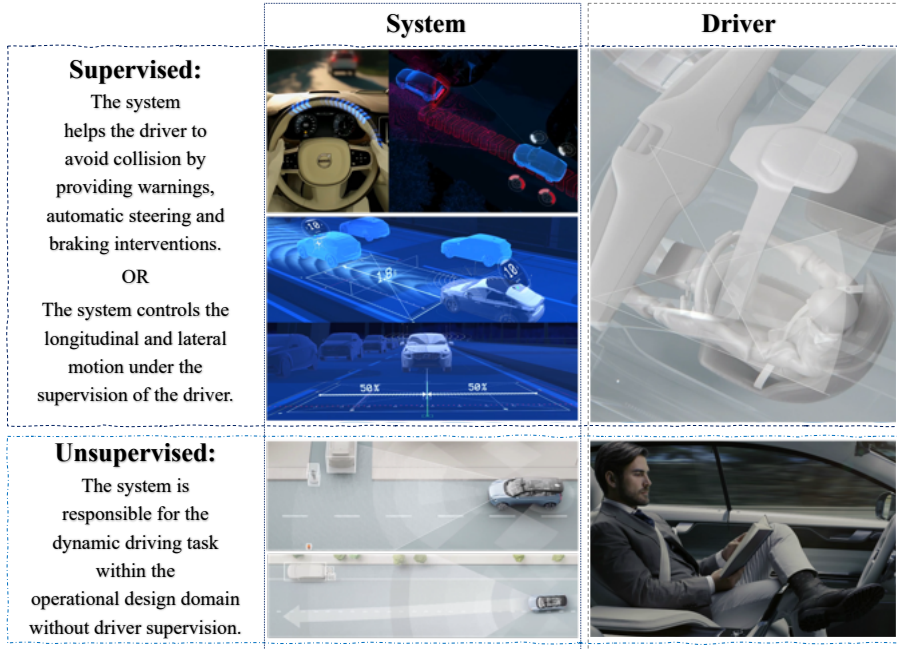


Figure 1.1: Human-centric automation levels. Figure created by the author, incorporating images from [1]

fallback is needed, the system will request a takeover but will not rely solely on the driver, as they might not be available. Thus, ADS is fully responsible for maintaining the safety of the driving task. While ADAS functions such as Automatic Emergency Braking (AEB) have demonstrated improvements in road safety, supported by statistics [2], the overall benefit of ADS on traffic safety is still theoretical. This highlights research gaps in both ensuring the safety of ADS and understanding its effect on traffic safety.

1.1.1 Safety of ADS

Despite the fact that the development of ADS started back in the 1980s [8], its safety remains a challenge, as there are still loss events. For instance, a pedestrian was severely injured in a mishap involving a robotaxi [9]. The pedestrian was pushed into the robotaxi's lane as a result of an initial hit-and-run collision with another vehicle. The ADS ran over the pedestrian, decelerated initially, and then moved forward while pulling the pedestrian.

The investigation [9] showed that the cause was neither hardware nor software failure. As mentioned in the investigation report [9], the ADS could initially detect both the pedestrian and the adjacent vehicle and even predict a potential collision between them, but it could not predict either of them entering

²Real-world benefits of crash avoidance technologies: summary of IIHS-HLDI findings, accessed April 20, 2026, <https://www.iihs.org/topics/advanced-driver-assistance>

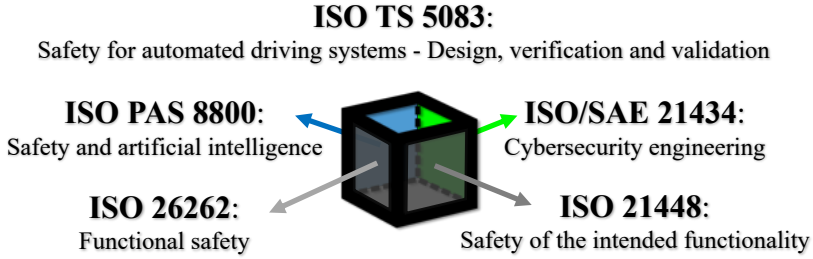


Figure 1.2: Depicting ISO standards for road vehicles relevant to the safety of ADS. ISO 5083 is specific to ADS applications and address gaps in relation to other standards or refer to them when applicable. Other standards mentioned are not specific to ADS applications, although ISO 21448 and ISO PAS 8800 primarily address challenges related to ADS.

its lane. After the collision between the other vehicle and the pedestrian, the ADS classified the pedestrian as an unknown object and started deceleration. The collision detection system incorrectly detected the pedestrian’s position as being on the side. As a consequence, the vehicle made the wrong maneuver instead of stopping in the lane.

Weak recognition and response to a nearby incident, as well as an inaccurate post-crash world model, are some of the technical issues and challenges as highlighted by Koopman [10]. There are multiple hints for a human driver in this situation, such as reducing speed when predicting a nearby incident even if both remain in the adjacent lane. However, as the system was following the design, it failed to adapt the trajectory based on this unforeseen event. Investigating these incidents sheds light on the need for novel or improved approaches in the design of ADS that enable the system to adapt to these unforeseen, complex, and numerous events.

Introducing the system without sufficient confidence or necessary fallback strategies can lead not only to safety risks but also to delays in deployment or even termination of the project. For instance, failing to assure public safety during the operation of ADS (without a human operator in the vehicle) led to the immediate revocation of ADS deployment and testing permits by the California Department of Motor Vehicles (DMV)³

1.1.2 Defining Safety

To assure and subsequently argue the safety of ADS, the term ‘safety’ must initially be defined both qualitatively and quantitatively. This definition is then broken down into more granular requirements for each building block of the ADS, which are used as criteria for pass or fail in the verification and validation phases.

³DMV STATEMENT ON CRUISE LLC SUSPENSION, accessed April 20, 2026, <https://www.dmv.ca.gov/portal/news-and-media/dmv-statement-on-cruise-llc-suspension/>

Several stakeholders are involved in defining the safety criteria, such as legislators. For instance, the United Nations Economic Commission for Europe (UNECE) published UN Regulation No. 157 [11], which defines qualitative and quantitative criteria to ensure the safety of Automated Lane Keeping Systems (ALKS). Additionally, UN Regulation No. 157 indirectly recommends compliance with ISO 26262 [12], ISO 21448 [13], and ISO 21434 [14] by requiring a competent auditor and assessor in these standards. Each standard defines safety criteria in a specific way, depending on its scope. For instance, ISO/TS 5083 [15] defines it as the “Absence of Unreasonable Risk (AUR).” ISO 26262, ISO 21448, and ISO 8800 [16] refine the definition by adding a second part, making it more specific based on the root causes they target:

ISO 26262: “Absence of unreasonable risk due to hazards caused by **mal-functioning behavior of E/E systems**”

ISO 21448: “Absence of unreasonable risk due to hazards resulting from **functional insufficiencies of the intended functionality or its implementation.**”

ISO 8800: “Absence of unreasonable risk due to **errors of the AI system.**”

There are several concepts for defining the quantitative safety criterion, such as Positive Risk Balance (PRB) [17]. This high-level quantitative safety criterion can be broken down and used as acceptance criteria for requirements addressing ISO 21448 [13]. PRB expects the ADS to “cause fewer crashes on average compared to those made by drivers” [17]. However, questions such as “How much less?” (e.g., X% less) and “which driver?” (e.g., an attentive driver) need to be answered to calculate the quantitative target [18].

As highlighted in the paper “Redefining Safety for Autonomous Vehicles” by Koopman and Widen [19], both concepts of AUR and PRB fall short when applied to ADS. The authors point out that AUR fails to account for scenarios requiring moral and contextual understanding and might lead to risk redistribution to other road users, including emergency vehicles and vulnerable communities. In addition, it is difficult to define a comparable baseline for PRB between human drivers and ADS. These issues underscore the need for updated safety definitions [19].

ISO 26262 [12] (functional safety, or so-called FuSa) prescribes remedies to avoid or mitigate systematic failures in hardware and software, as well as random hardware faults, in the design, implementation, verification, validation, and field monitoring phase. These activities are then used in a safety case to argue the achievement of functional safety of the function.

ISO 21448 [13] (Safety of Intended Functionality or SOTIF), is a complementary standard to ISO 26262, which contains recommendations for avoiding hazardous events caused by functional insufficiencies, incorrect/inadequate Human-Machine Interface (HMI), and insufficiencies of Artificial Intelligence (AI)-based algorithms. SOTIF starts by requiring the definition of acceptance criteria, which can be both qualitative and quantitative (e.g., one event per X km). The fulfillment of each acceptance criterion is then argued based on evidence from analysis, tests, and examinations for each validation target (e.g.,

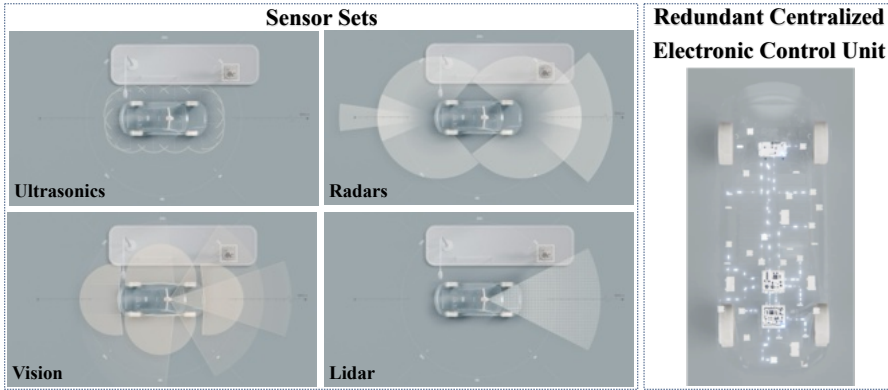


Figure 1.3: Presenting some of the diverse sensor technologies, their field of view, and redundant Electronic Control Units (ECUs). This includes the diversity of sensor technologies and redundant ECUs employed in Automotive Safety Integrity Level (ASIL) decomposition and arguing for sufficient independence between decomposed channels. Figure created by the author, incorporating images from [1]

no hazardous behavior during X hours of testing) [13]. In case any of the acceptance criteria are not being met, a functional modification is then needed, which leads to changes in design and specification.

ISO/TS 5083 [15] is the standard that results from the ‘Safety First for Autonomous Driving’ white paper [20]. These ADS-specific standards discuss about additional process aspects of ADS in relation to ISO 26262 and ISO 21448. Moreover, they contain technical and architectural recommendations for ADS. UL 4600 is another ADS-specific standard that addresses system-level aspects [21].

Using AI technology in automotive applications is not specific to ADS. However, relying on it for safety-critical tasks in a complex environment without human supervision is a new challenge for the automotive industry, which ISO 8800 [16] aims to address. It will propose a framework covering all phases from development to operation, such as data completeness and quality.

1.1.3 ADS Technology

The ultimate goal in the development of ADS is unconditioned autonomous driving capability. However, due to known safety challenges, immature technologies in ADS such as AI, and the unbounded complexity of the environment, an incremental approach is required. Therefore, there is a need to divide the scenario space and start with one that is seen as less risky, which is referred to as the Operational Design Domain (ODD). When enough confidence is gained, the ODD would be expanded until all limitations are removed. For instance, highways with structured separations, where the exposure to Vulnerable Road Users (VRU) is lower [17] can be seen as one. Others, however, decide to start

in an urban area such as complex city streets [22].

Similar to all other robotic systems, ADS also utilize the sense-plan-act paradigm [17]. Perceiving relevant static and dynamic objects, localization, and trajectory prediction of relevant surrounding objects are the main capabilities expected from sensing. [17].

Since perception relies solely on the sensors in the system, it must reliably detect dynamic objects (e.g., other road users) and static objects (e.g., road boundaries), especially those with a risk of collision [17]. Moreover, the dynamic and complex nature of the environment makes it more challenging to create the world model. For instance, false positives or negatives in object detection might lead to a collision. As shown in Fig. 1.3, diverse sensing technologies such as radar, lidar, camera, and ultrasonic, each with different detection capabilities, lead to a more reliable perception system due to the complementary and redundant information received by each [23]. Diversity can be used in arguing for independence from systematic failures or technical limitations between two sensor clusters when a safety requirement is decomposed between them as two parallel channels to satisfy the same capability [12]. Diversity and redundancy requirements are not limited to the sensor technology but include the entire chain of components involved, such as the power supply and computation unit. The sensor fusion component analyzes, evaluates, and arbitrates between diverse received information from sensor clusters, and creates a relevant representation of the world around the ego vehicle [23].

Planning is responsible for creating a “collision-free and lawful driving plan” [17]. Moreover, it shall avoid being exposed to unsafe situations, for instance, through precautionary behaviors. Correctly executing the trajectory decided by the plan and communicating with other road users, including the driver, are the main capabilities of the actuators [17]. The trajectory execution is carried out through lateral and longitudinal motions using actuation systems such as steering, braking, and propulsion.

1.2 ADS Development Process

1.2.1 Automotive Safety Development Process

As presented in Fig. 1.4, the development starts with specification and design. In this phase, the function is defined for a specific ODD. Data collection is also conducted in this phase to use the data for training the Machine Learning (ML)-based software and for extracting relevant scenarios. Fig. 1.5 shows one example of these vehicles, which, in addition to production-intent sensor sets, contain high-precision sensors on the rooftop that can be used as reference sensors. Next, the software and hardware components are implemented and tested. They are then integrated to form the system, which is tested in different environments. At the end, assessment, audit, and certification (depending on the region) are conducted. The function is monitored in the field, and modifications might be required to improve the performance of the function or expand the ODD.

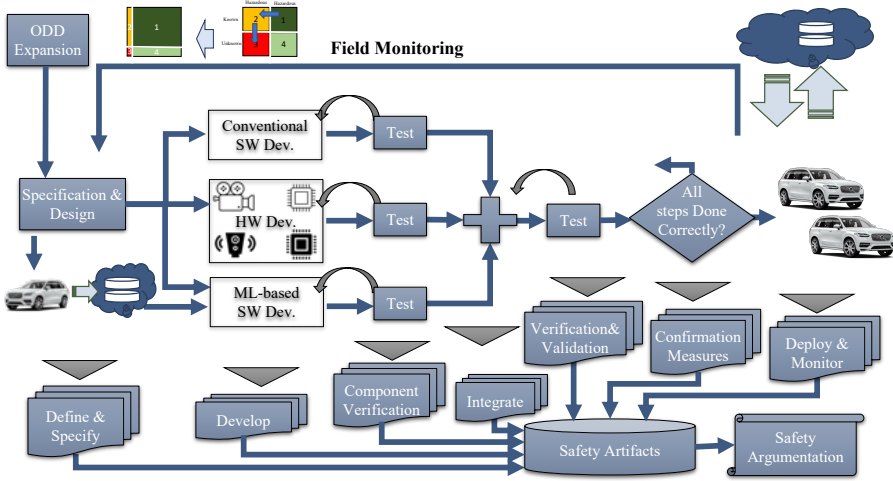


Figure 1.4: Presenting the development process for autonomous driving and its safety argumentation. It starts with specification and design, followed by implementation and testing. Subsequently, after assessment, the software will be updated in the vehicle. This process will be continuously repeated as part of ODD expansion. The elements in the figure are abstracted, with each representing an encapsulation of several activities. For instance, both clouds (top right and middle left) represent not only data storage but also activities such as data cleaning, preparation, and feature extraction, as prescribed by relevant guidelines and standards (e.g., Annex D, ISO 21448 [13]).



Figure 1.5: Presenting a data collection vehicle. The collected data is used for identifying relevant scenarios, utilized in the development of rule-based or machine learning software, and ultimately used for verification and validation.

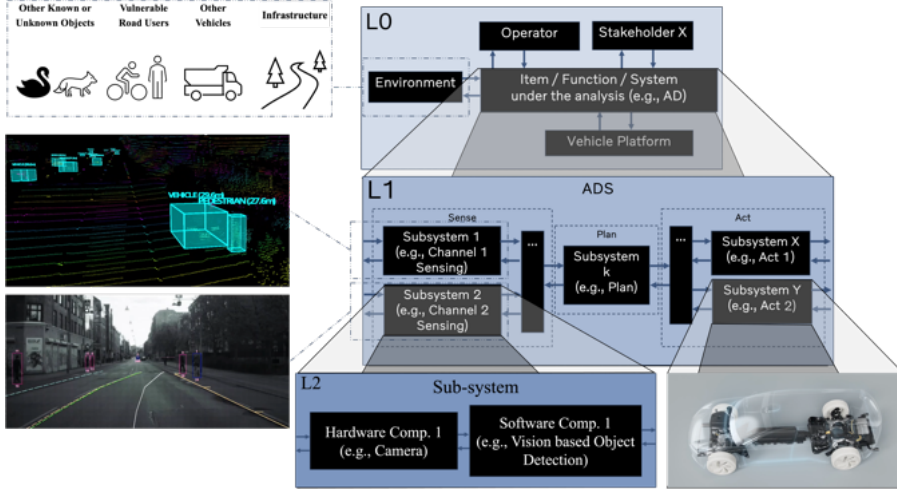


Figure 1.6: Illustrating multi-abstraction levels in a system such as AD. In Level 0, the interaction between the system, users, and environment is depicted. In Level 1, the system itself is depicted. It is important to employ abstraction to reduce the complexity of the system. As the system’s complexity increases, the number of abstraction levels also increases. The abstraction levels and their modules in the figure are for illustration purposes and do not include all modules, such as cloud-based components, other sensor technologies, or actuators. Figure created by the author, incorporating images from [1]

1.2.2 Abstraction Levels

Safety assurance of such a software-intensive system in an environment with unbounded complexity requires a modularization and abstraction approach [24]. Automotive systems engineering follows a well-established modular and multi-abstraction architecture, as illustrated in Fig. 1.6. The modules in each abstraction level are abstracted by removing or merging unnecessary characteristics of that module [25]. Each module shall be traceable to the lower or higher abstraction levels to enable impact analysis and maintainability of the system after modifications during the life cycle of the system. Fig. 1.7 presents a bird’s-eye view of the abstraction levels in the development process of the V model in the context of both ISO 26262 and ISO 21448. Each abstraction level and its relevant activities follow the flow of the V model as outlined in Sec. 1.2.3 to Sec. 1.2.6.

1.2.3 Item (Concept Phase)

“Item” is the initial abstraction level in design phase.

- “**Item Definition**” is a short description of the item, describing its functionality and ODD (Fig. 1.8). It also includes the boundary diagram, which represents the item’s interaction with the environment and stakeholders such as the driver.

