Sentinel: A Software Architecture for Safe Artificial Intelligence in Autonomous Vehicles

Sentinel: A Software Architecture for Safe Artificial Intelligence in Autonomous Vehicles

By

Spencer Deevy, B.Eng. Software Engineering

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfillment of the Requirements for the Degree

Master of Applied Science

McMaster University

© Copyright by Spencer Deevy, September 2019

Master of Applied Science (September 2019) McMaster University

(Software Engineering) Hamilton, Ontario

Title:	Sentinel: A Software Architecture for Safe Artificial Intelligence in Autonomous Vehicles
Author:	Spencer Deevy, B.Eng. Software Engineering (McMaster University)
Supervisors:	Dr. Alan Wassyng, Dr. Mark Lawford
Number of Pages:	x, 106

Lay Abstract

Artificial intelligence techniques are enabling increased autonomy in autonomous vehicles, however, a major concern that has yet to be addressed is how to ensure vehicle safety while utilizing potentially volatile artificial intelligence. This issue in addition to the lack of existing high-level roadmaps towards fully autonomous vehicles are the driving factors behind this work. We provide Sentinel, a safety-oriented software architecture for autonomous vehicles utilizing artificial intelligence techniques. The literature intersecting the fields of artificial intelligence and autonomous vehicles was reviewed to identify unique safety considerations and mitigation strategies that could be applied. The Sentinel architecture was then synthesized from information in the literature review, while also taking into consideration design choices that would increase industry adoption of Sentinel. An assurance case was then constructed to determine the assumptions and evidence required to justify the non-negative impacts of the architecture on the safety or reliability of a vehicle employing the Sentinel design.

Abstract

Trends in the automotive industry indicate rapid adoption of artificial intelligence techniques such as machine learning algorithms, enabling increasingly capable autonomous vehicles. However, the major focus has been to improve the performance and accuracy of these techniques, with a clear lack of development towards corresponding safety systems. Artificial intelligence techniques are characterized by high complexity, high variability, and low diagnosability. These issues all pose risks to the safety of autonomous vehicles and need to be taken into consideration as we move towards fully autonomous vehicles.

Sentinel, a fault-tolerant software architecture is presented as the main contribution of this thesis. Sentinel has been designed to mitigate safety concerns surrounding artificial intelligence techniques employed by upcoming SAE J3016 level 5 autonomous vehicles. The architecture design process involved careful consideration of issues inherent to artificial intelligence techniques being utilized in autonomous vehicles and their corresponding mitigation strategies. Following this, a survey of software architectures was conducted, drawing inspiration from existing autonomous vehicle architectures as well as architectures in the related domains of artificial intelligence, organic computing, and robotics. These existing architectures were then iteratively combined, guided by an autonomous vehicle hazard analysis, resulting in the final architecture.

Additionally, an assurance case was constructed to delineate the assumptions and evidence required to justify the continued safety of autonomous vehicles employing the Sentinel architecture. This work is presented to provide a safety-oriented framework towards fully autonomous vehicles.

Acknowledgments

I would first like to thank my thesis advisors Dr. Alan Wassyng and Dr. Mark Lawford of the Department of Computing and Software at McMaster University. They consistently allowed this paper to be my own work but steered me in the right direction whenever they thought I needed it. Their patience, knowledge, and guidance played a large role in the production of this thesis, and I thank them for their support. I extend my thanks to Dr. Saeid Habibi of the Department of Mechanical Engineering at McMaster University for sponsoring this research. Additionally, I would like to thank each member of my thesis committee: Dr. Alan Wassyng, Dr. Mark Lawford, Dr. Saeid Habibi, and Dr. Ryszard Janicki of McMaster University for their support and evaluation of this thesis.

I thank my fellow labmates for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last few years.

Finally, I must express my very profound gratitude to my parents and to my brother for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Contents

Descriptive Note ii

Lay Abstract iii

Abstract v

Acknowledgments vi

Table of Contents x

List of Figures xi

List of Tables xiii

List of Acronyms xiv

Declaration of Academic Achievement xvi

1 Introduction 1

1.1 Motivation 1

1.1.1 AI Autonomy 2

1.1.2 Autonomous Vehicles 3

1.2 Objectives 4

1.3 Contributions 5
1.4 Outline 6

2 Preliminaries **7**

2.1 Autonomous Vehicles 7
2.2 Artificial Intelligence 9
2.3 Fault-tolerance 11
2.4 Fully Autonomous Vehicle Challenges 12