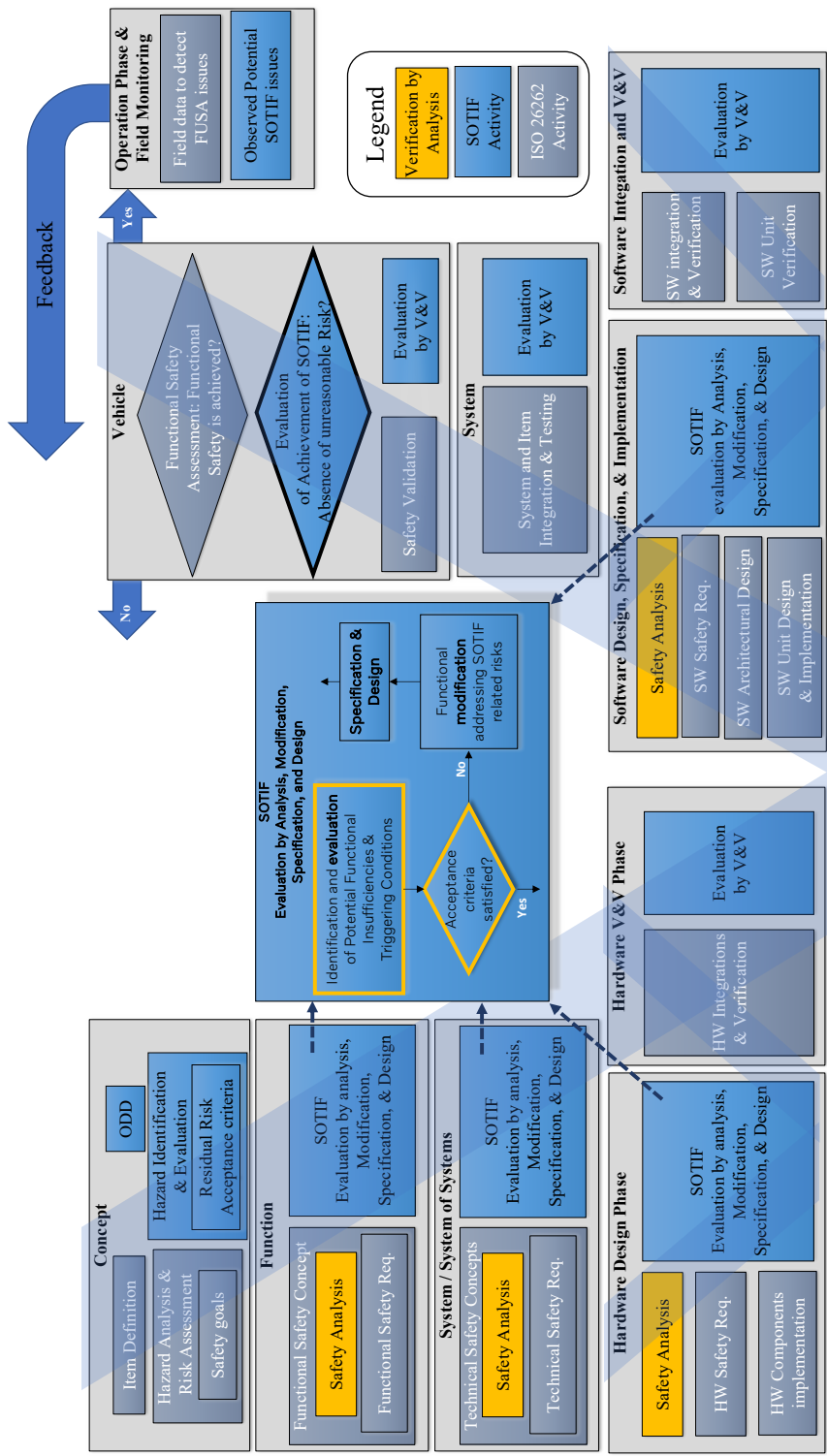


Figure 1.7: Presenting ISO 26262 and ISO 21448 activities mapped to each abstraction level. Unlike ISO 26262, the activities in ISO 21448 are not clustered based on abstraction levels, however, they can be mapped to each abstraction level. For instance, the “SOTIF evaluation by Analysis, Modification, Specification, and Design” process is repeated for each abstraction level, although the scope and level of granularity are adapted to the element under development.

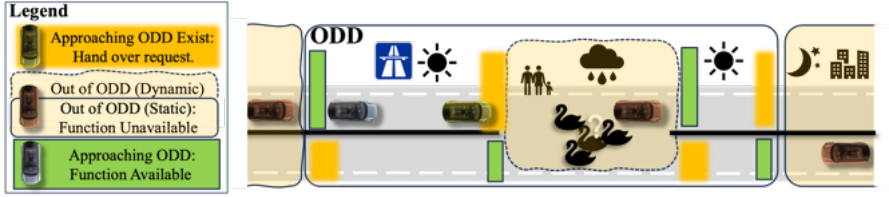


Figure 1.8: Elements of the Operational Design Domain (ODD) and the status of the system: static and dynamic aspects of the ODD are illustrated. Some parts of the ODD can be dynamically excluded if the safety risk exceeds an acceptable level due to newly identified safety-related events, system weaknesses, or limitations.

- **“Hazard Analysis & Risk Assessment”** (HARA) and Hazard Identification & Evaluation are then enhanced to identify all hazardous events that result from the occurrence of malfunctions or functional insufficiencies in a specific scenario. The risk for each hazardous event is then assessed. If a hazardous event is safety-critical, a safety goal (ISO 26262) or an acceptance criterion (ISO 21448) is specified and allocated to the hazardous event to avoid or mitigate it.

1.2.4 Function

“Function” level is the next abstraction level, which contains the functional architecture, safety analysis, and functional safety requirements gathered in a functional safety concept and SOTIF safety concept.

- **“Deductive Safety Analysis”** (Top-Down) methods are brainstorming tools used to identify the potential causes of the violation of a safety requirement under analysis. Fault Tree Analysis (FTA) is a deductive method that can be used to analyze the parent requirement and identify the potential faults (causes) in the modules at the current abstraction level.

- **“Inductive Safety Analysis”** (bottom-up) methods can be used to identify the effects of a specific failure mode on higher levels, which might lead to a violation of safety requirements. Failure Mode and Effect Analysis (FMEA) is an inductive approach that starts from the failure modes of each module at the current level and identifies the effects at higher levels, which can violate the safety goal.

- **“Functional Safety Requirements”** and SOTIF safety requirements are then designed to avoid or mitigate the unsafe failure modes of the functional blocks. This is achieved through modifications to the original functional block and the specification of functional safety requirements, which are derived from safety goals and acceptance criteria. Ideally, the functional safety requirements are technology-agnostic.

- **“Validation of Safety Goals and Safety Concepts”** lies on the right side of the V-model, where it ensures the achievement of safety goals and the appropriateness of safety concepts after system integration into the vehicle.

1.2.5 System

“**System**” level is the first level where technical aspects are designed. A function can be realized by multiple systems, and each system consists of at least a controller, and internal or external sensor(s) and actuator(s). A technical safety concept is the container for system-level safety requirements, including the safety analysis.

- “**Safety Analysis**” methods are performed similarly to those at the function level. For instance, System-Theoretic Process Analysis (STPA) may be employed as described in detail in Sec. [1.4.1](#)

- “**Technical Safety Requirements**” are specified for each component in the system and derived from the functional safety requirements. Decomposition is one of the techniques that can be used to design safety solutions. Decomposition is not specific to the system level and can be applied at higher or lower abstraction levels.

- “**Hardware-Software Integration and Testing**” ensure correct interaction of software on target hardware, including interfaces, timing, and safety mechanism behavior.

- “**System Integration and Testing**” ensure that system elements (i.e., components or subsystems) are integrated according to the architectural design and safety concepts, and operate correctly without unintended behaviors violating safety goals.

- “**Vehicle Integration and Testing**” started with hardware-software integration, followed by system integration to realize the complete item, and finally vehicle-level integration. Each integration stage includes defined verification activities to demonstrate correct interaction and compliance with safety requirements.

1.2.6 Software - Modular Approach

“**Software**” level starts with design and specification, and depending on complexity, it might contain multiple abstraction levels. ADS software consists of both rule-based and ML-based software units. The rule-based software is implemented based on specified behavioral requirements and tested against them. In ML-based software, the model learns the behavior from the training data set by optimizing the parameters of the selected model, rather than being fully defined by explicit rules.

- “**Software Safety Requirements**” are derived from the system level technical safety requirements. Relevant software safety requirements are specified as a result of the safety analysis.

- “**Software Architecture**” is designed to satisfy all software requirements, including specified safety solutions with their required ASIL. The software architecture describes both static (e.g., interfaces) and dynamic (e.g., timing behavior) aspects. At this stage and before implementation, relevant analysis techniques such as dependent failure analysis, FTA, and FMEA are performed to identify whether any safety requirements are violated.

- “**Software Unit**” is the atomic-level element in the software architecture

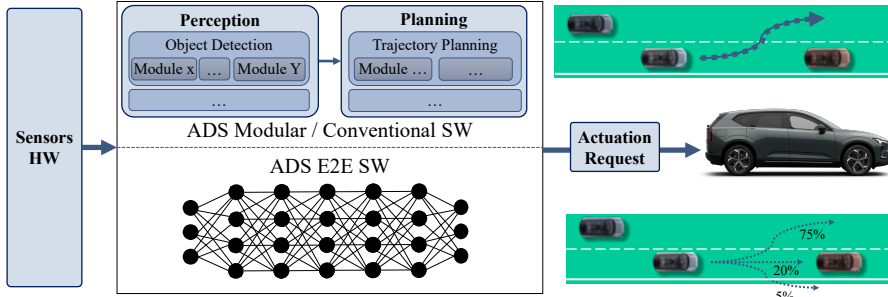


Figure 1.9: Illustration of two main paradigms in ADS software development: In the conventional approach, the software architecture consists of multiple abstraction levels, starting with perception and planning, each of which consists of multiple software modules (e.g., components or units). The behavior of each module is described by requirements, which are essential for the implementation, verification, and validation of that module. However, in the end-to-end (E2E) development, a model is responsible for both perception and planning, and the behavior of the model is learned from training data.

that is designed and implemented at this step. Each software unit can be either rule-based or ML-based.

- **“Software Unit Verification”** is performed to provide evidence that a software unit satisfies the allocated safety requirements. To evaluate the completeness of verification, relevant structural coverage metrics shall be selected, which include statement coverage, branch coverage, and Modified Condition/Decision Coverage for rule-based software.

- **“Software Integration and Testing”** ensures that the integrated software units and software components are implemented and verified according to the software architectural design and meet safety, functional, and quality requirements.

1.2.7 Software - “End-to-End” Approach

Conventional ADS software typically employs ML-based components within a modular architecture, where ML is used for realizing specific capabilities such as object detection. ML is especially crucial for perception-related tasks, as these tasks face unbounded environmental complexity that is difficult to capture with explicit rules. With advances in hardware capabilities and training infrastructure, it has become feasible to gradually extend model boundaries into the “Planning” module. This then leads to the system directly consuming sensor outputs and outputting actuation requests to vehicle actuators, a paradigm known as end-to-end (E2E). Fig. 1.9 illustrates modular vs. E2E paradigms.

In a modular approach, the rules are hand-crafted and require manual updates whenever new requirements emerge or weaknesses are identified. In contrast, E2E approaches rely on data-driven logic that can be updated automatically as new datasets are collected and integrated through MLOps loops.

This automation improves responsiveness to long-tail events and enhances software scalability, since system capabilities evolve primarily through data rather than manual rule engineering. Maintenance is also improved, as faster iteration becomes possible once data and training pipelines are established.

Unlike the modular approach, where selected architectural patterns can be analyzed using techniques such as software FMEA, the error propagation and causal relationships between different model elements are not explicitly traceable in E2E. Similarly, in testing, it is more challenging to identify the cause of failures. Attempts such as [26] and [27] address causality in E2E autonomous driving at the level of learned model behavior rather than at the architectural design level. These approaches inherently require a fully implemented and executable E2E architecture and are not applicable at the design phase, unlike methods such as FMEA for modular approaches.

Vision-Language-Action (VLA) [28], [29], [30] models are vision-only E2E systems combined with Large Language Models. VLAs are developed as a potential approach to enable richer scene understanding and explainability. In VLAs, the models can also generate natural language (NL) descriptions of the intended action, often described as reasoning, in addition to generating actuation requests. These textual outputs are often proposed as a means to improve interpretability. However, current research does not provide strong evidence that the generated textual information reflects the actual control logic of the system or how accurate they are. According to [27], these explanations can be entirely disconnected from the actual decision-making process due to model hallucinations.

1.3 DevSafeOps

1.3.1 Software Update and DevOps

Similar to other domains, the role of software in automotive systems is growing, and a significant part of the consumer experience is driven by the vehicle's software, known as software-defined vehicle [31]. Centralized computing units, over-specified hardware components (e.g., additional sensors for future functions), and over-the-air (OTA) updates are enablers that facilitate software updates.

Consumers' expectations, legislation, and technological aspects are evolving continuously, which require continuous function improvements to stay competitive. Moreover, field monitoring and fast reactions to unknown, rare, but still possible incidents are crucial to mitigate and to avoid them before they lead to loss events. Iterative and continuous software development, deployment, and monitoring are widely used for non-safety-critical software applications to enable rapid and innovative feature improvements and bug fixes. This approach, known as DevOps, couples continuous Development (Dev) and Operations (Ops) [31].

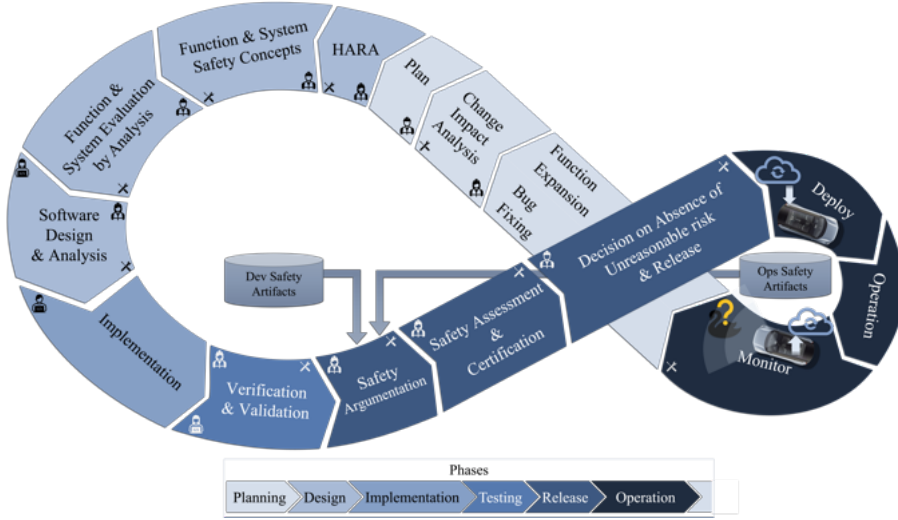


Figure 1.10: Presenting DevSafeOps cycles, which include the required safety activities (shown in Fig. 1.7) integrated into DevOps cycles. The significant number of activities required on the development side, compared to the operation phase, is illustrated by the asymmetrical shape of the loop.

1.3.2 Safety of DevOps

Continuous and iterative software updates and system modifications are not a new challenge for automotive systems. Change management [12] (ISO 26262, Part 8, clause 8) is a systematic process for maintaining the safety of the item while implementing changes to the item or its elements during an item’s lifecycle. For instance, as part of the safety planning at the start of each loop, performing a “Change Impact Analysis” is required to identify the affected work products and elements in the item, as well as the potential impact on them. Fig. 1.10 presents the integration of safety activities in DevOps’ infinity loop, which, unlike non-safety-critical applications, does not rely solely on final stage testing in the field. Instead, the process heavily requires “Safe-by-Design” principles.

Munk and Schweizer [31] proposed the SafeOps approach, while Siddique [32] introduced the SafetyOps framework, which first focuses on ensuring safety in iterative development and second on integrating safety and DevOps by emphasizing the need for adaptation to unique needs of safety. We propose DevSafeOps as a term to emphasize Safe-by-Design in the development phase, which the earlier terms miss.

1.3.3 MLOps

As described in Sec. 1.2.7, ML is highly dependent on continuous data collection and iterative model updates, which is referred to as MLOps (Machine Learning Operations). These frequent updates improve the performance of the model

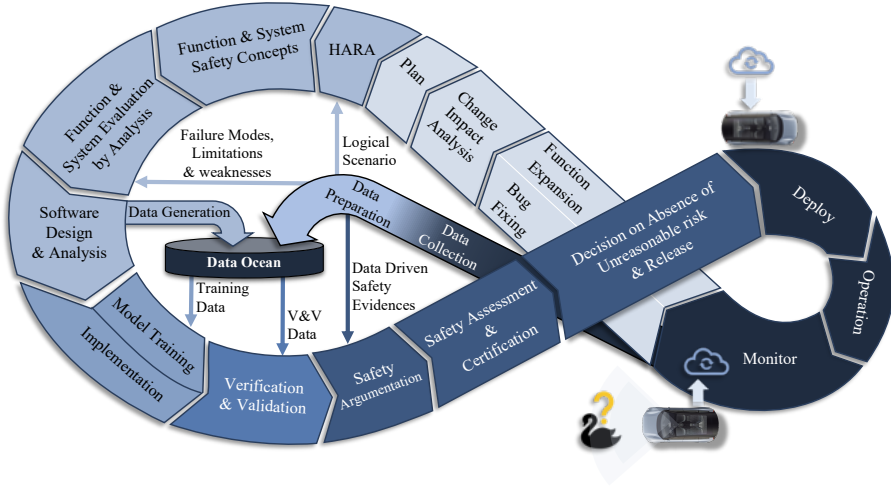


Figure 1.11: Presenting the MLOps lifecycle activities within DevSafeOps.

and mitigate distributional shift (e.g., new types of road user) and Out-of-Distribution ⁴ issues. As shown in Fig. 1.11, MLOps can be integrated into DevSafeOps loops. The data are continuously collected and used to update scenario failure mode catalogs, and are also used for training, verification, and validation of the model. These data can also be used in data-driven safety argumentation. The collected data can also be altered through reconstruction techniques for enriching and improving the diversity of the datasets, which is further discussed in Sec. 1.4.4

1.4 Methods and Tools

As described in Sections 1.2 and 1.3 a wide range of methods and tools may be employed across the different stages of ADS development, including safety analysis techniques, testing environments, and enabling technologies for design, implementation, and data handling. This section introduces the methods and tools that are relevant for this thesis. In particular, STPA is considered as a safety analysis method (Sec. 1.4.1) and simulation environments are used for testing (Sec. 1.4.2). Generative AI (LLMs/VLMs) is explored as a potential enabler for design and implementation activities (Sec. 1.4.3) and 3D Gaussian Splatting (3DGS) is discussed as a synthetic data reconstruction technique supporting a data-centric development and MLOps (Sec. 1.4.4).

1.4.1 System Theoretic Process Analysis (STPA)

Verification by analysis can be employed to identify functional inefficiencies, technical limitations ¹³ and design mistakes ¹² that can lead to safety-critical

⁴OOD: The training data do not include all relevant aspects or features of the environment the system is operating in ¹⁶.

hazardous events during an early development phase. Analysis is one of the key activities in the safe-by-design approach. STPA provides a systematic approach to identify and analyze hazardous events and corresponding causes in complex systems such as ADS. STPA is one of the methods referenced for AD development by regulators such as UN Regulation No. 157 [11]. The following summary of each step in STPA is gathered by aggregating and adapting the proposed steps from the STPA Handbook [33], SAE J3187 [25], Annex B.4 in ISO 21448 [13], and Appendix B in “Safety of the Intended Functionality of Lane Centering and Lane Changing Maneuvers of a Generic Level 3 Highway Chauffeur System” by NHTSA [34].

Step 1 - Define the Purpose of Analyses: The goal of this step is to define the function and then to identify hazards and relevant safety constraints to avoid them. This step consists of several sub-steps:

- **System Scope:** Here, system boundaries are defined and shall include information such as purpose, constraints (e.g., ODD), and actors [25]. This step is similar to “item definition” in the context of ISO 26262.
- **Losses:** The losses are the overall goal of the system to fulfill or topics of concern that affect the stakeholders of the system such as “financial loss” and “loss of life or injury”. Based on the instructions and examples in the STPA Handbook [33], they should be generic for each context (e.g., financial, safety, privacy) and they are not allowed to be more granular. For example, regarding safety-related aspects, the only loss is “loss of life or injury”. Unlike ISO 26262, which assesses the hazard based on four levels of severity from no injuries (S0) to fatality level (S3) [12], STPA only *identifies* the hazard and does not intend to *assess* it.

Most of the time, such losses are connected such as “loss of life or injury”, which leads to “financial loss”. But in some cases, satisfying one can also violate another one. This is why there is a need to prioritize the losses. Then, the priority would be inherited by traceability between the results of each step and the relevant loss [25].

- **Hazards:** A hazard is caused by malfunctioning behavior [12] (Functional Safety causes) or insufficiencies [13] (SOTIF causes) of the function, which in a specific scenario would lead to a loss.
- **System Level Constraints:** Based on the example mentioned in SOTIF [13], system level constraints are in the same abstraction level as safety goals in the context of FuSa or same as validation targets in SOTIF. Since in automotive abstraction levels, a vehicle function consists of several systems and thus, “vehicle-level safety constraints” is a more suitable name for this sub-step as also proposed in SOTIF [13].

Step 2 - Model Control Structure: A control structure is a model representing the interaction between elements (i.e., control action and feedback) and the hierarchy of control [33] at each abstraction level. The relevant properties of elements and their interactions need to be chosen for each designated abstraction level.