3 Relevant Research **14**

3.1 Search Method 15
3.2 Supervised Machine Learning 16
3.2.1 Training Dataset 16
3.2.1.1 Safety Concerns 16
3.2.1.2 Mitigation Strategies 16
3.2.2 Model Feature Selection 17
3.2.2.1 Safety Concerns 17
3.2.2.2 Mitigation Strategies 17
3.2.3 Optimization Strategy 18
3.2.3.1 Safety Concerns 18
3.2.3.2 Mitigation Strategies 19
3.3 Reinforcement Learning 19
3.3.1 Exploration 20
3.3.1.1 Safety Concerns 20
3.3.1.2 Mitigation Strategies 20
3.3.2 Reward Hacking 21
3.3.2.1 Safety Concerns 21

3.3.2.2 Mitigation Strategies . . . 22
3.4 Artificial Intelligence Architectures . . . 22
3.4.1 Arguing Machines Framework . . . 23
3.4.2 Runtime Assurance Framework . . . 24
3.4.3 Adaptive Supervisor Architecture . . . 26
3.5 Organic Computing Architectures . . . 27
3.5.1 Hybrid Bio-inspired Control Architecture . . . 28
3.5.2 Multi-Layer Observer/Controller Architecture . . . 29
3.6 Autonomous Vehicle Architectures . . . 30
3.6.1 Cognitive ADAS Architecture . . . 31
3.6.2 Autonomous Vehicle Control Architecture . . . 33
3.6.3 Autonomous Vehicle Decomposition Framework . . . 34
3.6.4 Safe-AV Architecture . . . 37
3.7 Hazard Analysis . . . 41
3.8 Summary . . . 48

4 Sentinel **49**
4.1 Design Methodology . . . 50
4.2 Architectural Overview . . . 51
4.2.1 Operational Modules . . . 54
4.2.2 Support Modules . . . 55
4.2.3 Safeguard Modules . . . 57
4.3 Module Decomposition . . . 58
4.3.1 Sensors . . . 59
4.3.2 Perception . . . 59
4.3.3 Planning . . . 62

4.3.4 Plan Execution 64
4.3.5 Actuator Interface 65
4.3.6 Actuators 65
4.3.7 Agent Interface 66
4.3.8 Data Management 67
4.3.9 Simulation 68
4.3.10 Safety 69
4.3.10.1 Proactive Safety 70
4.3.10.2 Reactive Safety 70
4.3.10.3 Control Mode Selector 71
4.4 Revision History 72
4.4.1 Milestone 1 72
4.4.2 Milestone 2 75
4.4.3 Milestone 3 77
4.4.4 Milestone 4 79

5 Assurance Case **81**
5.1 Assurance Case Diagrams 83
5.2 Discussion 90

6 Conclusion **92**
6.1 Results 92
6.2 Discussion 93
6.3 Future Research 96

List of Figures

2.1 SAE J3016 Levels of Driving Automation [19] 8

3.1 Arguing Machines Framework Diagram [12] 23
3.2 Runtime Assurance Framework Diagram [6] 25
3.3 Adaptive Supervisor Architecture Diagrams [22] 26
3.4 Hybrid Bio-inspired Control Architecture Diagrams [5] 28
3.5 Multi-layer Observer/Controller Architecture Diagram [28] . . . 29
3.6 Cognitive ADAS Architecture Diagram [7] 31
3.7 Autonomous Vehicle Control Architecture Diagram [27] 33
3.8 Autonomous Vehicle Functional Decomposition Diagram [34] . . 35
3.9 Safe-AV Architecture Diagram [35] 38

4.1 Sentinel Abstraction Level 1 52
4.2 Sentinel Abstraction Level 2 53
4.3 Sentinel Milestone 1 74
4.4 Sentinel Milestone 2 76
4.5 Sentinel Milestone 3 78
4.6 Sentinel Milestone 4 80

5.1 Excerpt from Eurocontrol WAM PSC High-Level Argument [31] 82

5.2 Top-level claim of J3016 compliance and safety enhancement . . 83
5.3 J3016 compliance claim 1/5 84
5.4 J3016 compliance claim 2/5 84
5.5 J3016 compliance claim 3/5 85
5.6 J3016 compliance claim 4/5 86
5.7 J3016 compliance claim 5/5 86
5.8 Improvement in safety attributes over existing architectures 1/3 87
5.9 Improvement in safety attributes over existing architectures 2/3 88
5.10 Improvement in safety attributes over existing architectures 3/3 89