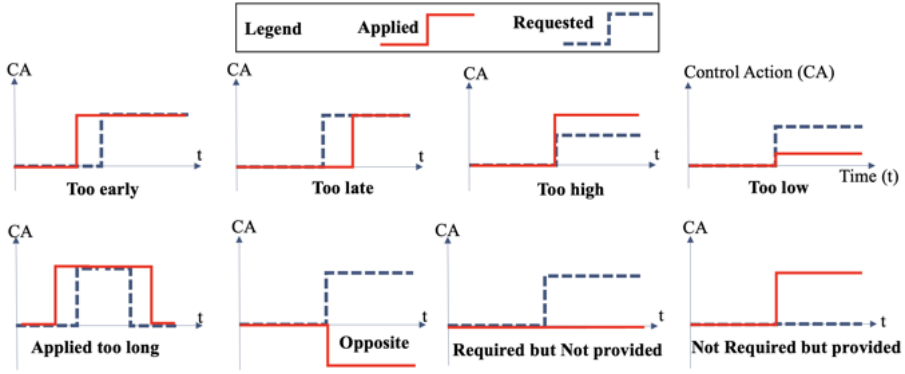


Figure 1.12: Illustrating the suitable undesired or unsafe manners, which lead to UCAs in AD, extracted from the STPA Handbook [33].

Although the main part of the analysis will be done in the next steps, drawing the control structure itself may highlight design weaknesses, as raised by the STPA Handbook [33]. Missing a block, control action, feedback, or having an incorrect hierarchy of control in the control structures, such as the one presented by Xing et al. [35], can result in an incomplete analysis in steps 3 and 4, potentially leading to missing or incorrectly defined safety requirements. The control structure in [35] is presented as a physical model rather than as a functional model, which can lead to the omission of functional elements mapping to the main blocks in a control structure (e.g., controlled process). Additionally, the hierarchy of elements does not follow the order proposed by the STPA Handbook [33]. For instance, the driver, as the operator, should be positioned at the top of the control structure, but in this case [35], it is placed at the bottom.

Step 3 - Identify Unsafe Control Actions: Based on the STPA Handbook, an Unsafe Control Action (UCA) is a state or a behavior of control action that, in a specific scenario, would lead to a hazard [33]. In other words, each UCA leads to a violation of the vehicle-level safety constraint. The UCAs shall be traceable to at least one hazard; otherwise, it is an indication for a missing hazard and the relevant hazard shall be introduced.

Using a set of guidewords combined with control actions can help to find UCAs. The STPA Handbook proposes “not providing,” “providing,” “too early or late,” “out of order,” “stopped too soon,” and “applied too long.” Since these guidewords will be used in combination with control actions, they shall be adapted to the control action and the context. In Fig. 1.12, a set of suitable guidelines for AD are shown with respect to the required and safe control action. The list is not limited to these and based on the nature of the Control Action (CA), more guidewords can be used, or some might not be applicable. In addition, enhancing the pre-existing knowledge about the undesired or unsafe control actions from other projects might be helpful.

Safety Requirements: In ISO 21448 [13] and the STPA Handbook [33], the next sub-step is to specify safety constraints to prevent UCAs. SAE J3187 [25]

introduced this sub-step only in step 4, which leads to an incomplete analysis in step 3, as there would not be any safety requirement to avoid UCAs.

Step 4 - Identify Loss Scenario: Identifying the causal factors is the last step, after which relevant safety constraints shall be specified. There are four main types of causes that shall be analyzed in this step [33]:

- Controller-related such as “inadequate control algorithm”
- Feedback or sensing-related such as “unsafe control inputs” or “incorrect feedback”
- Execution or actuation-related such as “the command is not received” or “improperly executed”
- Other factors related to controlled processes such as “the actuation is not effective”

The first two types in the list are the causes for unsafe control actions in step 3, and the last two are the causes for the improper execution of a correct control action.

Safety Requirements: Similar to step 3, the safety requirements are specified to mitigate or prevent the identified causal factors [25]. Safety requirements, which are defined in steps 3 and 4, are communicated to the next abstraction level to be used as a starting point of the analysis.

1.4.2 Simulation Environment

As illustrated in Fig. 1.13, each closed-loop testing environment targets a specific aspect of the ADS. Real-world testing through Vehicle-in-the-Loop (ViL) has the highest fidelity, with the customer using the function during the operation phase. However, this environment has limitations, such as being slow, costly, and not scalable, so it is used as the final step in verification and validation. Hence, multiple environments with different level of fidelity and scope are designed and developed, each of which aims to verify a specific aspect of the ADS.

In each of these environments, the rest of the system, which are out of the scope of testing, is assumed ideal. In Hardware-in-the-Loop (HiL), the environment is assumed ideal and the focus is on system or hardware–software integration testing, while in Software-in-the-Loop (SiL), hardware is assumed ideal and the focus is on the relevant software chain. For instance, “Car Learning to Act” (CARLA) [37] provides camera, lidar, and radar information through ideal sensor models, which makes it suitable to test perception and planning of the ADS software, while Environment Simulator Minimalistic (esmini) [36] aims only at planning and considers ideal perception.

1.4.3 Generative Artificial Intelligence (Gen-AI)

Generative Artificial Intelligence (Gen-AI) refers to a class of models capable of generating content, such as text, code, or images, based on learned patterns from

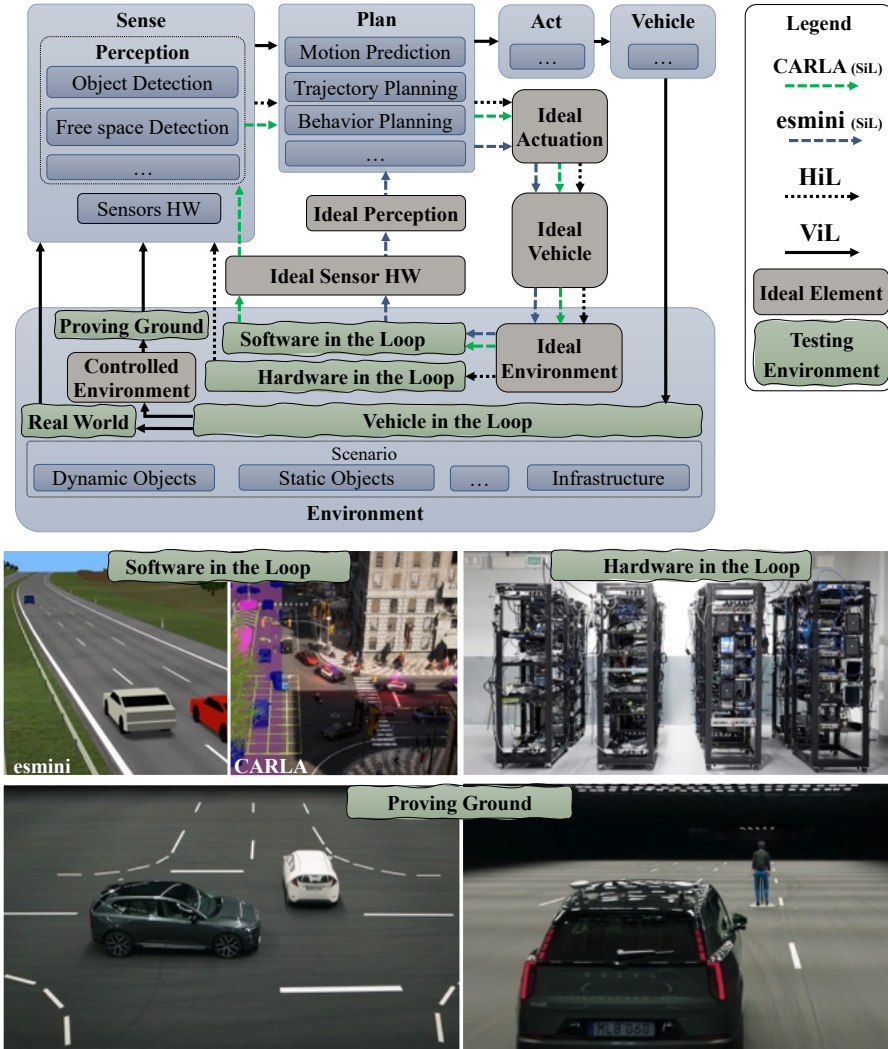


Figure 1.13: Presenting different closed-loop testing environments and elements under test. As each closed-loop environment is designed to test a specific aspect of the ADS, the remaining elements are considered ideal. The arrows represent the testing loop of each environment, the tested subjects, and the assumed ideal elements. Software-in-the-Loop (SiL) environments, such as CARLA and esmini, test a specific part of the software, while the hardware is considered ideal. Hardware-in-the-Loop (HiL) testing evaluates the integration of software and hardware. Finally, the software is integrated into the vehicle for the final stage of testing in a proving ground, which is a controlled environment, and, if satisfactory, the ADS is tested in the real world. Figure created by the author, incorporating images from [1], [36], [37]

large amounts of data. LLMs are Gen-AI-based tools capable of text processing and generation. LLMs are trained on vast amounts of text across various domains, and their outputs are perceived as relevant for diverse tasks [38]. As the model generates text based on the input, crafting the input, known as prompt engineering, is crucial for obtaining relevant output. Some of these prompt patterns are suggested by studies such as [39]. LLMs have also demonstrated the ability to generate and modify source code, which has led to their widespread adoption in software development tools such as Claude Code and GitHub Copilot.

Vision Language Models are another category of Gen-AI-based tools that combine visual perception modules with language understanding and generation capabilities. They can interpret images and a given prompt to extract information from an image.

Despite their capabilities, Gen-AI models exhibit several known weaknesses and limitations that require identification and consideration when they are employed. Due to the stochastic nature of these models, multiple weaknesses are reported, such as generating text that is not correct with respect to reality, commonly referred to as hallucination [40].

1.4.4 Synthetic Data Generation

Data plays a crucial role in model training and testing within MLOps loops. However, collecting data from the real world is often slow and, for some scenarios such as safety-related events, hard to obtain. Hence, synthetic data generation can be seen as a potential solution to improve data availability and coverage.

One approach is to use recordings from high-fidelity simulation environments. However, such data is typically less realistic than real-world data, and all assets in the environment must be manually developed and maintained for each new object in the scenario. This limits scalability and realism. An alternative approach is to use scene editing and reconstruction-based techniques using real-world recordings, such as Neural Radiance Fields (NeRF) and 3D Gaussian Splatting (3DGS).

In 3DGS, multi-view sensor data recorded from the real world is used to reconstruct the scene as a set of 3D Gaussian primitives that jointly encode geometry and appearance. This representation enables efficient rendering while maintaining high visual fidelity. Importantly, the reconstructed scene can be modified or edited, allowing controlled changes to the environment and the generation of new synthetic scenarios based on real-world data.

1.5 Motivation and Problem Domain

Integrating automotive systems engineering, with all its complexities and numerous activities as explained in Sec. 1.2, with the rapid and frequent iterative DevOps process (Sec. 1.3) is challenging. In other applications, the risk of prioritizing speed over quality might be seen as an option; however, this is not acceptable in safety-critical applications, and an OEM will not release

the product without performing all required activities to ensure the product is safe. Hence, the automotive industry is lacking comprehensive methodological and technical approaches to enhance the speed and efficiency of DevSafeOps iterative loops.

1.6 Potential Approaches

Distributed development and automation using software tools are already part of the process in automotive systems engineering, which also improves development speed. Distributed development (Sec. 1.6.1), as one approach, represents a category of development models that enhance the efficiency and effectiveness of teams of engineers during system design. “Gen-AI for Systems Engineering” (Sec. 1.6.2), on the other hand, represents technologies that can enable automation of activities involved in the automotive process.

In addition to natural language generation capabilities of Gen-AI, their code generation capability also improves speed and efficiency in software update cycles. However, for ML-based software, which increasingly replaces rule-based software, data are required, for which 3DGS (Sec. 1.6.3) can be seen as a potential approach to populate the data collected from the real world.

1.6.1 Distributed Development

Distributed development is essential due to the complexity of such systems and the short time to market. It also enables the OEM to leverage the expertise from specialized suppliers in each technology, ensuring the system is developed to the highest performance and quality levels. Thus, modularization needs to be employed to clearly define the perimeter of the responsibility of each internal or external team involved in development. Moreover, each team is responsible only for their own module, which also protects the intellectual property and business factors of each party.

1.6.2 Gen-AI for Systems Engineering Automation

Most existing automation pipelines, such as CI/CD pipelines, are applied to repetitive tasks by leveraging predefined processes that do not require creativity, while most engineering tasks are intellectual activities. As Gen-AI technologies such as LLMs and VLMs have demonstrated capabilities in generating natural-language and code-based content, they can be considered promising enablers for automation of engineering tasks. The following are some use cases that can be enabled by Gen-AI in DevSafeOps cycles:

Safety Analysis: Analysis methods, such as HARA, are natural-language-based activities which consist of multiple steps, each of which is well described and can be used in an LLM-based system to be automated.

Requirements Engineering: Requirement specification is the next step, which, through defining strategies, can benefit from LLMs and generate multiple proposed strategies to mitigate the failure modes detected in the safety analysis step.

Code Generation: The LLM capability in code generation, if combined with SiL, can be pre-validated and then delivered to engineers, which improves efficiency.

Code as X: The code generation capability can also be used as a formal notation for specifying safety solutions instead of requirements being expressed in natural language, which then be automatically verified in a SiL environment. Using code as requirements tackles the ambiguity in natural language. This can be extended to concept descriptions as well. Moreover, they can also be used as a representation for the architecture.

Monitoring: These models are capable of accepting unstructured textual data (e.g., incident reports) and visual (e.g., images from real-world scenarios) information, which makes them also suitable for the monitoring phase during operation.

1.6.3 3DGS

Data plays the same role for ML-based software as requirements for rule-based software [41]. 3DGS, as a scene editing and reconstruction technique, can be employed to populate the datasets. However, the fidelity of such technology needs to be further investigated, as the data are used in the development of safety-related software.

1.7 Research Goal and Questions

The goal of this study is accelerating safety aspects of ADS in DevOps. This research goal is multidisciplinary, encompassing software engineering and automotive systems safety engineering. Moreover, the complexity of ADS and its development process justify breaking down the research goal into more granular objectives to tackle each challenge individually. At the time of this research, there was no systematic study on the existing challenges and corresponding solutions, making it crucial to first identify the gaps and then propose approaches to tackle them (see Paper B for a detailed discussion ??). The following list presents the two research goals and their corresponding research questions:

G1: Identify challenges, existing solutions, and gaps.

RQ1: What are the challenges to rapid DevSafeOps of ADS?

RQ2: What solutions are proposed in the literature to address the challenges, and what are the gaps?

G2: Identify the root causes for each challenge and addressing them by proposing relevant technical and methodological approaches.

RQ3: What are the root causes of each challenge?

RQ4: What technical or methodological approaches can be proposed to tackle the challenges in RQ3?

The research questions in this thesis are addressed in the appended papers. Tables 1.1 and 1.2 present the traceability between the research questions in this thesis and the ones in the appended papers.

1.8 Scope

This thesis centers on the safe design and development process of an ADS, specifically identifying the challenges of DevSafeOps and collecting evidence to develop a proposed approach to govern them. The study starts with safety analysis methods and requirements engineering as the foundation for a safe-by-design approach. Then, it continues with the implementation phase and finally finishes with monitoring. The study focuses on software development, while hardware aspects of ADS are out of scope.

The technical aspects (e.g., sensor technologies), social or legal requirements (e.g., defining safety), and system design are out of scope of this thesis. However, for illustration purposes, some of these aspects are used to clarify the proposed approaches or serve as examples for applying the methods or using the tools.

The study uses LLMs and VLMs as a tool for safety analysis, requirement engineering, code generation, and monitoring tasks without delving into their underlying development processes, as the internal development and technical aspects of LLMs are out of scope. However, relevant aspects such as LLMs' weaknesses, concerns, and limitations are assessed to understand their effects on the output and to design appropriate mitigation mechanisms. Legal and ethical aspects of LLMs are also out of scope of this study, although identified concerns are reported.

1.9 Structure of the Thesis

In Chapter 2, we describe our research design, including the research paradigms and methods employed in this study. In Chapter 3, we summarize the key findings of the included papers and highlight our contributions in the context of the respective research area. Chapter 4 presents a discussion of the research contributions and potential threats to validity. Finally, we conclude our findings and suggest directions for future research in Chapter 5.

The kappa is intended to introduce the PhD research and highlight the key contributions of each paper, offering a snapshot of their significance without duplicating the in-depth findings, analyses, and discussions present in the papers. For further details, including comprehensive evidence and findings, readers are encouraged to refer to the individual papers.

Table 1.1: Presenting the mapping between thesis research questions and appended papers’ research questions (part 1). An ID similar to RQX.Y.Z is allocated to each research question, where X denotes the main research question ID as defined in Sec. 1.7, Y refers to the appended paper ID, and Z represents the ID of the research question in that paper.

RQs	Contributing research questions in appended papers
RQ1	RQ1.A.1: What are safety-related challenges during field monitoring of AD?
	RQ1.A.2: What are safety-related challenges for continuous development and deployment?
	RQ1.B.1: What challenges are identified in literature when applying DevOps to safety-critical AD functions?
RQ2	RQ2.B.2: What solutions are proposed in literature for these challenges while fostering rapidity and safety?
	RQ2.B.3: What challenges for DevOps in safety-critical AD applications still remain open in literature?
RQ3	RQ3.C.1: What are the challenges of applying STPA in an automotive context with multi-abstraction levels in distributed development for OEM and suppliers?
	RQ3.C.2: What do existing guidelines and papers propose for handling the multi-abstraction level designs in a distributed development environment for each involved party?
	RQ3.E.1: What are the limitations of using LLMs for specifying safety requirements for AD functions?
	RQ3.F.1: How do state-of-the-art LLM models perform in code generation tasks for automotive functions?
	RQ3.F.2: What are the key limitations and risks of using LLM-generated code in safety-related applications?
	RQ3.G.1: What are limitations of an LLM-based automatic code generation for safety-related, automotive systems?
	RQ3.H.1: What activities in the DevSafeOps lifecycle can be accelerated or improved using LLM-based multi-agent pipelines?
	RQ3.H.2: What are the capabilities and limitations of LLMs when employed to support activities within the DevSafeOps?
	RQ3.I.1: How reliable is 3DGS for reconstructing novel-view traffic scenarios from real-world sensor data in AD?

Table 1.2: Presenting the mapping between thesis research questions and appended papers’ research questions (part 2). An ID similar to RQX.Y.Z is allocated to each research question, where X denotes the sub-goal ID, Y refers to the paper’s name, and Z represents the ID of the research question in that paper.

RQs	Contributing research questions in appended papers
RQ4	RQ4.C.3: How could STPA be modified to overcome the identified and confirmed shortcomings in RQ3.C.1 for each involved party? RQ4.D.1: What are the steps in a prompting pipeline to propagate the context needed to generate a HARA? RQ4.E.2: What is the task breakdown to enhance the LLMs’ performance in specifying safety requirements using HARA? RQ4.E.3: How can prompt engineering enhance the LLMs’ performance in specifying safety requirements for AD functions? RQ4.F.3: What adaptations to existing software engineering processes are necessary to safely integrate LLM-generated code into automotive software development workflows? RQ4.G.2: What are potential automatic approaches to tackle those limitations and improve the quality of the code? RQ4.G.3: How do state-of-the-art LLM models perform under the proposed approach, when evaluated against quantitative key safety-related performance indicators? RQ4.H.3: What are the potential approaches to address the limitations and risks of LLM-based multi-agent pipelines within the DevSafeOps? RQ4.H.4: What adaptations to the DevSafeOps loop are required when employing LLM-enabled tools? RQ4.I.2: How can the fidelity of reconstructed novel-view traffic scenarios from real-world sensor data be dynamically evaluated and assessed for AD?

Chapter 2

Research Design

The methodology defines the process of conducting research, starting from problem investigation, data collection, analysis, and interpretation. Establishing the appropriate research methodology and tailoring it to suit the research question forms the backbone of any research, aiming reliability, validity, and generalizability of the results.

2.1 Research Paradigms

Knowledge or Design Questions:

One way to classify research questions is into “Knowledge questions” and “Design questions”. Knowledge questions investigate the current state of software engineering in a particular sub-area, while the latter focuses on designing better methods, tools, or processes [42].

Academia (contrived) vs. Industry (natural) Setting:

A natural setting studies a phenomenon without altering the real-world environment, whereas in contrived settings, researchers study the subject within an artificial environment (e.g., simulation) [43]. Studies in natural settings contribute to relevance, realism, and richer qualitative data but might lack precision, replicability, and isolation of variables since the environment is not designed and controlled by the researcher.

The multi-disciplinary nature of software engineering does not solely depend on technological aspects but also includes numerous sociological aspects. Understanding the development and maintenance of such complex and evolving systems requires more than just examining processes or tools; it also involves considering the social and cognitive aspects of the process [42]. To properly understand the complexity of automotive software engineering, the study must be conducted in industrial settings [42]. Otherwise, the proposed approaches may not be suitable for real-world usage. Continuous involvement of industrial practitioners in such research is an essential factor, ensuring the relevance of

collected data, applicability of findings, and confirming the benefits for the industry.