List of Tables

3.1 Hazard Analysis Safety Requirements and Supporting Architectures 47

List of Acronyms

Acronyms

ADAS	advanced driver assistance system	32
AI	artificial intelligence	1
DDT	dynamic driving task	57
ECU	electronic control unit	65
GPS	global positioning system	59
GSN	goal structuring notation	81
HMI	human machine interface	35
IMU	inertial measurement unit	59
IRL	inverse reinforcement learning	20
LOC	lines of code	17
MIT	Massachusetts Institute of Technology	12
ML	machine learning	1
NASA	National Aeronautics and Space Administration	11
OEM	original equipment manufacturer	39
RL	reinforcement learning	12
SAE	Society of Automotive Engineers	8

STPA systems theoretic process analysis 41

TMR triple modular redundancy 79

V2I vehicle-to-infrastructure 32

V2V vehicle-to-vehicle 32

V2X vehicle-to-everything 30

Declaration of Academic Achievement

I, Spencer R. Deevy, declare this thesis to be my own work. I am the sole author of this document. No part of this work has been published or submitted for publication or for a higher degree at another institution. To the best of my knowledge, the content of this document does not infringe on anyone's copyright.

My supervisors, Dr. Alan Wassyng and Dr. Mark Lawford, as well as the members of my thesis committee, Dr. Alan Wassyng, Dr. Mark Lawford, Dr. Saeid Habibi, and Dr. Ryszard Janicki, have provided guidance and support at all stages of this project. I completed all of the research work contained within this document.

Chapter 1

Introduction

This chapter provides the motivational factors and the overarching structure of this thesis. The motivation for this work is described first in the general context of autonomy and artificial intelligence in section 1.1.1, followed by motivations concerned with the specific application domain of autonomous vehicles in section 1.1.2. The research objectives and scope of the work undertaken are then described in section 1.2. Finally, a list of contributions stemming from this work is given in section 1.3, followed by an outline of the remainder of this thesis in section 1.4.

1.1 Motivation

Artificial intelligence (AI) techniques such as machine learning (ML) algorithms are useful tools to create systems that exhibit complex behaviour with growing levels of autonomy. Machine learning research remained stagnant for decades following its inception in the 1950s due to the lack of large quantities of annotated data needed to yield useful algorithms. The development

of the internet, mass collection of user data, and increasing computational capabilities resulted in ludicrous amounts of well-annotated data now known as Big Data. As such, in recent years there has been a massive resurgence in machine learning development with little signs of abatement. This growth can likely be attributed to the ability of machine learning algorithms to offer unrivalled performance on a multitude of complex tasks, often far surpassing human capabilities, and other computational techniques.

Recent advances in deep learning techniques and neuromorphic computing are now enabling technologies normally reserved for science fiction novels, such as smart cities, predictive medical monitoring and diagnostic tools, and autonomous vehicles. Initially drowned out by the promise of intelligent autonomous machines aiding humanity, several concerns are arising that illuminate the ethical implications and safety risks such systems could pose if left unchecked and unsupported by complementary approaches.

1.1.1 AI Autonomy

Superintelligent machines marking the end of humanity is a cautionary tale told in countless science fiction stories, but perhaps it is the wrong story to tell. While this possibility is on the horizon, a danger much closer to fruition is that of incompetent AI being trusted with tasks that put human lives at risk.

The machine learning algorithms employed in many of these systems have to learn from training data, which is primarily curated and supplied by humans. Should the data target the wrong task, if rare examples are not given, or if environmental conditions change, the AI could perform poorly. Worse still is

that the results in simulation and the laboratory may perform exceptionally, giving developers and engineers a false impression that the system will perform just as well in the field. Yet another disadvantage is that these AI systems, once deployed, are black boxes. Diagnosing a fault in these systems is nearly impossible, and often the most one can say is that the training data was insufficient to handle the current circumstances.

Now, with all of these disadvantages, why are we still utilizing these methods? The answer is simply that no other methods exist that are as flexible or as adept at complex tasks as machine learning. As such, the vast majority of research in the machine learning community has been focused on improving the performance and accuracy of machine learning models. Largely neglected, however, are the underlying concerns surrounding non-deterministic adaptive AI and methods to mitigate the risks involved when AI is used in safety-critical systems.

1.1.2 Autonomous Vehicles

Several systems in the vanguard of artificial intelligence development are those pertaining to autonomous vehicles. The incorporation of machine learning algorithms into various vehicular systems is rapidly being adopted by automotive manufacturers due to their ability to supplant tasks normally reserved for human drivers. Features such as automatic cruise control, automatic lane changing, driver monitoring systems and networked communication capabilities are incrementally improving the autonomy of vehicles, with the eventual goal of complete vehicle autonomy.

As was implied in the previous section, machine learning techniques must

be incorporated into safety-critical systems with utmost caution and with a substantial effort put towards ensuring safe operation in the event of a failure due to faulty AI output. A troubling concern for the automotive domain is that the standard governing functional safety of autonomous vehicles, *ISO 26262* [1], does not currently incorporate specific considerations for utilizing machine learning and other artificial intelligence techniques in automotive systems. Some preliminary research and recommendations have been done to this end, but the aggregation and incorporation of these recommendations into automotive design have not been conducted. This is a crucial step needed to ensure the adoption of safe AI development and deployment practices in upcoming autonomous vehicles to avoid the many concerns intrinsic to AI systems.

1.2 Objectives

To delineate the goals of this thesis within the context of autonomous vehicles, a set of research objectives are given below, followed by concisely defining the scope in which this work is to be conducted.

- Determine the unique properties of AI techniques that directly affect the safety of autonomous systems
- Determine mitigation strategies and system architectures that make use of these techniques to mitigate AI safety concerns
- Design a system architecture combining safety techniques in an attempt to promote safe AI use in fully autonomous vehicles

- Conduct an assurance case analysis on the resulting architecture to provide theoretical claims on compliance with SAE J3016 level 5 recommendations

The scope of these research objectives is restricted to AI techniques readily being incorporated into automotive vehicles, and those being utilized in applications that share a majority of their functionality with autonomous automotive technology. Additionally, while all subsystems of autonomous vehicles will be considered and acknowledged, the focus of this work is on ensuring safe AI integration and utilization. Therefore the components involved in system safety and AI safety in particular are the primary concern of this work.

1.3 Contributions

The contributions of this thesis are as follows:

- Review and commentary on several architectures applicable to fully autonomous vehicles
- Sentinel, a novel software architecture for SAE J3016 level 5 autonomous vehicles, that aims to achieve adequate safety even when AI techniques are used in producing the functional behaviour of the vehicle
- An assurance case demonstrating Sentinel compliance with SAE J3016 level 5 recommendations.

1.4 Outline

The remainder of this thesis is structured as follows. Chapter 2 focuses on background information pertaining to artificial intelligence, machine learning, and autonomous vehicles, in order to prime the reader for the information contained in the next chapter. Following this, Chapter 3 consists of a literature review on AI safety concerns, existing mitigation strategies, and safety-oriented architectures that may apply to AI and autonomous vehicles. Utilizing this information, Chapter 4 presents a proposed architecture for incorporating AI safely in autonomous vehicles. An analysis of the proposed architecture is then presented via an assurance case in Chapter 5. A final summary and discussion are provided in Chapter 6 to conclude this work and give direction for future research.

Chapter 2

Preliminaries

This chapter aims to provide the much-needed background information and context for the work contained within. Terms pertaining to related fields, the state of autonomous vehicle development, and key high-level issues faced by the industry are described. This background information is given as a primer to ease the transition into the more detail-oriented Chapter 3.

2.1 Autonomous Vehicles

To begin, it is prudent to discuss what autonomous vehicles are, how they differ from existing vehicles, and what the ultimate goal is that they aim to achieve. An autonomous vehicle is a highly automated version of a regular automotive vehicle, usually referring to 4-wheeled varieties rather than motorcycles. Existing automotive vehicles either have no automation or low automation, requiring the majority of driving tasks to be completed by the human operator. Autonomous vehicles aim to progressively replace driving tasks normally reserved for the human operator with built-in functionality, shifting

the responsibility of control from the human to the vehicle.

To refine the notion of varying degrees of automation, the Society of Automotive Engineers (SAE) released the SAE J3016^{TM} Levels of Driving Automation [19], seen in Figure 2.1. These automation levels range from no automation at level 0 to full automation under all conditions at level 5.