Positivism, Constructivism, and Pragmatism:

Positivism, constructivism, and pragmatism are the philosophical stances related to this study. Positivism seeks a logical relationship between knowledge and facts, while constructivism focuses on the human context of knowledge. Meanwhile, pragmatism evaluates the value of knowledge based on its usefulness for solving real-world problems, introducing a subjective element into the evaluation of results [42].

Exploratory, Descriptive, Explanatory, and Improving:

Exploratory methods are employed to study a phenomenon in its natural context through flexible and observation-driven investigation. Descriptive methods aim to depict a situation or phenomenon and are suitable for answering research questions that start with 'What,' 'When,' or 'Where.' Then, explanatory methods are used to address research questions that start with 'How' or 'Why' and are aimed to discover causal relationships. When a specific aspect of the subject under study presents challenges, improving methods are employed to improve the relevant elements in that phenomenon [44].

Inductive vs. Deductive:

In the inductive approach, the researcher first gathers data and then constructs a theory based on that data. In the deductive approach, they start with a theory and then validate it using data [44].

Qualitative, Quantitative, or Mixed methods:

The collected data, consisting of words, diagrams, pictures, and descriptions, is categorized as qualitative data, while quantitative data consists of numbers [44]. Quantitative research typically employs statistical techniques on numerical data to examine the relationship between variables, while qualitative research explores human behavior and experience, usually using non-numerical data.

2.2 Prescribed Methods

The research questions are classified based on the categories previously explained and are illustrated in Fig. 2.1. Understanding these categories can better guide us in selecting methods and designing a study. The necessity for a natural setting and the categorization of the main study goal primarily as pragmatism in this study narrows down the methods to the following:

RQ1, RQ2 & RQ3: The exploratory and qualitative nature of these knowledge questions makes the following two methods suitable candidates:

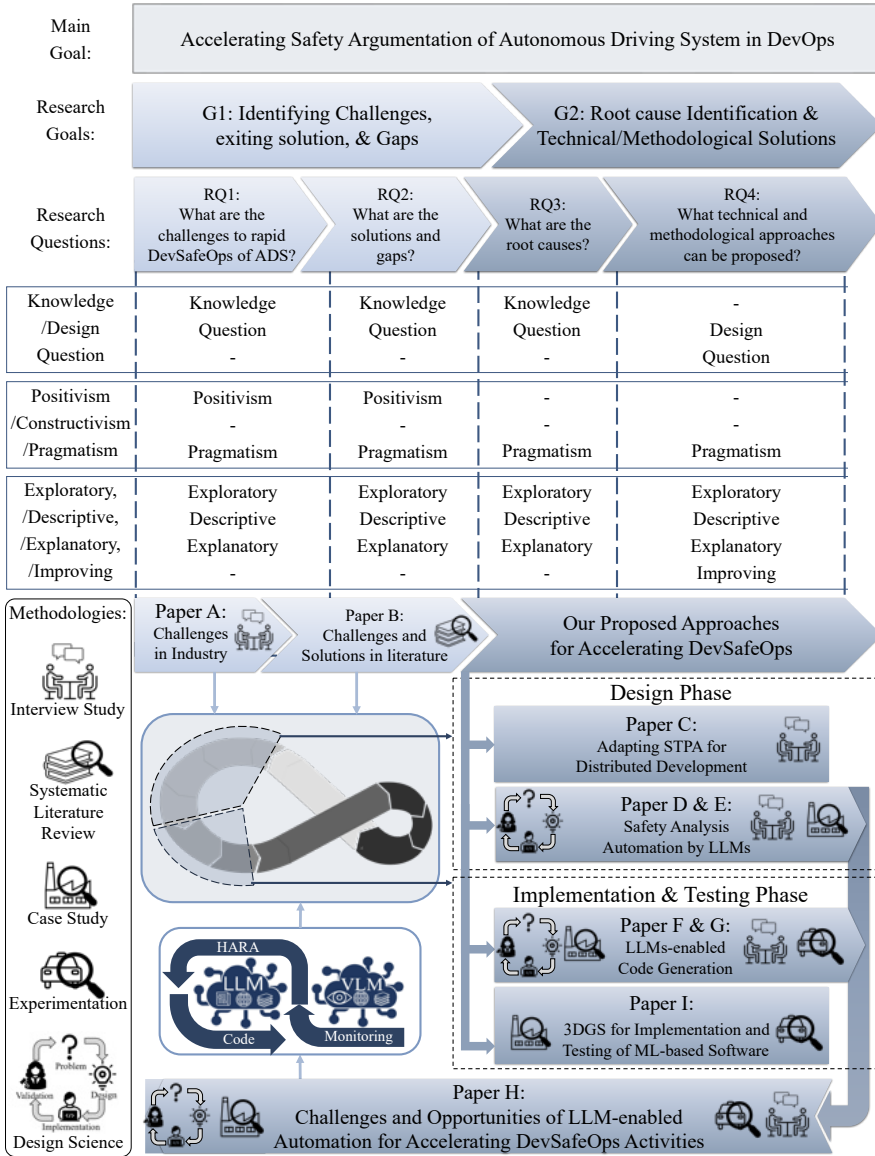


Figure 2.1: Presenting the traceability between research goals, research questions, and the studies performed in the appended papers. The top part presents the research goal and the breakdown into research questions. The middle part presents the categorization of the research questions. The lower part presents the scope and methods of the appended papers. Papers A and B contribute to identifying the challenges and existing solutions in the DevSafeOps. The Papers C to I propose approaches for the challenges in the design phase (C, D, and E), implementation & testing (F, G, and I). For the LLM-enabled approach in paper H, we zoomed out and look at the whole DevSafeOps.

- Expert Interviews:** To explore and discover the unknown limitations and challenges, experts were interviewed. (For RQ1.A.1, RQ1.A.2, RQ3.C.1, RQ3.F.2 and RQ3.H.1, and RQ3.H.2)
- Literature review:** To collect and synthesize the challenges identified by other researchers or industrial practitioners from literature, as well as suitable approaches to address them, multiple literature review were conducted. (For RQ2.B.2, RQ2.B.3, RQ3.C.2, and RQ3.E.1, RQ3.F.1, RQ3.F.2, RQ3.G.1, RQ3.H.1, RQ3.H.2, and RQ3.I.1)
- Case Study:** To investigate specific industrial needs, constraints, and contextual limitations, an industrial setting is considered. (For RQ3.E.1, RQ3.F.1, RQ3.F.2, RQ3.G.1, RQ3.H.1, RQ3.H.2, RQ3.I.1)
- Experimentation:** To explore the root causes both quantitatively and qualitatively, experiments were conducted when possible. (For RQ3.F.1, RQ3.F.2, RQ3.G.1, RQ3.H.1, RQ3.H.2, and RQ3.I.1)
- RQ4:** This research question is categorized as ‘improving’ and is connected to design questions. Given the nature of the topics and their qualitative aspects, the following methods have been selected as suitable:
- Design Science:** To facilitate the design of treatments for identified problems in a systematic and iterative manner. (For RQ4.C.3, RQ4.D.1, RQ4.E.2, RQ4.E.3, RQ4.F.3, RQ4.G.2, RQ4.G.3, RQ4.H.3, and RQ4.H.4)
- Expert Interviews:** To validate the effectiveness of the proposed approaches, experts were asked to review the design and provide comments. (For RQ4.C.3, RQ4.E.2, RQ4.E.3, RQ4.G.3, RQ4.H.3, and RQ4.H.4)
- Case Study:** To validate the effectiveness of the proposed approach in an industrial setting, the method can be applied in an actual industrial environment. (For RQ4.D.1, RQ4.E.2, RQ4.E.3, RQ4.F.3, RQ4.G.2, RQ4.G.3, RQ4.H.3, RQ4.H.4, and RQ4.I.2)
- Experimentation:** To quantitatively evaluate the performance of the proposed approaches and identify their capabilities and limitations under controlled conditions. (For RQ4.F.3, RQ4.G.2, RQ4.G.3, RQ4.H.3, RQ4.H.4, and RQ4.I.2)

2.2.1 Systematic Literature Review

A systematic literature review (SLR) is a method for identifying, evaluating, and synthesizing the scientific literature relevant to a research question. Kitchenham et al. [45] provide recommendations for designing the SLR, covering the definition of the review protocol, search strategy, inclusion, and exclusion criteria.

In Paper B, an SLR is employed to identify the state of the art for our main research goal and to capture existing gaps. Its primary goal is to identify the

challenges highlighted within these papers and to map them to existing solutions proposed in the literature. Paper B serves as a foundation for other parts of our research, where we focus on addressing each identified challenge. A literature review is used as a primary step in other papers as well, although some aspects, such as the documentation of the protocol and review strategy, are simplified. For example, in Papers C to I, queries are employed to capture relevant literature, aiming to identify the state of the art as well as the limitations of the subject under study.

2.2.2 Interview Study

In an industrial setting, especially for novel products and their development aspects, much of the knowledge resides in the experience of experts, as the documentation may not be mature. Therefore, there is a need for a systematic research method to extract this information. During interviews, the researcher engages with the participants and asks questions to collect data on the subject under study. These interviews can be conducted for primary data collection or for the validation of data from other sources [44]. During the design of the interview study, the researcher must make various design choices, including, but not limited to:

- a. Type of questions (e.g., open- or closed-ended)
- b. Length of the interview, and number of sections
- c. Expert's profile (e.g., experts with more than X years of experience in ...)
- d. Method of participant selection (e.g., convenience sampling)
- e. Data collection tools and environment (e.g., physical vs. online)
- f. Number of participants and saturation point

When the interview is designed, a pilot interview can be conducted to assess the effectiveness of the interview design, as well as the relevance, clarity, and structure of the questions. It also helps the researcher to practice managing the session and improve the flow of the interview for the main interview sections. Expert interviews are used in Paper A for exploring, and in Papers C and E for both exploring and validating the designed improvements.

In Paper A, the purpose is to collect insights from industrial practitioners working in various aspects of autonomous vehicle development. These insights are then analyzed and clustered into relevant challenges in DevSafeOps cycles. The study employs open-ended questions to first inquire about the challenges they are facing and, secondly, to ascertain if they agree with the challenges identified by the researchers.

In Paper C, the limitations of STPA are identified, and the method is adapted to better suit automotive systems engineering. Expert interviews are used to validate the limitations experienced by the researcher, as well as the effectiveness of the proposed solutions. The study is designed to maintain exploratory aspects, allowing experts the freedom to express their opinions freely and provide answers beyond given options or select hybrid solutions (i.e., open-ended questions).

In Paper E, the effectiveness of the prototype is evaluated by reviewing the outputs, such as the HARA table and safety requirements. These results are

forwarded to experts for a detailed analysis and review. After receiving their comments, interviews are conducted to first clarify these remarks and then to ask predefined questions aimed at assessing the quality of the outputs. Finally, the experts are informed about how the outputs were generated, allowing the collection of their suggestions to improve the prototype, thereby adding an exploratory aspect to the study. In Papers G and H, interviews are performed as an exploratory method to capture potential use cases, limitations, and concerns regarding the proposed approaches.

2.2.3 Case Studies

Another method for studying a phenomenon in its natural setting, such as an industrial environment in this research, is the case study. This approach is particularly valuable when the distinction between the phenomenon and its environment is not clear, leading to a lack of control [44]. The case definition, data collection/analysis methods, data selection strategy, and validity assurance are some of the key elements in designing a case study.

In Paper E, a prototype is designed and implemented for performing HARA using a pipeline of LLMs. The prototype is implemented within an IP-protected service, allowing for fine-tuning of the process for the case company and utilizing real input in an industrial setting. The vehicle function used as input is a novel feature, not previously accessible for training the LLM. Initially, the prototype is presented to the team responsible for HARA. Subsequently, four experts are asked to use the prototype and to provide feedback on their experiences. The case study serves as a form of triangulation alongside the interview study, as it involves a real function that already has an associated HARA within the company, enabling participants to compare the prototype's output quality against the existing company standard.

Similarly, the prototypes and frameworks in Papers F, G, H, and I are presented to relevant teams to explore their opportunities and limitations. Moreover, through continuous dialogue with practitioners during each design cycle, potential improvement strategies are explored, and the final proposals and their results are validated.

2.2.4 Experimentation

Experimentation is a fundamental research method in scientific and engineering fields [46]. Understanding a phenomenon involves modeling its elements and conducting controlled studies. In such studies, one factor (independent variable) is manipulated while other variables are kept constant in order to identify relationships between factors through measurable effects in a controlled environment. In Papers F to I, experimentation is employed as a primary validation method. The suitable environment for each study and the corresponding evaluation metrics are presented as follows.

Environments

As described in Sec. 1.4.2, simulation environments act as enablers for SiL, which is the focus of papers F to I.

In Papers F, G, and H, the conceptual code of AD functions requires only an abstract representation of behavior, and perception-related aspects are not needed. As presented in Fig. 1.13, esmini [36] focuses on behavior and trajectory planning, while the rest of the system is idealized or minimally modeled to enable closed-loop execution. At the same time, fast response time is essential, as the simulation environment serves as an integrated abstract world model to mitigate LLM hallucination. Hence, esmini is selected as a suitable environment, and it is developed and employed in relevant industrial contexts. It serves not only as an integrated component of the pipeline, but its outputs also provide experimental evidence. These outputs are used to evaluate LLM performance in Paper F and the performance of the implemented pipeline in Papers G and H.

In Paper I, the proposed framework requires a photorealistic environment to provide both the original viewpoint dataset for 3DGS training and the ground truth from novel viewpoints for evaluation purposes. Therefore, CARLA [37] is employed as a high-fidelity simulation environment that supports perception-related experiments by providing AD sensors and automatically generated ground-truth annotations.

In Paper H, the focus of one of the experiments (i.e., VLM-enabled HARA and ODD detection) is on the evaluation of perception, and since closed-loop behavior is not required, a single snapshot of each scenario is adequate. Instead, the experiment requires highly diverse (e.g., illumination and weather variations), photorealistic, and semantically rich scenario frames that challenge perception. While simulation platforms such as CARLA and reconstruction techniques such as 3DGS are powerful for sensor-level and dynamic validation, they lack the efficiency and flexibility needed to generate sufficiently diverse and complex photorealistic scenes for this purpose. Hence, SORA is employed to generate scenario snapshots.

Evaluation Metrics

The Structural Similarity Index Measure (SSIM) [47] and Peak Signal-to-Noise Ratio (PSNR) [47] are widely used in scene reconstruction studies to quantitatively evaluate rendering quality compared to the ground truth. In Paper I, these two metrics were employed as a first step in the evaluation of reconstructed scenes, for quantitative performance comparison of different models.

PSNR quantifies pixel-wise fidelity between a reconstructed image $\hat{I}$ and a ground-truth image I . It is derived from the Mean Squared Error (MSE) between reconstructed and ground-truth images, based on pixel-level differences, and computed as:

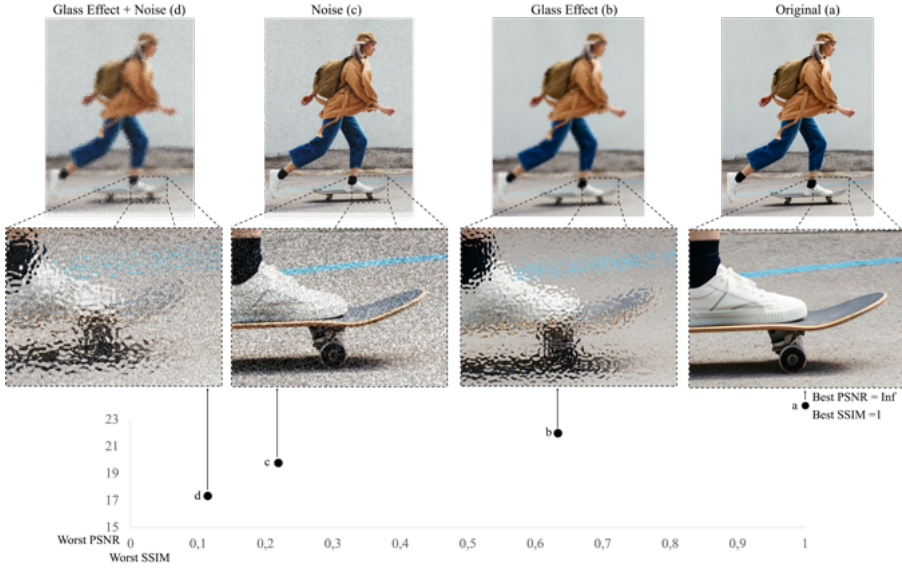


Figure 2.2: Illustrating visual and quantitative comparison of different controlled distortion types relative to the original image. The vertical axis represents Peak Signal-to-Noise Ratio (PSNR), while the horizontal axis represents the Structural Similarity Index Measure (SSIM). Original image (a) corresponding to a perfect reconstruction (PSNR = ∞ , SSIM = 1.00). Image (b) is subjected to a glass effect and image (c) is degraded with additive noise. Image (d) degraded with noise and then subjected to glass effect, representing the worst-case. The base image in “Original (a)” is sourced from Microsoft 365 Stock Images.

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N \left(I_i - \hat{I}_i \right)^2, \quad (2.1)$$

where N denote total number of pixels in the image. PSNR is then calculated as:

$$\text{PSNR} = 10 \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right) \quad (2.2)$$

where MAX is maximum possible pixel intensity value. As illustrated in Y axis of Fig. 2.2. Lower (worse) PSNR values indicate higher distortion in the reconstructed image with respect to the ground truth. However, PSNR does not account for structural information or local correlations within the image, as it depends solely on squared intensity differences.

SSIM, as a complementary metric, evaluates structural information rather than absolute pixel errors. SSIM is computed as

$$\text{SSIM}(I, \hat{I}) = \frac{(2\mu_I\mu_{\hat{I}} + C_1)(2\sigma_{I\hat{I}} + C_2)}{(\mu_I^2 + \mu_{\hat{I}}^2 + C_1)(\sigma_I^2 + \sigma_{\hat{I}}^2 + C_2)} \quad (2.3)$$

where μ_I and $\mu_{\hat{I}}$ are local mean intensities of the reconstructed image $\hat{I}$ and the ground-truth I . C_1, C_2 are small positive constants to avoid division by zero. σ_I^2 and $\sigma_{\hat{I}}^2$ denote the local variances, calculated as:

$$\sigma_I^2 = \frac{1}{N-1} \sum_{k=1}^N (I_k - \mu_I)^2 \quad (2.4)$$

$$\sigma_{\hat{I}}^2 = \frac{1}{N-1} \sum_{k=1}^N (\hat{I}_k - \mu_{\hat{I}})^2 \quad (2.5)$$

and $\sigma_{I\hat{I}}$ is the local covariance between I and $\hat{I}$, which can be estimated as:

$$\sigma_{I\hat{I}} = \frac{1}{N-1} \sum_{k=1}^N (I_k - \mu_I) (\hat{I}_k - \mu_{\hat{I}}) \quad (2.6)$$

Unlike PSNR, SSIM explicitly measures how well local structural relationships are preserved. This makes SSIM more sensitive to structural distortion. However, in Paper I, the reconstructed scenes are intended to be used for ML-based software development and testing. Hence, they are evaluated not only using image quality metrics (i.e., PSNR and SSIM) but also from a machine vision perspective. Therefore, the quality of the reconstructions is evaluated through object detection tasks using YOLO [48] and DETR [49] models.

Moreover, the semantic reconstruction quality is also evaluated through semantic segmentation metrics, including mean Intersection-over-Union [50] (mIoU) and the Dice [51] coefficient, computed from segmentation predictions generated by SegFormer [50]. Fig. 2.3 presents the value of one reconstructed data sample versus the original data point.

Object detection performance is quantified using True Positives (TP), False Positives (FP), and False Negatives (FN), through Precision and Recall, calculated as

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2.7)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (2.8)$$

IoU measures region overlap between predicted and ground-truth segmentation masks. For a given class k , IoU is defined as:

$$\text{IoU}_k = \frac{TP_k}{TP_k + FP_k + FN_k} \quad (2.9)$$

Then, mean IoU (mIoU) averages across all classes:

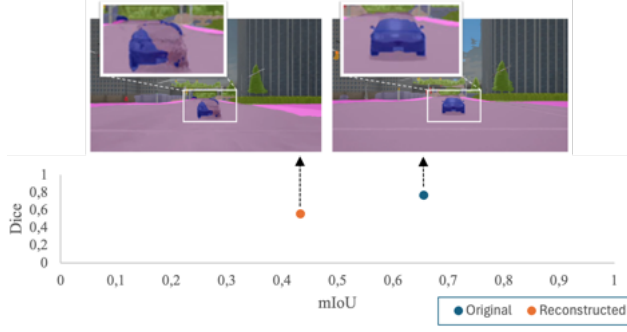


Figure 2.3: Illustrating the semantic segmentation comparison between reconstructed (left) and ground-truth (right) images. The upper row shows the predicted segmentation masks, and the lower plot presents mIoU (horizontal axis) and Dice (vertical axis).

$$\text{mIoU} = \frac{1}{K} \sum_{k=1}^K \text{IoU}_k \quad (2.10)$$

The Dice coefficient emphasizes on boundary agreement and is more sensitive to small object regions than IoU, which is defined as:

$$\text{Dice} = \frac{2TP}{2TP + FP + FN} \quad (2.11)$$

In Paper H, in addition to precision and recall, the F1-score and balanced accuracy are employed to evaluate and compare the performance of different VLMs for detecting ODD elements within the captured scene. The F1-score measures the balance between precision and recall. The F1-score is computed as:

$$\text{F1} = \frac{2TP}{2TP + FP + FN} \quad (2.12)$$

The F1-score penalizes extreme imbalance between precision and recall. For instance, a model with high precision but very low recall (or the opposite) will result in a low F1-score. However, precision, recall, and F1-score may still suffer when class distributions are imbalanced, which may pose a common threat to the validity of these metrics in ODD detection, where rare elements (the focus of study H) may appear infrequently. In such cases, the aforementioned metrics can be biased toward the majority class. To mitigate this issue, balanced accuracy is employed and computed as follows:

$$\text{Balanced Accuracy} = \frac{1}{2} \left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP} \right) \quad (2.13)$$

Balanced accuracy measures the average of sensitivity (true positive rate) and specificity (true negative rate), ensuring equal weighting of positive and negative classes regardless of class imbalance.

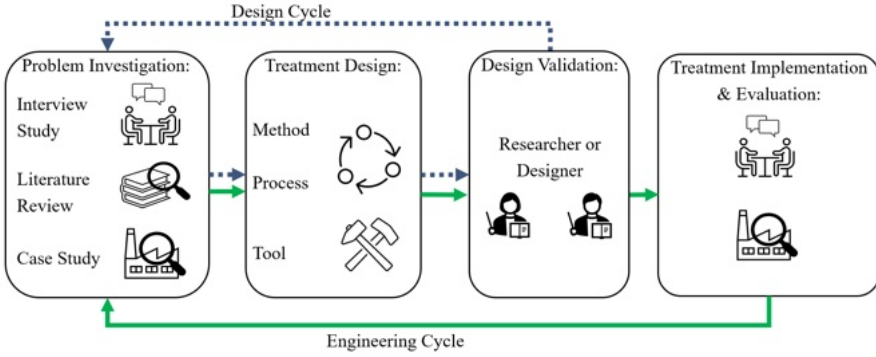


Figure 2.4: Illustrating the Design Science Research Methodology process. The design undergoes iterations within an internal cycle called the Design Cycle. It is then implemented and evaluated as part of a larger cycle known as the Engineering Cycle.

2.2.5 Design Science

The design science paradigm enables systematic problem investigation, treatment design and design evaluation to answer design questions. Hevner et al. [52] recommended seven guidelines to be followed in each phase of design science, including problem relevance, design evaluation, research rigor, and design as a search process. Design science is an iterative process comprising two cycles: one internal (the Design cycle) and one external (the Engineering cycle) [53], as shown in Fig. H.3. The Design cycle is the internal cycle, beginning with problem investigation, followed by treatment design, and concluding with treatment validation. Once validation is satisfactory, the validated treatment is applied in a real-world environment, and its effectiveness is evaluated. The engineering cycle iterates until the evaluation of the treatment in the real world is deemed satisfactory.

In Paper E, G, and H, Design Science serves as the foundational methodology framework, targeting the iterative design and evaluation of a prototype through interviews, case studies, and experiments. This iterative process facilitates the continuous improvement of the pipelines and frameworks, addressing identified weaknesses in the treatment or new LLM/VLM limitations discovered through validation and real-world evaluation.

2.3 Methodological Considerations

There are multiple considerations to account for before, during, and after designing a study, depending on the subject under study, the chosen method, and the data collection strategies.

2.3.1 Level of Researcher's Involvement

The level of involvement from the researcher and participants should be decided in the planning phase of the study, and the study should be designed to adhere to that decision. For instance, in an exploratory case study, the researcher does not have control over the setting, unlike in an experimental study, which requires full control [43].

2.3.2 Generalizability (Universal vs. Particular)

The degree to which the conclusions of a study can be extended or applied to different contexts, such as time or populations, is known as generalizability. Lack of generalizability is one of the threats to external validity which shall be considered during the design. For instance, it is a known weakness for case studies that they typically report one single case [43]. It is important for the researcher to carefully analyze the external validity of the results and clearly report on the scope and limitations to avoid overgeneralizing their findings.

The focus of this study is on safety processes aligned with international standards, and regulations, which companies are required to follow to ensure compliance. By aligning the study with these globally recognized frameworks and involving industrial experts, the generalizability of the findings can be enhanced. In addition, each study employs further measures to enhance generalizability, such as selecting experts from various companies from both OEMs and suppliers.

2.3.3 Ethical Considerations

The researcher shall analyze the study with respect to various ethical considerations, and appropriate measures should be implemented to ensure ethical integrity in the research. The lessons learned from more mature fields such as medicine can be adapted to the specific field and used as guidelines, similar to what Strandberg [54] proposed. He prescribed ethical recommendations and considerations for each step of interview studies in software engineering. For instance, consent, confidentiality, and scientific value are among the ethical principles that should be considered. Considering these principles would lead to designing limitations and mitigation strategies, such as anonymization. However, there are also novel and immature aspects, such as the usage of generative AI, that lack consensus on the ethical and legal guidelines to be followed. Transparent discussions among researchers and reporting on these can serve as initial steps to mitigate risks. Employing critical thinking about what could potentially go wrong is another vital mechanism. In this thesis, ChatGPT was used to improve readability and assist with proofreading; however, all ideas and scientific content were developed solely by the author.

Chapter 3

Summary of Included Papers

This chapter is a summary of all included papers. First, the problems they aimed to address are described. Then, the supporting methodology is explained, and finally, the results and contributions are summarized. Fig. 3.1 presents the breakdown and traceability between the main research goal, the two sub-goals, and the four main research questions addressed by each paper. Papers A and B provide a bird's-eye view of the challenges and existing solutions in both the literature and industry. Papers C to I each address one or more of these challenges, which are presented in Fig. 3.2

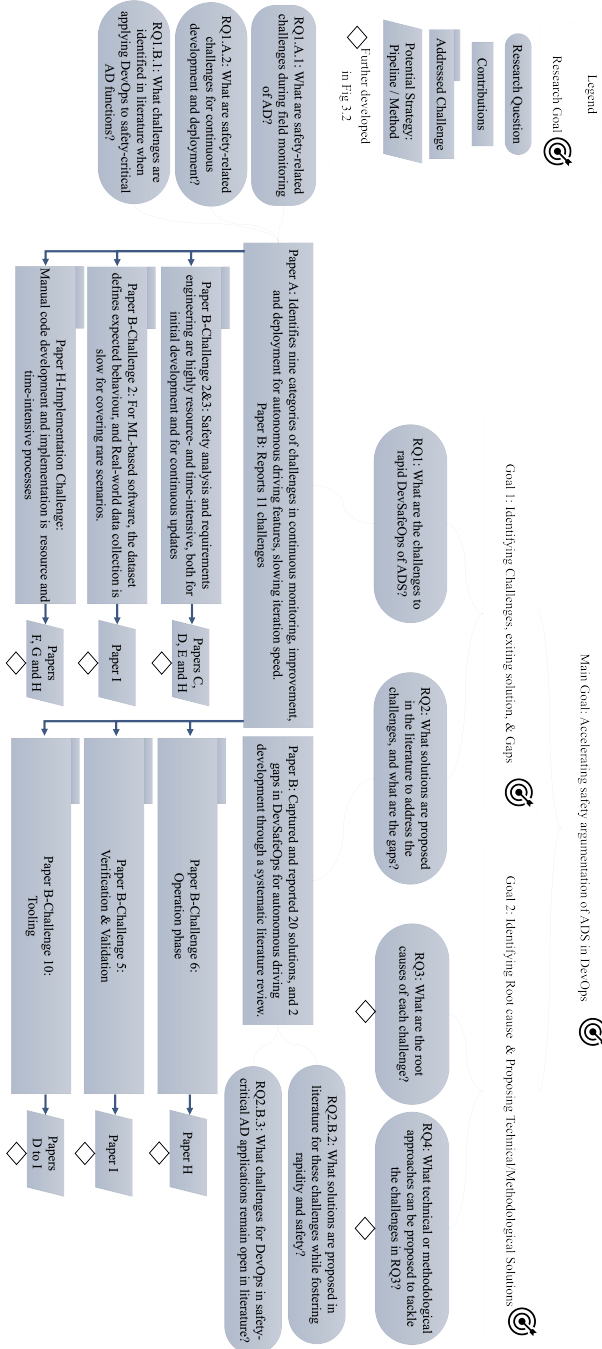


Figure 3.1: Presenting the traceability from the main research goal to the detailed research questions and the contribution of each paper. The arrows indicate the connection between the publications, from the identification of challenges to the proposed approaches. Publications A and B contribute to RQ1, while Publication B alone contributes to RQ2. Six challenges are addressed by Goal 2 in Papers C to I, which are presented in detail in Fig. 3.2. Safety analysis and requirements engineering challenges are addressed in Papers C, D, E, and H. Moreover, since data acts as a requirement for ML-based software, Paper I addresses this aspect of the challenge. Although implementation was not identified as a challenge in Papers A and B, it can be accelerated with the rise of LLMs, which is explored in Papers F, G, and H. Paper H also addresses challenges in the operation phase through runtime scenario and hazardous event detection. Paper I addresses verification and validation challenges through scene reconstruction. Finally, Papers D to I introduce tools for accelerating DevSafeOps and analyze their limitations and capabilities, thereby addressing the tooling challenge.

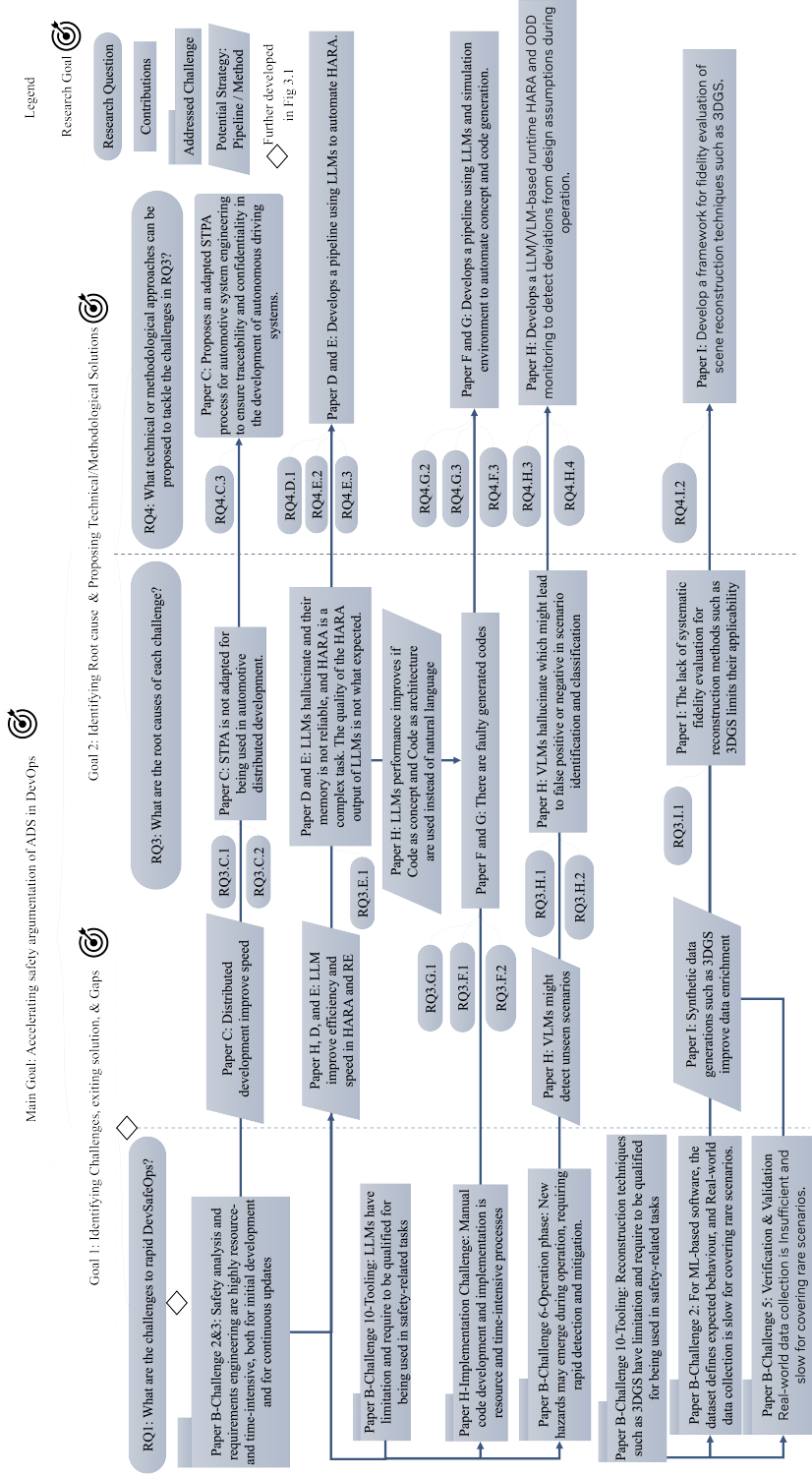


Figure 3.2: Presenting the traceability between the contributions of each paper and the main research questions. The challenges (RQ1), shown on the left side, are addressed through potential strategies. Subsequently, the root causes are identified (RQ3) and addressed on the right side (RQ4). The figure also illustrates the connections between the individual papers, their respective contributions, and the research questions (see Tables [I.1](#) and [I.2](#)).

3.1 Paper A - An Industrial Experience Report about Challenges from Continuous Monitoring, Improvement, and Deployment for Autonomous Driving Features

Problem: The paper addressed the challenges faced by an OEM in rapid continuous development, deployment, and monitoring in the context of complex and safety-critical applications such as autonomous driving features.

Methodology: The findings were aggregated from existing relevant safety standards, reported experiences, and interviews with eight experts in this field.

Contribution: The paper summarizes the iterative DevOps process for ADS, highlighting its critical role in ensuring safety of ADS, beyond the known advantages observed in other industries. Then, paper A identifies key challenges associated with the rapid continuous development, deployment, and monitoring of ADS, through an in-depth analysis based on industry standards and expert interviews, presenting them in nine clusters. Paper A provides an overview of ADS development, compiling a detailed list of challenges and identifying research gaps, compared to literature [24], [55], [56], which addresses specific challenges.

3.2 Paper B - The DevSafeOps Dilemma: A Systematic Literature Review on Rapidity in Safe Autonomous Driving Development and Operation

Problem: This study aimed to identify and cluster reported challenges and relevant solutions to the rapidity of DevSafeOps in the literature.

Methodology: A systematic literature review was performed on DevOps in safe autonomous driving development. A total of 319 papers was screened. 59 of them were selected based on predefined inclusion and exclusion criteria, and an additional 3 studies were identified through forward and backward snowballing. In total, 62 studies were included in the data extraction and synthesis process. To complement the academic sources with relevant industrial perspectives, we studied and analyzed all publicly available Voluntary Safety Self-Assessment (VSSA) reports from ADS developers published on the NHTSA website. This was followed by forward and backward snowballing to identify additional relevant industrial reports and publications.

Contribution: 11 challenges, 26 solutions, and identified gaps were discussed. The reported solutions were then mapped to the challenges. Requirement and safety analysis updates on the left side of the V model, verification and validation on the right side of the V model, and safety argumentation and certification updates are among the identified challenges. Data-driven development [57], shadow mode testing [58], and dynamic safety cases [59] are some of the proposed solutions for improving the speed of safety activities. Through the analysis

of VSSA reports, an imbalance is observed in the aspects that companies emphasize and report. For instance, most companies mainly reported on verification and validation aspects, such as simulation environments (e.g., [60], [61], [62], [63]) and real-world testing (e.g., [64], [65], [66], [67], [68], [69]), while providing limited coverage of design-phase activities, such as requirements engineering (only [70]) and safety analysis (only [71]).

Paper B introduces a novel and comprehensive systematic literature review of DevOps challenges specific to safety-critical automotive functions such as ADS, uniquely mapping these challenges to state-of-the-art solutions, which has not been fully addressed in prior literature (cf. Munk and Schweizer [31], Siddique [32], Fayollas [58]).

3.3 Paper C - Using STPA for Distributed Development of Safe Autonomous Driving: An Interview Study

Problem: To ensure the completeness and effectiveness of safety analysis methods such as STPA, the entire system, including all detailed software units and hardware components, must be included in the analysis. However, the inherent complexity of the AD system and the need to protect intellectual property necessitate modularization and abstraction levels, as followed by ISO 26262. Unlike FMEA and FTA, STPA is not inherently designed for multiple abstraction levels.

Methodology: A literature review was conducted to capture state-of-the-art processes and proposals. The identified challenges and the effectiveness of the proposed modifications were validated through a semi-structured interview study with 14 participants.

Contribution: The existing STPA process from the main four guidelines [13], [25], [33], [34] was reviewed, and a literature review [72], [73], [74], [75], [76] was conducted. The aggregated STPA process was then adapted and mapped to automotive systems engineering, specifically for a complex system such as AD. The proposed process was then tailored for subsystem teams to meet both traceability and confidentiality requirements. First, this involved replacing the activities in step 1 (i.e., defining the purpose of analyses) with a collection of received requirements from steps 3 and 4 of the system team. Second, in step 2 (i.e., modeling the control structure), the process involved encapsulating and abstracting the rest of the system (i.e., out of the scope of supply) as a controlled process. Hence, the proposed adaptations address the challenges of using STPA in distributed automotive systems engineering, compared to existing literature [72], [73], [74], [75], [76] and guidelines (cf. ISO 21448 [13], SAE J3187 [25], STPA Handbook [33]).

3.4 Paper D - Welcome Your New AI Teammate: On Safety Analysis by Leashing Large Language Models

Problem: The study investigates the feasibility of using a state-of-the-art LLM as a potential solution to accelerate the safety analysis activities in rapid DevOps.

Methodology: The feasibility of the concept was examined by prototyping and applying it to a specific function, and the results were reviewed.

Contribution: In this study, we designed a pipeline for the LLM by breaking the HARA process into sub-tasks, each handled by an individual LLM. Relevant prompt engineering techniques were then identified and used to create a prompt template for each sub-task. The study demonstrated the potential of LLMs to be employed in performing HARA, which was investigated and validated in detail in Paper E. This paper's contributions propose a framework for fully automating HARA using LLMs for the first time, in contrast to existing literature, where LLMs are used as supportive text generation tools without automating the safety analysis (cf. Qi et al. [77], Diemert and Weber [78]).

3.5 Paper E - Engineering Safety Requirements for Autonomous Driving with Large Language Models

Problem: This study investigated the limitations of LLMs for specifying safety requirements for AD functions by performing HARA. It then explored effective task breakdown and prompt engineering to improve the accuracy and efficiency of LLMs in specifying safety requirements.

Methodology: Design science was used in designing the prototype. In the first engineering cycle, the results of the system were evaluated by nine experts from three companies. In the second engineering cycle, after making improvements, a case study at an OEM was performed. A cloud-based internal LLM was employed in the case study, and the responsible team for HARA used the prototype for an internal function.

Contribution: This study builds on Paper D, which represented the first engineering cycle. In this paper, further engineering cycles were conducted by refining the pipeline and its prompts. The results generated by the prototype for a less established function were evaluated and reviewed by domain experts, and their feedback was collected. Additionally, the overall performance of the models was assessed against ten different Key Performance Indicators (KPIs). In the second engineering cycle, the prototype was first presented, and then the experts tested it, with their reflections reported in the paper. LLM's limitations in this specific task, such as misunderstandings of domain-specific terms and insufficient interpretation of non-textual information such as diagrams or figures, were identified and reported.

Moreover, a threat to the validity of a commonly used experiment (e.g., testing

the LLM for AEB [77]), which involves testing the LLM’s output for a well-known function against publicly available baselines, was investigated, analyzed, and reported. The final prototype, including the pipeline and the prompt templates, is presented in the paper. The weaknesses and limitations, such as irrelevant or incorrectly formulated requirements in the sample output of the prototype, were reported with examples from review reports. On average, the KPIs were fulfilled in most of the HARA as reported in detail in the paper. The experts found the prototype useful, although they emphasized the absolute necessity of human supervision and review over the output, which aligns with our goal. This paper introduces a novel, structured approach for engineering safety requirements in autonomous driving using LLMs, validated through a systematic process that contrasts with existing literature, which lacks both comprehensive frameworks and rigorous validation methods (cf. Qi et al. [77], Diemert and Weber [78]).

3.6 Paper F - Large Language Models in Code Co-generation for Safe Autonomous Vehicles

Problem: Due to stochastic behavior [79] and hallucination [40] of LLMs, multiple tries might be needed to reach the expected output. Hence, it is crucial to pre-evaluate generated code to filter out non-compilable, non-executable, or faulty candidates before delivering them to engineers for review.

Methodology: An industrial Software-in-the-Loop environment (esmini) is employed to evaluate six state-of-the-art LLMs (CodeLlama, CodeGemma, DeepSeek-r1, DeepSeek-Coders, Mistral, GPT-4) on four code generation tasks. This contrasts with existing benchmark-based code generation and ranking approaches (cf. Liu et al. [80], Lyu et al. [81], Siddiq et al. [82]), thereby reducing the risk of benchmark leakage [83]. The functions are designed at four different levels of complexity to examine how task complexity and novelty might affect the performance of LLMs. F1 (i.e., braking based on ego speed) focuses only on the capability of the generated code to adhere to the defined interface (APIs and outputs), follow the required input–output structure, and be integrable into a predefined software architecture without any manual modification. In addition, F2 (lane change when another vehicle is present) requires understanding the state of surrounding agents. Adaptive Cruise Control (ACC) is selected as F3, which is a real industrial function. However, as there are multiple examples of this function publicly available, we introduced Unsupervised Collision Avoidance by Evasive Manoeuvre to reduce the risk of benchmark leakage.

Contribution: In this study, we designed and implemented an automated code generation pipeline which delivers the code together with a preliminary evaluation to enable our proposed “generate fast, eliminate fast” concept. The pipeline automatically evaluates whether the generated code compiles, runs, and passes scenario-based test cases, and the results are used to rank candidates. We highlighted how task characteristics such as benchmark leakage and task complexity affect the quality of the generated code. We designed four

functions with different complexities to evaluate the outputs quantitatively and qualitatively. We also reported on recurring failure-mode to support the review process. Finally, we proposed a framework to integrate the proposed pipeline into the software engineering process in a safety-related automotive process. A clear performance gap is observed between simple functions (F1, F2) and advanced functions (F3, F4) in the quantitative results. Moreover, the performance ranking in some cases changes as the complexity of the function increases and the risk of benchmark leakage decreases. While all LLMs could deliver multiple code versions that pass all test cases for F1, GPT-4 was the only model that could deliver a successful code version for F4. None of the open-weights models was able to deliver a fully successful code for CAEM; however, CodeGemma:7B and Mistral:7B were able to generate successful code for ACC, which is a less complex and more established function. The quantitative comparison further shows that there is no direct relationship between the number of model parameters and the observed performance. Through a quantitative analysis, multiple failure modes are identified, which can be grouped into three main clusters, which are presented in detail in the paper. Unlike previous studies focusing on stand-alone code generation [80, 81, 82, 84], in paper F, we target automatically generated code that is directly compilable and integratable into an existing architecture and is pre-evaluated against a set of safety-relevant acceptance criteria, before human review and rigorous testing. Moreover, we propose an LLM-augmented process to ensure the safety of their outputs.

3.7 Paper G - On Simulation-Guided LLM-based Code Generation for Safe Autonomous Driving Software

Problem: LLMs' output, such as code, is generated stochastically, learned from existing knowledge, which does not necessarily follow required deterministic behaviors. This might lead to incorrect output, especially if the task is not seen before in the training data. Moreover, the LLM, similar to a programmer, would require iterations on the same code based on feedback to improve the quality of the output instead of elimination of the previous code and starting from scratch.

Methodology: A Design Science approach is employed to design, implement, and evaluate the proposed pipeline. Initially, through a literature review, known weaknesses and limitations of LLMs in code generation tasks are identified. The proposed pipeline, in each Design and Engineering Cycle, is evaluated through experimentation in an industrial simulation environment. The final version of the pipeline is presented to eleven experts, and the pipeline is validated by asking open-ended and multiple-choice questions.

Contribution: In this study, we initially identified the limitations and weaknesses of LLMs in code generation tasks. A potential root cause is the LLM's lack of an integrated deterministic world model, which can reduce hallucination.

This world model also acts as a safeguard for the LLM’s output by grading it based on satisfied acceptance criteria. Hence, we designed and developed a fully automated pipeline consisting of five main components: a specification prompt, an LLM, a simulation environment, a report generator, and a code correction prompt. This pipeline enables an automated conversation between the LLM and the simulation environment, closing the loop between the LLM and an abstract representation of a world model.

The pipeline is LLM-agnostic, allowing the integration of models ranging from small, local open-weights LLMs to proprietary cloud-based LLMs. The LLM receives the function description and non-functional requirements such as interface descriptions and available functions and classes through the prompt. The generated code is then evaluated in a simulation environment (i.e., esmini), and a test report is generated. Based on this report, a code correction prompt is produced, and the LLM generates a new version of the code. The correction loop did not only improve the number of satisfied test cases by 9.2%, but also enabled the exploration of diverse algorithmic strategies that would otherwise be eliminated without iteration, as they require parameter (e.g., Time-to-Collision thresholds) tuning. Unlike prior work, which relies on human-in-the-loop feedback, manual verification, or user-guided iteration to improve LLM-generated code [80, 82, 84, 85, 86, 87], we propose an automated LLM–simulation conversation that provides a feedback for iterative code improvement without human intervention

3.8 Paper H - Accelerating DevSafeOps through Large Language Models: Opportunities and Challenges

Problem: Given the capabilities and limitations of LLMs identified in Papers D to G, the question arises as to which additional activities within the DevSafeOps cycles can be accelerated. Furthermore, how can these pipelines be integrated to further enhance DevSafeOps efficiency? Since LLMs behave similarly to humans in some aspects and differently in others, and the current safety process is designed to be performed by human engineers, there may be a need to adapt the process. This adaptation may involve tailoring or modifying the process itself, as well as the nature of its inputs and outputs, in order to improve the integrability of LLMs into DevSafeOps and mitigate the risks they may introduce.

Methodology: Design science is employed to develop a multi-agent, LLM-enabled DevSafeOps framework composed of multiple pipelines, each responsible for a specific activity within the DevSafeOps process. Each pipeline is initially designed and evaluated in isolation using multiple relevant methods, including literature reviews, interview studies, case studies, and experimental evaluations conducted across iterative design cycles. The final design of each pipeline is derived from these validation steps. Subsequently, a separate interview study is conducted to first identify the expectations and wish lists of the participants

and then to showcase the overall framework. This evaluation involves 29 participants from two OEMs, two software suppliers, two Tier 1 suppliers, two consultancy companies in the automotive domain, and one from the IT sector.

Contribution: This study initially gathered the potential use cases of LLM-enabled tools and the expectations on them in automotive systems engineering through interview studies, literature review, and a case study. The identified use cases range from chatbot and translation-related features toward analysis, concept generation, V&V, and safety argumentation activities. We discussed the required qualification level of each use case and identified feasible use cases such as HARA, code generation, and operation-phase monitoring, based on the current limitations and strengths of the technology.

Employing LLMs in HARA through natural language scenario descriptions, as presented in Papers D and E, demonstrated potential not only for the first design-phase version but also for extension to the operation phase. Through VLMs, this capability can be expanded to runtime HARA, improving both known and unknown risk identification. This enables a continuous feedback loop between operation and design, strengthening the DevSafeOps cycle. Moreover, we propose employing VLMs as an additional system to detect the driving scenario, compare it against the ODD, request a driver handover when necessary, and trigger data collection for further analysis and learning.

While natural language remains the predominant approach in requirements engineering, it inherently suffers from ambiguity. On the other hand, as demonstrated in Papers F and G, LLMs are capable of generating executable code, and when integrated into a simulation environment, this code can even be evaluated against predefined safety-related test cases. Hence, we propose the concept of Code-as-a-Concept and Architecture, which addresses ambiguity in natural language specifications and improves concept validation through SIL and HIL-based pre-evaluation. Furthermore, toolchains for maintaining and validating code are significantly more mature than those for natural language artefacts, which also improves traceability and maintainability. Using code as an artefact to communicate between agents provides a less ambiguous representation of safety requirements.

Unlike prior works that employ LLMs primarily as chatbots [88], engineering assistants [88], or embedded components within AD software [89] or HMI systems [90, 91, 92, 93], this study focuses on systematically improving the efficiency of DevSafeOps cycles. Rather than treating LLMs as supportive tools for isolated tasks, we tailor the DevSafeOps process itself to enable structured and governed integration of LLM- and VLM-enabled pipelines. To the best of our knowledge, existing literature has not proposed a process-level adaptation of DevSafeOps specifically designed for multi-agent LLM integration across safety analysis, implementation, and runtime monitoring.

3.9 Paper I - From Concept to Capability: Evaluating 3D Gaussian Splatting for Synthetic Scene Editing in Autonomous Driving

Problem: The safety and robust performance of ADS rely on models trained and tested on large-scale datasets. Moreover, continuous improvement of these models requires data-intensive and rapid DataOps within DevSafeOps cycles. However, the collection and labeling of real-world driving data are resource-intensive and slow. Additionally, certain scenarios are rare, difficult or dangerous to capture through real-world data collection. Therefore, scene reconstruction techniques such as Neural Radiance Fields (NeRF) [94] and 3D Gaussian Splatting (3DGS) [95] are considered promising approaches. However, ensuring the correlation between the outputs of these models and real-world conditions remains a significant challenge, particularly since such data may be used for the training, testing, and validation of safety-related models.

Methodology: The performance of multiple 3DGS models is evaluated through experimentation on novel-view reconstruction. The models are first examined using real-world datasets such as Waymo Open Dataset [96], Zenseact Open Dataset [97], and KITTI [98]. However, existing real-world datasets do not provide ground truth for comparison with reconstructed novel viewpoints. Existing studies validate performance using a percentage of data captured along the original camera trajectory and hence, they evaluate interpolation between neighboring viewpoints rather than extrapolation beyond the original trajectory. For instance, according to [99], every 10th frame (i.e., a 10% split), and according to [95], every other frame (i.e., a 50% split), from the original trajectory is selected as the testing dataset to be used as ground truth in the final evaluation. Such evaluations provide valuable preliminary insight into rendering fidelity, which might be sufficient for ADAS functions that control only speed and braking. However, they do not assess reconstruction reliability for deviations from the original trajectory, particularly lateral and vertical deviations, which are critical in ADS applications that require closed-loop testing for lateral motion. To address this limitation, we employ the CARLA [37] simulation environment, which enables the execution of the same scenario from multiple novel viewpoints. These viewpoints can serve as ground truth and are compared against the reconstructed scenes generated by 3DGS from the original viewpoint. These rendered views serve as ground truth and are compared against reconstructions generated by 3DGS models trained on the original viewpoints. The novel viewpoints are distributed at varying lateral, longitudinal, and vertical distances relative to the training trajectory to assess extrapolation capability of the reconstruction. Two categories of evaluation metrics are applied. First, standard measures (SSIM [99] and PSNR [99]) for evaluating visual fidelity are used to assess reconstruction quality. Second, since the reconstructed scenes are intended for AD perception tasks, we evaluated the quality deviations from a machine-vision perspective using object detection models (i.e., YOLO [48] and DETR [49]) and semantic segmentation metrics (mIoU [50] and Dice [51]).

Contribution: In Paper I, we proposed and implemented a framework to systematically and dynamically analyze the capabilities and limitations of scene reconstruction techniques in safety-related contexts. The framework defines fidelity Key Performance Indicators (KPIs) across controlled deviations from the original viewpoint, enabling the identification of correlation boundaries beyond which reconstruction quality degrades significantly. Experimental results show that even a 10-meter longitudinal deviation maintains precision and recall above 80%, whereas a 3-meter lateral deviation causes some metrics to drop below 60%. The framework is agnostic to “reconstruction method” and can be applied to evaluate, compare, and qualify different techniques for acceptable use in safety-related scenario reconstruction.

Chapter 4

Discussion

This section discusses the contribution of each paper to the main goal of this thesis and the links between them. Then, identified threats to validity and the mitigation strategies will be presented.

4.1 Contributions

DevOps processes, methods, and tools are well-studied and quite mature, continually improving with industry advancements. However, rapid DevSafeOps remains relatively new, as highlighted in Paper A and B, which shed light on several research opportunities to confirm the identified challenges and better address them with relevant solutions. Several challenges and their corresponding solutions are identified in Papers A and B, with the aim of reporting the gaps in the scientific field to be further investigated by our study and other researchers. Fig. 3.1 presents the challenges addressed in this thesis, while Fig. 3.2 illustrates the traceability between these challenges and the contributions of Papers C to I, which first identify the root causes and then propose corresponding approaches.

We highlighted the required prerequisites for certain solutions, as they depend on foundational enablers. For instance, Model-Based Systems Engineering (MBSE) serves as an enabler for solutions such as model-based simulation, since machine-readable design artefacts are necessary to support verification and validation [100], [101], [102]. We also discussed the potential of certain solutions to be applied to other challenges. For instance, contract-based design, proposed as a remedy for safety argumentation, can also be employed to other challenges.

In some proposed solutions, such as the one for requirements updates, the proposed approach can be seen as a solution only for one aspect of the challenge. For instance, “data-driven development” [57], [103] is only applicable to the lowest software abstraction level and not any higher levels, such as the function or system level. Hence, there is a need for further studies to identify new methodological or technological solutions suitable for other abstraction levels and compatible with natural language.

One potential reason for the limited number of papers is the necessity of an industrial setting for both the identification and validation of proposed approaches, making it more challenging without collaboration between academia and industry. Moreover, there is a need to analyze the published white papers by companies involved in ADS development. However, since these white papers are not peer-reviewed, their inclusion in academic papers might face resistance, as observed through feedback received from reviewers. This shed light on the importance of stronger partnerships between academia and industry to enhance both the applicability and validation of research findings. As a remedy, we conducted a separate analysis of VSSA reports and incorporated the findings into Paper B to systematically capture insights from the industrial landscape.

One practical way to improve development speed and manage continuous updates in complex systems is to adopt a distributed development approach, enabling multiple teams, each responsible for specific module in the system. As discussed in paper C, STPA is one of the methods that requires adaptation to handle unbounded complexity and frequent changes in a distributed development ecosystem. However, this challenge was not observed in most papers and guidelines [33], [34], [73], [74] because the problem domain was simplified and abstracted in most research projects. Moreover, real industrial setup constraints, such as intellectual property and liability in distributed development were not considered. In all studied papers on STPA, the analysis was performed on a simplified and abstracted architecture. For instance, most of the papers focus on a vision-only sensor set, with a large black box representing its software. This limitation prevented extending the analysis to apply STPA to all aspects of systems of systems as reported in Paper C. Other limitations, such as different terminologies and missing traceability, along with proposed remedies, are reported and validated in Paper C.

Updating requirements and the corresponding safety analysis method is necessary in each iteration. However, due to the system’s complexity, multiple abstraction levels, and the involvement of several internal or external teams, analyzing and identifying all affected activities and requirements is a time- and resource-intensive task. So, existing methods and tools in the design phase should be adapted, or new ones proposed. These proposed approaches need to be carefully analyzed and qualified before they can be used in real-world automotive systems engineering.

Design-phase activities such as requirements engineering, safety analysis, and implementation were regarded as exclusively human tasks, as they required intellectual capabilities. Hence, during the initial preparation phase of Paper B, automation of design-phase activities was not among the identified solutions. However, as highlighted in final version of Paper B, during the review process, new publications proposed such solutions, including our own Papers D and E. In paper D, we performed a feasibility study on the capability of LLMs to be used in the automation of some aspects of design phase activities. In the very initial experiments, the model was asked to perform HARA in a single prompt. The quality, relevance, and correctness of the output in this phase of the experiment were not satisfactory. However, as we broke down the tasks into subtasks and improved the prompts, the performance of the models improved

as well. Although the system is absolutely not sufficient to be relied upon for providing the design, it might be seen as a teammate whose output needs to be verified and confirmed through review. The weaknesses and limitations of LLM-based safety analysis and requirements engineering are identified and reported in paper E, which need to be taken into consideration when using such a tool in similar safety-critical applications.

Although implementation was not initially identified as a challenge, the rise of LLMs and their capabilities in code generation motivated further investigation in Papers F, G, and H. This is because LLM-based code generation can enable the rapid introduction of new code to be integrated into the system as safety mechanisms to avoid or mitigate newly identified hazards. Tasks that would traditionally require several days for a team of programmers can potentially be completed in less than an hour. Furthermore, code generation serves as an enabler for code as concept and code as architecture, which can replace natural language specifications that may suffer from ambiguity. Moreover, requirements generated in natural language are not automatically verifiable within existing verification and validation setups, such as simulation environments. This limitation does not exist for requirements, which are represented directly as executable code.

In Paper F, we initially conducted experiments to evaluate the code generation capabilities of multiple LLMs, ranging from smaller open-weights models (e.g., Mistral) to larger cloud-based models (e.g., GPT-4). In addition, we proposed a framework for integrating LLM-based tools into DevSafeOps and discussed aspects of software tool evaluation and qualification in accordance with ISO 26262. Based on our observations, when provided with feedback regarding test results, these models are often able to correct their initial outputs. Accordingly, we designed and developed a pipeline that enables a closed-loop interaction between the LLM and a simulation environment, allowing the LLM to receive feedback from predefined test cases. In this pipeline, the simulation environment acts as an abstract world model, mitigating hallucination.

In Paper H, we first identified potential use cases for LLM-enabled engineering tools through engagement with industrial practitioners and a case study. We extended the concepts introduced in Papers D to G and incorporated VLMs to enable scenario understanding during the operation phase. These pipelines were integrated into an LLM-enabled DevSafeOps framework. Furthermore, we discussed the required tool evaluation and qualification considerations for each use case. This led us to cautiously limit the scope of automation to presented activities at this stage due to the current maturity, reliability, and qualification readiness of LLMs. For instance, some studies have focused on automating safety argumentation, since in most cases it is natural language-based and relies on existing evidence and processes. Given the strong summarization capabilities of LLMs, satisfactory results can often be achieved. However, safety argumentation represents the final opportunity to identify missing evidence or inconsistencies; therefore, it requires a high level of confidence that current LLMs are not yet sufficiently mature to provide.

Papers F and G address the design and implementation of rule-based software of ADS, while ML represents the other branch of software technology,

particularly dominant in end-to-end (E2E) paradigms. As highlighted in Paper B, data acts as requirements for machine learning [57], since model behavior is defined by the data distribution used for training. The performance, reliability and robustness of ML models depend on the diversity and representativeness of the training data. However, collecting comprehensive real-world driving data is slow, and often inadequate for capturing rare safety-related scenarios. Moreover, the challenge is expanded to verification and validation challenges for both rule- and ML-based software. Hence, reconstruction and scene-editing techniques can partially support data enrichment by enabling the generation of diverse and rare scenarios. In Paper I, we studied the capabilities and limitations of 3DGS and compared multiple state-of-the-art methods. Then, we proposed a framework for fidelity assessment.

While speed is a priority in DevOps, which might sacrifice quality in some cases, it is not a priority in safety-critical systems, and all activities outlined in the safety plan shall be satisfied before the product can be released. The mindsets of the safety and software communities are also different, and they do not readily accept concepts from each other. A lack of deep understanding of each other's concepts, processes, and terminologies, or different priorities, can be some of the potential root causes of the observed resistances. For instance, resistance to adding novel tools and methods to the safety tool-chain has led to outdated methods and tools in the safety tool-chain compared to those in software engineering. Bridging the gap between DevOps and safety is a challenging task that requires not only hard skills but also soft skills.

4.2 Threats to Validity

Feldt and Magazinius [104] proposed the following clustering of potential threats.

4.2.1 Conclusion Validity

Statistical assessment of the effectiveness of the proposed approaches in the papers C and E is challenging, as they are based on qualitative expert opinions obtained from in-depth interviews than on large and objective empirical evidence from systematic and randomized experiments. This is due to the nature of requirements engineering and safety analysis, which requires subjective evaluation by experts who have accumulated years of subject matter experience. Although in both papers C and E, we also gathered some quantitative data to support the conclusions drawn from qualitative data. In Paper C, 11 out of 14 experts recommended our approach, and in Paper E, we asked each expert to rate the quality of the prototype's output.

4.2.2 Construct Validity

As the research goal requires the applicability, efficiency, and effectiveness of the treatments in a real industrial setup, validation shall be done through industrial experts or settings. In papers A, C, E, G, and H this is done through

semi-structured interview studies. Mitigation strategies such as reviewing the detailed interview protocols and conducting pilot studies are employed to assess the effectiveness of the interview process before the data collection stage.

In the reconstruction study (paper I), PSNR and SSIM are employed to quantify image fidelity relative to ground truth. However, these metrics were originally employed for image quality assessment, rather than for AD perception needs. Consequently, high similarity does not necessarily imply high reliability for AD software development use case. To mitigate this limitation, perception-related metrics related to object detection and semantic segmentation are employed to better reflect machine-vision interpretability of reconstructed scenes.

4.2.3 Internal Validity

Controlling the environment in an industrial setting is hardly possible, which introduces the risk of the existence of other influential factors on the results for both interviews and case studies. Moreover, the background, knowledge, and experience of participants, who are the main source of data in papers A, C, and E, are influential factors that are not fully under the control of the researchers. Defining the participants' profiles and careful selection of them were applied to reduce these influential factors.

In the simulation-based studies (Papers F, G, H, and I), the pipelines, frameworks, and experimental setups rely on simulation environments (esmini and CARLA). Simulation environments are abstract and simplified mathematical models of the real world, which may introduce inaccuracies. In the studies related to code evaluation (Papers F, G, and H), esmini is employed to validate functional behavior. As esmini has been correlated with physical testing and is used in industrial contexts, this use case remains within the qualified operational boundary. In Paper I, the comparisons were performed within the same simulation environment (e.g., CARLA). Hence, systematic simulator-specific issues may influence both the reconstructed outputs and the ground truth equally. While this ensures internal consistency, it may not fully mitigate random inconsistencies.

4.2.4 Dependability and Credibility

In paper B, all steps in the SLR are followed and documented to improve the consistency and replicability. However, the replicability in papers A, C, and E are based on subjective opinions of industrial experts, which highly depend on their organizational processes and their level of familiarity with the subject, which may be seen as a threat to dependability of these studies. Relying on multiple experts with diverse backgrounds from different companies was employed as a mitigation strategy to improve dependability. Moreover, saturation of opinions was monitored to further support dependability.

In Papers D, E, F, G, and H, LLM is the main influential component of the prototype with respect to output quality. Thus, the stochastic nature of LLM threatens the replicability of the experiments. Iterative output generation by

the prototype and random selection of the samples to be sent for review were employed to reduce the stochastic effect of LLM on the quality of the results.

Additionally, due to the immaturity of open-weights models at the time of the papers D and E, a state-of-the-art commercial black-box model, GPT-4, was utilized. As the model is continuously being improved, the effectiveness of the prompts may vary between versions. This limitation is addressed in Papers F, G, and H, where the focus shifts to open-weights models with fixed versions and full control over their execution on local hardware. Moreover, in all papers the pipeline and the processes described in the prompts are generalizable, as they remain unaffected by changes in the model itself.

4.2.5 Confirmability

Although the experts who participated in the studies are free with their opinions, measures such as recorded interview questions and anonymization of influential factors were used to avoid biasing the participants by the researcher. For instance, in Paper C, the experts were not informed which method was proposed by us and which one was the original, to avoid bias. Similarly, in Paper E, experts were not informed whether the results were generated by an AI or an engineer.

4.2.6 External Validity

Practical aspects in automotive and regulatory frameworks specific to automotive might limit the generalizability of the results of this thesis to other domains. Although the findings and approaches proposed in this thesis can be seen as potential candidates in other domains, they require further validation regarding their applicability and effectiveness. However, the application of the adapted STPA proposed in paper C can also be extended to other safety-critical applications such as avionics, as the complexity, modularization, and confidentiality of each module follow similar constraints as automotive. The generalizability of the findings is not seen as a threat, as international standards in the automotive domain are widely used in industry to harmonize the way of working. Moreover, due to distributed development and its global network, harmonization occurs in other aspects such as techniques outside standards. Furthermore, the reported findings are presented at international industrial conferences and discussed to identify potential weaknesses of the findings for other parties. As LLMs and VLMs are a fast-evolving field with a non-deterministic nature, the validity of the proposed approaches shall be studied carefully, although they are reported to be potential tools to be considered in system engineering.

The proposed framework in Paper I helps to identify systematic limitations and weaknesses. These limitations are difficult, and sometimes impossible, to capture within real datasets due to the lack of ground truth, particularly in cases involving extrapolated trajectories. The use of reconstructed data generated with qualified deviations for V&V purposes may be acceptable, as it does not directly affect the AD software. However, their use for training purposes may require additional analysis to identify the effects of potential

failure modes of such data on ML-based software.

Additionally, in Paper I, the selected object detection and semantic segmentation models (e.g., YOLO, DETR, and SegFormer) influence the reported metrics. Hence, different architectures or training data may therefore lead to different quantitative outcomes. Similarly, in simulation-related studies, generalization of the simulation environment beyond qualified use cases may introduce limitations. However, in both cases, the proposed frameworks and pipelines are not bound to a specific environment or model, and each element in the framework can be replaced by relevant alternatives in the target domains.

4.2.7 Residual Risks

As Wohlin and Rainer [105] identified, research in software engineering is threatened by various factors such as misinterpretation of data by producers or consumers, vested interests and biases, and inappropriate study designs. Although we followed best practice frameworks and guidelines to avoid or mitigate these threats, it is challenging to claim completeness in identifying all possible issues. However, we have conducted a thorough methodological design aiming for reducing threats to validity. The study designs are provided in detail in the papers so that any potential missing threats to validity can be identified during the peer review process, although according to Wohlin and Rainer [105], the peer review process can sometimes fail to catch limitations.

Chapter 5

Conclusions and Future Research

In this thesis, we explored the challenges and existing solutions for improving the speed of each iteration loop in the DevSafeOps way of working. In our first contribution, we reported and clustered challenges and mapped existing solutions to each. Major gaps were then identified and reported. Change impact analysis, safety analysis, requirements engineering, and architectural design are the major activities on the left side of the V-Model that demand significant attention in a safety-related development process compared to other applications. Change impact analysis is an essential step in safety planning for any change request to identify required activities that affect the safety argument in each iteration of DevSafeOps. After providing a bird's-eye view of the DevSafeOps landscape, we initially focused on the left side of the V-Model, which represents the first step in the development process. The focus was on design phase challenges and approaches, as the safe-by-design approach is one of the main arguments on safety of AD, and requirements resulting from this phase serve as input and foundation to the implementation and verification phases. Subsequently, we extended the study to activities in the implementation, verification, validation, and operation phases.

The Traceability Information Model (TIM) is identified as a potential approach in paper B for improving speed in impact analysis. It is essential to enable multi-abstraction and modularization to satisfy both traceability and modularization in activities such as STPA, which was the focus of paper C. This allows each team to perform the analysis for their module while still being able to track changes across the entire product and maintain coherence with the current version of the product. Traceability is an enabler for the automation of impact analysis, while modularization is an approach proposed for handling the complexity of a software-intensive system like ADS.

In papers D and E, we developed an LLM-based prototype for safety analysis and requirements engineering. We reviewed its performance with the help of experts, and their review comments were reported in paper E.

In Papers F and G, we focused on the implementation of rule-based software

by employing LLMs. We designed and developed a world model (i.e., an abstract simulation environment) coupled with the LLM, which enables the “generate fast, eliminate fast” and a “closed-loop World-Language-Model” concepts. In Paper H, we proposed the use of code as concept and code as architecture based on the observed coding capabilities of LLMs and a comparison with NL-based tasks to enable seamless idea-to-code transitions for complex and safety-related systems such as AD. In Paper H, the potential use cases and associated safety considerations for each use case are reported. In Paper H, we further extended the concepts introduced in Papers D and E to the operation phase, thereby closing the DevSafeOps feedback loop by employing VLMs to enable runtime scenario detection.

In parallel with proposing approaches for accelerating continuous rule-based software development (Papers C to G), challenges in continuous ML-based software development are also studied. As data are essential for training models to exhibit the required behaviour, methods proposed for generating additional domain-specific scenarios are investigated as an approach. In paper I, we studied and reported the capabilities and limitations of 3DGS models for scene reconstruction and editing to support data generation. We also proposed a model-agnostic fidelity evaluation framework that can be applied to other similar scene reconstruction and editing techniques.

In this thesis, we first identified the challenges associated with rapid DevSafeOps, which is a novel contribution because the shortcomings of the existing way of working are identified in a systematic manner. A collection of approaches aiming to improve development speed while maintaining safety within DevOps cycles is also reported. This provides industry with a structured catalogue of existing approaches, while enabling researchers to identify gaps and focus on developing solutions to address them. We examined various solution approaches, including distributed development, generative AI, virtual environments, and synthetic data generation highlighting the need for research efforts that adopt interdisciplinary approaches spanning organizational, software, safety, and data-centric perspectives. In addition to the proposed technical approaches, each study systematically investigated appropriate evaluation strategies and methodological setups tailored to real-world industrial problems. Furthermore, we discussed, reported, and mitigated key threats to validity in studies with similar objectives when applied to real-world industrial contexts, such as oversimplified case examples (Paper C), benchmark leakage (Papers D to G), and the fidelity limitations of synthetic data (Paper I). Altogether, this thesis systematically identifies challenges, existing solutions, and gaps in rapid DevSafeOps for ADS, analyzes their root causes, and addresses them through targeted technical and methodological approaches in an industrial environment.

The continuation of root cause analysis for each identified challenge, along with the identification of relevant approaches to address verification, validation, and certification challenge, constitutes the next step. This involves developing new methods and tools, or identifying and adapting existing ones, to ensure their efficiency and effectiveness throughout iterative DevSafeOps loops.

Future studies may also include further case studies of the proposed solutions from Papers B and C within real industrial settings, reporting on their

effectiveness, efficiency, and practical applicability. Moreover, systematically comparing different solutions for specific challenges would be beneficial. For instance, analyzing the strengths and weaknesses of various safety analysis methods can support the selection of the most appropriate approach for each use case. Additionally, constructing a comprehensive and maintainable (and potentially automated) safety argumentation for ADS within iterative DevSafeOps loops is crucial.

Given the demonstrated potential of LLMs, future studies may further investigate their limitations and weaknesses in performing DevSafeOps tasks and develop strategies to avoid or mitigate them. Such efforts would enhance the reliability of these tools and increase confidence in their use within safety-related systems.

As the proportion of ML-based software in AD systems seems to expand, future studies may further investigate the explainability and safety of these models, particularly for end-to-end (E2E) approaches. A significant portion of architectural complexity that was previously addressed at the software architecture level is now embedded within model architectures (e.g., transformer-based neural network architectures). This makes it challenging to apply conventional cause-and-effect reasoning methods that were applicable to traditional software, such as software FMEA. This shift may necessitate the development of novel approaches or the adaptation of existing methods for selecting the most appropriate model architecture with respect to safety.

As reliance on ML-based software in ADS continues to increase, future studies may also include investigations into the explainability and safety of these models, particularly for end-to-end (E2E) approaches. Moreover, since the performance of these models is highly data dependent, industrial challenges may include detecting gaps in dataset and data acquisition for continuous model updates to mitigate distributional shifts, while preventing regressions and maintaining safety of ADS.

References

- [1] Volvo Cars, *Volvo Cars Press & Media*, <https://www.volvocars.com/intl/media/>, Accessed: 2026-05-03, 2026 (cit. on pp. 4, 7, 10, 21).
- [2] “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles,” SAE, Standard, 2014 (cit. on p. 3).
- [3] L. Jakobsson, K. Bohman, B. Svanberg and T. Victor, “Occupant protection for AD cars—the paradigm shift in crash safety?” In *26th international technical conference on the enhanced safety of vehicles (ESV): Technology: Enabling a safer Tomorrow* National highway traffic safety administration, National Highway Traffic Safety Administration Eindhoven, Netherlands, 2019 (cit. on p. 3).
- [4] C. Tingvall and N. Haworth, “Vision Zero - An ethical approach to safety and mobility,” in *6th ITE International Conference Road Safety & Traffic Enforcement: Beyond 2000*, 1999 (cit. on p. 3).
- [5] M. Distner, M. Bengtsson, T. Broberg and L. Jakobsson, “City safety—a system addressing rear-end collisions at low speeds,” in *Proc. 21st International Technical Conference on the Enhanced Safety of Vehicles*, 2009 (cit. on p. 3).
- [6] I. Isaksson-Hellman and M. Lindman, “Real-world performance of city safety based on Swedish insurance data,” in *24th international technical conference on the enhanced safety of vehicles (ESV)*, Citeseer, vol. 8, 2015, pp. 15–0121 (cit. on p. 3).
- [7] I. Isaksson-Hellman and M. Lindman, “Real-world evaluation of driver assistance systems for vulnerable road users based on insurance crash data in Sweden,” in *Proceedings of the 26th International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, Eindhoven, The Netherlands, 2019 (cit. on p. 3).
- [8] R. Gudla, V. S. Telidevulapalli, J. S. Kota, G. Mandha et al., “Review on self-driving cars using neural network architectures,” *World Journal of Advanced Research and Reviews*, vol. 16, no. 2, pp. 736–746, 2022 (cit. on p. 4).

- [9] Q. E. T. Lawyer, “Report to the boards of directors of cruise LLC Gm cruise holdings LLC and general motors Holdings LLC regarding the October 2 2023 accident in San Francisco,” Quinn Emanuel Trial Lawyers, Jan. 2024 (cit. on p. [4](#)).
- [10] P. Koopman, “Anatomy of a Robotaxi Crash: Lessons from the Cruise Pedestrian Dragging Mishap,” *arXiv preprint arXiv:2402.06046*, 2024 (cit. on p. [5](#)).
- [11] UNECE, *UN Regulation No. 157 - Automated Lane Keeping Systems (ALKS)*, Standard, 2021. [Online]. Available: <https://unece.org/transport/documents/2021/03/standards/un-regulation-no-157-automated-lane-keeping-systems-alks> (cit. on pp. [6](#), [18](#)).
- [12] “ISO 26262:2018 (all parts), Road Vehicles — Functional Safety,” International Organization for Standardization, Standard, 2018 (cit. on pp. [6](#), [8](#), [16](#), [18](#)).
- [13] “ISO 21448:2022, Road Vehicles — Safety of the Intended Functionality,” International Organization for Standardization, Standard, 2022 (cit. on pp. [6](#), [7](#), [9](#), [17](#), [19](#), [45](#)).
- [14] “ISO/SAE 21434:2021, Road Vehicles — Cybersecurity Engineering,” International Organization for Standardization, Standard, 2021 (cit. on p. [6](#)).
- [15] “ISO/TS 5083:2025, Road vehicles - Safety for automated driving systems - Design, verification and validation,” International Organization for Standardization, Standard, 2020 (cit. on pp. [6](#), [7](#)).
- [16] “ISO/PAS 8800:2024, Road vehicles — Safety and artificial intelligence,” International Organization for Standardization, Geneva, Switzerland, Publicly Available Specification, Dec. 2024, Published December 2024. ISO/PAS 8800:2024 (Edition 1, 172 pages) (cit. on pp. [6](#), [7](#), [17](#)).
- [17] “ISO/TR 4804:2020, Road vehicles – Safety and cybersecurity for automated driving systems – Design, verification and validation methods,” International Organization for Standardization, Standard, 2020 (cit. on pp. [6](#), [8](#)).
- [18] P. Koopman and W. H. Widen, “Breaking the Tyranny of Net Risk Metrics for Automated Vehicle Safety,” *Available at SSRN 4634179*, 2023 (cit. on p. [6](#)).
- [19] P. Koopman and W. Widen, *Redefining Safety for Autonomous Vehicles*, 2024. arXiv: [2404.16768 \[cs.R0\]](#) (cit. on p. [6](#)).
- [20] Safety First for Automated Driving Consortium, “Safety First for Automated Driving,” Aptiv, Audi, Baidu, BMW, Continental, Daimler, FCA, HERE, Infineon, Intel, and Volkswagen, White paper, 2019. [Online]. Available: <https://www.aptiv.com/docs/default-source/white-papers/safety-first-for-automated-driving-aptiv-white-paper.pdf> (cit. on p. [7](#)).

- [21] P. Koopman, “Ul 4600: What to include in an autonomous vehicle safety case,” *Computer*, vol. 56, no. 05, pp. 101–104, May 2023, issn: 1558-0814. DOI: [10.1109/MC.2023.3236171](https://doi.org/10.1109/MC.2023.3236171) (cit. on p. [7](#)).
- [22] Waymo, “Waymo Safety Report,” Waymo LLC, Tech. Rep., 2020. [Online]. Available: https://downloads.ctfassets.net/e6t5diu0txbw/4mhZJxuCinbVNuyAKPPc0j/d1623d42ed7aeea46993c22ea7e50612/Waymo_Safety_Report_02-2021.pdf (cit. on p. [8](#)).
- [23] “Road vehicles – Data communication between sensors and data fusion unit for automated driving functions – Logical interface (ISO 23150:2021, IDT),” International Organization for Standardization, Standard, 2021 (cit. on p. [8](#)).
- [24] F. Warg, H. Blom, J. Borg and R. Johansson, “Continuous Deployment for Dependable Systems with Continuous Assurance Cases,” in *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 2019, pp. 318–325. DOI: [10.1109/ISSREW.2019.00091](https://doi.org/10.1109/ISSREW.2019.00091) (cit. on pp. [10](#), [44](#)).
- [25] F. S. Committee, *System Theoretic Process Analysis (STPA) Recommended Practices for Evaluations of Automotive Related Safety-Critical Systems*, https://doi.org/10.4271/J3187_202202, Feb. 2022 (cit. on pp. [10](#), [18](#), [20](#), [45](#)).
- [26] J. Li et al., “Exploring the causality of end-to-end autonomous driving,” *arXiv preprint arXiv:2407.06546*, 2024 (cit. on p. [15](#)).
- [27] J. Tang, Z. Zhou, Z. He, J. Zhang, K. Zhang and J. Pu, “Causalvad: De-confounding end-to-end autonomous driving via causal intervention,” *arXiv preprint arXiv:2603.18561*, 2026 (cit. on p. [15](#)).
- [28] T. Hu et al., “Vision-language-action models for autonomous driving: Past, present, and future,” *arXiv preprint arXiv:2512.16760*, 2025 (cit. on p. [15](#)).
- [29] NVIDIA et al., *Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail*, 2026. arXiv: [2511.00088](https://arxiv.org/abs/2511.00088) [cs.R0]. [Online]. Available: <https://arxiv.org/abs/2511.00088> (cit. on p. [15](#)).
- [30] H. Fu et al., “Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025, pp. 24 823–24 834 (cit. on p. [15](#)).
- [31] P. Munk and M. Schweizer, “DevOps and Safety? SafeOps! Towards Ensuring Safety in Feature-Driven Development with Frequent Releases,” in *Computer Safety, Reliability, and Security. SAFECOMP 2022 Workshops*, M. Trapp, E. Schoitsch, J. Guiochet and F. Bitsch, Eds., Cham: Springer International Publishing, 2022, pp. 145–157, ISBN: 978-3-031-14862-0 (cit. on pp. [15](#), [16](#), [45](#)).
- [32] U. Siddique, “SafetyOps,” *arXiv preprint arXiv:2008.04461*, 2020 (cit. on pp. [16](#), [45](#)).

- [33] N. G. Leveson and J. Thomas, “STPA Handbook, Mar. 2018,” <http://psas.scripts.mit.edu/home/materials>, (cit. on pp. 18, 20, 45, 54).
- [34] C. J. Becker, J. C. Brewer and L. J. Yount, “Safety of the intended functionality of lane-centering and lane-changing maneuvers of a generic level 3 highway chauffeur system (Report No. DOT HS 812 879),” in *National Highway Traffic Safety Administration*, 2020 (cit. on pp. 18, 45, 54).
- [35] X. Xing, T. Zhou, J. Chen, L. Xiong and Z. Yu, “A Hazard Analysis Approach based on STPA and Finite State Machine for Autonomous Vehicles,” in *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2021, pp. 150–156. DOI: [10.1109/IV48863.2021.9575425](https://doi.org/10.1109/IV48863.2021.9575425) (cit. on p. 19).
- [36] esmini contributors, *Esmini user guide*, esmini User guide, Accessed: 2024-05-01, 2024. [Online]. Available: <https://esmini.github.io> (cit. on pp. 20, 21, 35).
- [37] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez and V. Koltun, “Carla: An open urban driving simulator,” in *Conference on robot learning*, PMLR, 2017, pp. 1–16 (cit. on pp. 20, 21, 35, 51).
- [38] R. Bommasani et al., *On the Opportunities and Risks of Foundation Models*, 2022. arXiv: [2108.07258](https://arxiv.org/abs/2108.07258) [cs.LG] (cit. on p. 22).
- [39] J. White et al., *A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT*, 2023. arXiv: [2302.11382](https://arxiv.org/abs/2302.11382) [cs.SE] (cit. on p. 22).
- [40] O. Zheng, M. Abdel-Aty, D. Wang, Z. Wang and S. Ding, *ChatGPT is on the Horizon: Could a Large Language Model be Suitable for Intelligent Traffic Safety Research and Applications?* 2023. arXiv: [2303.05382](https://arxiv.org/abs/2303.05382) [cs.CL] (cit. on pp. 22, 47).
- [41] J. Bosch, H. H. Olsson and I. Crnkovic, “It takes three to tango: Requirement, outcome/data, and AI driven development.,” in *SiBW*, 2018, pp. 177–192. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/8417118/> (cit. on p. 24).
- [42] *Guide to Advanced Empirical Software Engineering. [electronic resource]*. Springer London, 2008, ISBN: 9781848000445. [Online]. Available: <https://search.ebscohost.com/login.aspx?direct=true&db=cacat07472a&AN=clec.SPRINGERLINK9781848000445&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds> (cit. on pp. 29, 30).
- [43] K.-J. Stol and B. Fitzgerald, “A Holistic Overview of Software Engineering Research Strategies,” in *2015 IEEE/ACM 3rd International Workshop on Conducting Empirical Studies in Industry*, 2015, pp. 47–54. DOI: [10.1109/CESI.2015.15](https://doi.org/10.1109/CESI.2015.15) (cit. on pp. 29, 40).

- [44] P. Runeson, M. Host, A. Rainer and B. Regnell, *Case Study Research in Software Engineering: Guidelines and Examples*. Wiley, 2012, ISBN: 9781118181003. [Online]. Available: <https://books.google.se/books?id=T7rXoaxqPIAC> (cit. on pp. 30, 33, 34).
- [45] B. A. Kitchenham and S. Charters, “Guidelines for Performing Systematic Literature Reviews in Software Engineering,” English, Keele University and Durham University Joint Report, Tech. Rep. EBSE 2007-001, 2007. [Online]. Available: https://www.elsevier.com/_data/promis_misc/525444systematicreviewsguide.pdf (cit. on p. 32).
- [46] V. R. Basili, R. W. Selby and D. H. Hutchens, “Experimentation in software engineering,” *IEEE Transactions on Software Engineering*, vol. SE-12, no. 7, pp. 733–743, 1986. DOI: [10.1109/TSE.1986.6312975](https://doi.org/10.1109/TSE.1986.6312975) (cit. on p. 34).
- [47] Z. Wang and A. C. Bovik, *Modern Image Quality Assessment*. Springer Cham, 2006. DOI: [10.1007/978-3-031-02238-8](https://doi.org/10.1007/978-3-031-02238-8) (cit. on p. 35).
- [48] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han et al., “Yolov10: Real-time end-to-end object detection,” *Advances in Neural Information Processing Systems*, vol. 37, pp. 107 984–108 011, 2024 (cit. on pp. 37, 51).
- [49] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov and S. Zagoruyko, “End-to-end object detection with transformers,” in *European conference on computer vision*, Springer, 2020, pp. 213–229 (cit. on pp. 37, 51).
- [50] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez and P. Luo, *Segformer: Simple and efficient design for semantic segmentation with transformers*, 2021. arXiv: [2105.15203 \[cs.CV\]](https://arxiv.org/abs/2105.15203). [Online]. Available: <https://arxiv.org/abs/2105.15203> (cit. on pp. 37, 51).
- [51] S. Jadon, “A survey of loss functions for semantic segmentation,” in *2020 IEEE conference on computational intelligence in bioinformatics and computational biology (CIBCB)*, IEEE, 2020, pp. 1–7 (cit. on pp. 37, 51).
- [52] A. R. Hevner, S. T. March, J. Park and S. Ram, “Design science in information systems research,” *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004, ISSN: 02767783. Accessed: 5th Nov. 2023. [Online]. Available: <http://www.jstor.org/stable/25148625> (cit. on p. 39).
- [53] R. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*. Jan. 2014, pp. 1–332, ISBN: 978-3-662-43838-1. DOI: [10.1007/978-3-662-43839-8](https://doi.org/10.1007/978-3-662-43839-8) (cit. on p. 39).
- [54] P. E. Strandberg, “Ethical Interviews in Software Engineering,” in *Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, Porto de Galinhas, Brazil, 2019, pp. 1–11. DOI: [10.1109/ESEM.2019.8870192](https://doi.org/10.1109/ESEM.2019.8870192) (cit. on p. 40).

- [55] M. Zeller, “Towards continuous safety assessment in context of devops,” in *International Conference on Computer Safety, Reliability, and Security*, Springer, 2021, pp. 145–157 (cit. on p. 44).
- [56] O. Willers, S. Sudholt, S. Raafatnia and S. Abrecht, “Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks,” in *International Conference on Computer Safety, Reliability, and Security*, Springer, 2020, pp. 336–350 (cit. on p. 44).
- [57] J. Bosch, H. H. Olsson and I. Crnkovic, “It Takes Three to Tango: Requirement, Outcome/Data, and AI Driven Development,” in *Proceedings of the International Workshop on Software-intensive Business: Start-ups, Ecosystems and Platforms (SiBW 2018), Espoo, Finland, December 3, 2018*, S. Hyrynsalmi, A. Suominen, C. Jud, X. Wang, J. Bosch and J. Münch, Eds., ser. CEUR Workshop Proceedings, vol. 2305, CEUR-WS.org, 2018, pp. 177–192. [Online]. Available: <https://ceur-ws.org/Vol-2305/paper14.pdf> (cit. on pp. 44, 53, 56).
- [58] C. Fayollas, H. Bonnin and O. Flebus, “SafeOps: A Concept of Continuous Safety,” in *2020 16th European Dependable Computing Conference (EDCC)*, IEEE, 2020, pp. 65–68. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9237051/> (cit. on pp. 44, 45).
- [59] P. Schleiss, F. Carella and I. Kurzidem, “Towards Continuous Safety Assurance for Autonomous Systems,” in *2022 6th International Conference on System Reliability and Safety (ICSRS)*, 2022, pp. 457–462. DOI: [10.1109/ICSRS56243.2022.10067323](https://doi.org/10.1109/ICSRS56243.2022.10067323) (cit. on p. 44).
- [60] N. Webb et al., “Waymo’s safety methodologies and safety readiness determinations,” *arXiv preprint arXiv:2011.00054*, 2020 (cit. on p. 45).
- [61] M. Schwall, T. Daniel, T. Victor, F. Favaro and H. Hohnhold, “Waymo public road safety performance data,” *arXiv preprint arXiv:2011.00038*, 2020 (cit. on p. 45).
- [62] Waymo, *Waymo safety report*, Accessed: 2025-02-28. [Online]. Available: <https://waymo.com/research/waymo-safety-report/> (cit. on p. 45).
- [63] Aurora, *Aurora Safety Report*, Accessed: 2025-02-28. [Online]. Available: <https://aurora.tech/safety> (cit. on p. 45).
- [64] TUSIMPLE, *TUSIMPLE SAFETY REPORT*, Accessed: 2025-02-28. [Online]. Available: <https://www.tusimple.com/wp-content/uploads/2021/03/TuSimple-Safety-Report.pdf> (cit. on p. 45).
- [65] Volkswagen, *Volkswagen’s Safety Self-Assessment for Automated Mobility Services*, Accessed: 2025-02-28. [Online]. Available: <https://media.vw.com/assets/documents/original/17321-VolkswagenVSSAV12023.pdf> (cit. on p. 45).

- [66] ZOOX, *SAFETY INNOVATION AT ZOOX*, Accessed: 2025-02-28. [Online]. Available: https://www.datocms-assets.com/106048/1696537818-zoox_safety_report_volume1_2018.pdf (cit. on p. 45).
- [67] Apple, *Our Approach to Automated Driving System Safety*, Accessed: 2025-02-28. [Online]. Available: <https://www.apple.com/ads/ADS-Safety.pdf> (cit. on p. 45).
- [68] a2z, *Safety Report a2z Smart City Solution*, Accessed: 2025-02-28. [Online]. Available: <https://autoa2z.ai/Safety> (cit. on p. 45).
- [69] Ford, *A Matter of Trust: Ford's Approach to Developing Self-Driving Vehicles*, Accessed: 2025-02-28. [Online]. Available: <https://media.ford.com/content/dam/fordmedia/North%5C%20America/US/2021/06/17/ford-safety-report.pdf> (cit. on p. 45).
- [70] Torc-Robotics, *Torc Safety Report*, Accessed: 2025-02-28. [Online]. Available: <https://torc.ai/wp-content/uploads/2024/06/Torc-Robotics-VSSA.pdf> (cit. on p. 45).
- [71] IKE, *IKE Safety Report*, Accessed: 2025-02-28. [Online]. Available: https://global-uploads.webflow.com/5d5c3db08993c73c35421c12/5d9e954a4a6e6c7c7b0804d4_Ike-SafetyReport-AC-03.pdf%5C#page=8.34 (cit. on p. 45).
- [72] A. Mallya, V. Pantelic, M. Adedjouma, M. Lawford and A. Wassyn, "Using STPA in an ISO 26262 Compliant Process," in *Computer Safety, Reliability, and Security - 35th International Conference, SAFECOMP 2016, Trondheim, Norway, September 21-23, 2016, Proceedings*, ser. Lecture Notes in Computer Science, vol. 9922, Springer, 2016, pp. 117–129. DOI: [10.1007/978-3-319-45477-1_10](https://doi.org/10.1007/978-3-319-45477-1_10). [Online]. Available: https://doi.org/10.1007/978-3-319-45477-1_10 (cit. on p. 45).
- [73] V. Nguyen Tran, L. Vu Tran, V. Nguyen Tran and D. Ngoc Vu, "Hazard Analysis Methods for Software Safety Requirements Engineering," in *Proceedings of The 5th International Conference on Software Engineering and Information Management (ICSIM)*, ser. ICSIM 2022, Yokohama, Japan: Association for Computing Machinery, 2022, pp. 11–18, ISBN: 9781450395519. DOI: [10.1145/3520084.3520087](https://doi.org/10.1145/3520084.3520087). [Online]. Available: <https://doi.org/10.1145/3520084.3520087> (cit. on pp. 45, 54).
- [74] A. Abdulkhaleq et al., "A Systematic Approach Based on STPA for Developing a Dependable Architecture for Fully Automated Driving Vehicles," *4th European STAMP Workshop 2016, ESW 2016, 13-15 September 2016, Zurich, Switzerland*, pp. 41–51, 2017, ISSN: 1877-7058. DOI: <https://doi.org/10.1016/j.proeng.2017.03.094>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877705817312109> (cit. on pp. 45, 54).

- [75] A. Abdulkhaleq, S. Wagner and N. Leveson, “A Comprehensive Safety Engineering Approach for Software-Intensive Systems Based on STPA,” *In proceedings of the 3rd European STAMP Workshop 5-6 October 2015, Amsterdam*, pp. 2–11, 2015, ISSN: 1877-7058. DOI: <https://doi.org/10.1016/j.proeng.2015.11.498> [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877705815038588> (cit. on p. 45).
- [76] E. Acar Celik, C. Cărlan, A. Abdulkhaleq, F. Bauer, M. Schels and H. J. Putzer, “Application Of STPA For The Elicitation Of Safety Requirements For A Machine Learning-Based Perception Component In Automotive,” in *Computer Safety, Reliability, and Security: 41st International Conference, SAFECOMP, Munich, Germany, September 6–9, 2022, Proceedings*, Munich, Germany: Springer-Verlag, 2022, pp. 319–332, ISBN: 978-3-031-14834-7. DOI: [10.1007/978-3-031-14835-4_21](https://doi.org/10.1007/978-3-031-14835-4_21) [Online]. Available: https://doi.org/10.1007/978-3-031-14835-4_21 (cit. on p. 45).
- [77] Y. Qi, X. Zhao, S. Khastgir and X. Huang, *Safety Analysis in the Era of Large Language Models: A Case Study of STPA using ChatGPT*, 2023. arXiv: [2304.01246](https://arxiv.org/abs/2304.01246) [cs.CL] (cit. on pp. 46, 47).
- [78] S. Diemert and J. H. Weber, *Can Large Language Models assist in Hazard Analysis?* 2023. arXiv: [2303.15473](https://arxiv.org/abs/2303.15473) [cs.HC] (cit. on pp. 46, 47).
- [79] E. M. Bender, T. Gebru, A. McMillan-Major and S. Shmitchell, “On the dangers of stochastic parrots: Can language models be too big?” In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, ser. FAccT ’21, Virtual Event, Canada: Association for Computing Machinery, 2021, pp. 610–623, ISBN: 9781450383097. DOI: [10.1145/3442188.3445922](https://doi.org/10.1145/3442188.3445922) (cit. on p. 47).
- [80] M. Liu, J. Wang, T. Lin, Q. Ma, Z. Fang and Y. Wu, “An Empirical Study of the Code Generation of Safety-Critical Software Using LLMs,” *Applied Sciences*, vol. 14, no. 3, 2024, ISSN: 2076-3417. DOI: [10.3390/app14031046](https://doi.org/10.3390/app14031046) (cit. on pp. 47–49).
- [81] Z. Lyu, X. Li, Z. Xie and M. Li, “Top pass: Improve code generation by pass@ k-maximized code ranking,” *Frontiers of Computer Science*, vol. 19, no. 8, p. 198 341, 2025 (cit. on pp. 47, 48).
- [82] M. L. Siddiq, J. C. Da Silva Santos, R. H. Tanvir, N. Ulfat, F. Al Rifat and V. Carvalho Lopes, “Using Large Language Models to Generate JUnit Tests: An Empirical Study,” in *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*, 2024, pp. 313–322 (cit. on pp. 47–49).
- [83] K. Zhou et al., “Don’t Make Your LLM an Evaluation Benchmark Cheater,” *arXiv preprint arXiv:2311.01964*, 2023 (cit. on p. 47).

- [84] V. Liventsev, A. Grishina, A. Härmä and L. Moonen, “Fully Autonomous Programming with Large Language Models,” in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO ’23, ACM, Jul. 2023. DOI: [10.1145/3583131.3590481](https://doi.org/10.1145/3583131.3590481). [Online]. Available: <http://dx.doi.org/10.1145/3583131.3590481> (cit. on pp. 48, 49).
- [85] E. Nijkamp, B. Pang, H. Hayashi, L. Tu, H. Wang and Y. Zhou, “CodeGen: An Open Large Language Model for Code with Multi-Turn Program Synthesis,” *arXiv preprint arXiv:2203.13474*, 2023. arXiv: [2203.13474](https://arxiv.org/abs/2203.13474) [cs.LG] (cit. on p. 49).
- [86] P. Vaithilingam, T. Zhang and E. L. Glassman, “Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models,” in *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, ser. CHI EA ’22, New Orleans, LA, USA: Association for Computing Machinery, 2022, ISBN: 9781450391566. DOI: [10.1145/3491101.3519665](https://doi.org/10.1145/3491101.3519665) (cit. on p. 49).
- [87] M. Chen, J. Tworek, H. Jun, Q. Yuan and H. P. de Oliveira Pinto, “Evaluating large language models trained on code,” *CoRR*, vol. abs/2107.03374, 2021. arXiv: [2107.03374](https://arxiv.org/abs/2107.03374). [Online]. Available: <https://arxiv.org/abs/2107.03374> (cit. on p. 49).
- [88] Ford Motor Company. “Ai wary,” Ford 2024 Trend Report, Ford Motor Company, Accessed: 30th Apr. 2026. [Online]. Available: <https://corporate.ford.com/microsites/ford-trends-2024/ai-wary.html> (cit. on p. 50).
- [89] Wayve. “Lingo-1: Exploring natural language for autonomous driving,” Wayve, Accessed: 30th Apr. 2026. [Online]. Available: <https://wayve.ai/thinking/lingo-natural-language-autonomous-driving/> (cit. on p. 50).
- [90] Tesla, Inc. “Grok,” Tesla Support, Tesla, Inc., Accessed: 30th Apr. 2026. [Online]. Available: <https://www.tesla.com/support/grok> (cit. on p. 50).
- [91] CNBC. “Gm explores using chatgpt in vehicles,” CNBC, Accessed: 30th Apr. 2026. [Online]. Available: <https://www.cnbc.com/2023/03/10/gm-explores-using-chatgpt-in-vehicles.html> (cit. on p. 50).
- [92] Mercedes-Benz Group. “Mercedes-benz takes in-car voice control to a new level with ChatGPT,” Mercedes-Benz Group, Accessed: 30th Apr. 2026. [Online]. Available: <https://group.mercedes-benz.com/innovation/digitalisation/connectivity/car-voice-control-with-chatgpt.html> (cit. on p. 50).
- [93] N. Marks. “Generative AI developers harness NVIDIA technologies to transform in-vehicle experiences,” NVIDIA Blog, Accessed: 30th Apr. 2026. [Online]. Available: <https://blogs.nvidia.com/blog/generative-ai-in-vehicle-experiences/> (cit. on p. 50).

- [94] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021 (cit. on p. 51).
- [95] G. Hess, C. Lindström, M. Fatemi, C. Petersson and L. Svensson, “Splatad: Real-time lidar and camera rendering with 3d gaussian splatting for autonomous driving,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 11 982–11 992 (cit. on p. 51).
- [96] P. Sun et al., “Scalability in perception for autonomous driving: Waymo open dataset,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454 (cit. on p. 51).
- [97] M. Alibeigi et al., “Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023 (cit. on p. 51).
- [98] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, “Vision meets robotics: The kitti dataset,” *The international journal of robotics research*, vol. 32, no. 11, pp. 1231–1237, 2013 (cit. on p. 51).
- [99] Z. Chen et al., “Omnire: Omni urban scene reconstruction,” *arXiv preprint arXiv:2408.16760*, 2024 (cit. on p. 51).
- [100] D. F. Blanco, F. Le Mouél, T. Lin and M.-P. Escudié, “A comprehensive survey on software as a service (saas) transformation for the automotive systems,” *IEEE Access*, 2023 (cit. on p. 53).
- [101] M. Mörtstrand, *Continuous verification of automotive software*, 2023 (cit. on p. 53).
- [102] G. Tomar, “Migration of model-based software components to service-oriented architectures,” *ATZelectronics worldwide*, vol. 19, no. 5, pp. 50–54, 2024 (cit. on p. 53).
- [103] S. Kugele and M. Broy, “Architecture as a Backbone for Safe DevOps in Automotive Systems,” in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2022, pp. 4145–4150. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9922097/> (cit. on p. 53).
- [104] R. Feldt and A. Magazinius, “Validity threats in empirical software engineering research-an initial survey,” in *Seke*, 2010, pp. 374–379 (cit. on p. 56).
- [105] C. Wohlin and A. Rainer, “Challenges and recommendations to publishing and using credible evidence in software engineering,” *Information and software technology*, vol. 134, p. 106 555, 2021 (cit. on p. 59).