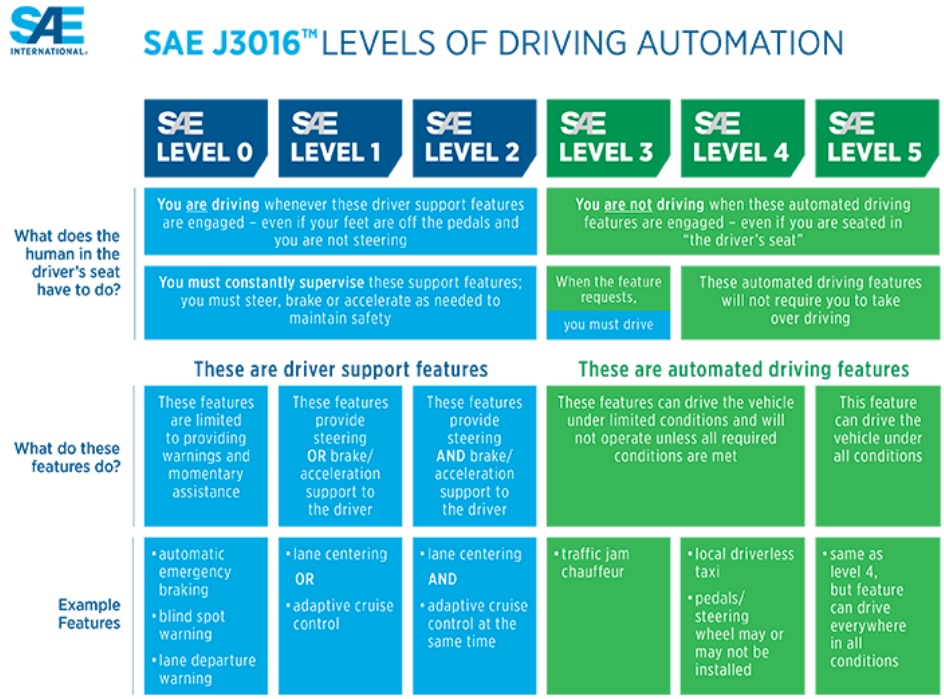


Figure 2.1: SAE J3016 Levels of Driving Automation [19]

Currently on the market are level 2 features such as the GM Super Cruise, the Tesla Autopilot and the Nissan ProPilot. Emerging on the market and currently being refined are level 3 features as demonstrated in the new Audi A8 and some of the upcoming features in the Tesla Autopilot. Additionally, much of ongoing research and early testing focuses on level 4 autonomy with a notable demonstration being given by the BMW Group at their NEXTGen event [16], showcasing autonomous navigation as well as pedestrian/obstacle detection and avoidance. At this level of autonomy, all driving responsibility

is placed on the vehicle and is intended to work under a very precise set of environmental conditions. The vehicles utilize advanced sensor suites and a host of machine learning software providing the basis for their intelligent behaviour, including autonomous navigation, predictive planning, and obstacle avoidance. These test vehicles are logging tens of terabytes of information, to be processed later by AI algorithms to improve vehicle functionality. This trend towards increasingly complex autonomous vehicles utilizing artificial intelligence mechanisms such as machine learning has no end in sight. Many view this approach as our only option towards achieving level 5 autonomy.

2.2 Artificial Intelligence

Artificial intelligence can be broadly defined as the study of intelligent agents, where an intelligent agent normally refers to a device that perceives its environment and takes actions that maximize the chance of the agent achieving its goals. Many different approaches have been undertaken to create AI systems, with the earliest methods collectively referred to as symbolic artificial intelligence. These systems manipulated collections of high-level symbolic representations of problems and goals, with the most popular of these methods being expert systems. Symbolic AI methods fell out of practice because the symbolic manipulation rules would need to be code distilled from human experts in the related fields. This was time-consuming and was limited in the accuracy and complexity of the results the systems could provide.

After this stage in AI development, new statistical methods were brought to the forefront. Methods such as support vector machines and k-nearest-neighbours used statistical analysis on large quantities of data to arrive at

new conclusions. Largely used for classification problems, these methods provided more nuanced and accurate results than symbolic AI systems. Despite the advancement over previous AI methods, statistical analysis methods were only really applicable for small numbers of variables and relatively simplistic classification problems.

To overcome these challenges, machine learning techniques began to shift towards those based on biological neural structures. Genetic algorithms, neural networks, deep neural networks, and bayesian deep learning are all examples of biologically-based AI methods. Given large quantities of data and sufficient computational resources, these algorithms not only produce more accurate results than symbolic or statistical methods but are also capable of handling problems of much higher complexity.

These machine learning algorithms are supplied with enormous amounts of data, and in most systems via a reward and/or loss function, alter themselves to achieve better performance than their previous iterations. In neural networks, this can be adjusting neuron weights, and genetic algorithms often run several variants in parallel while only letting sufficiently adept variants on to the next iteration. While these methods are extremely powerful in their ability to solve complex problems, a measure of uncertainty is always present. If the provided training data is poorly curated, if unintended solutions to the goal are not considered, if the machine learning algorithm is underfit or overfit, there will be issues with the performance, accuracy, and validity of the results. As machine learning and similar AI techniques expand into problem domains of increasing scope, these issues will only worsen. As such, methods to safeguard against AI use in safety-critical systems must be considered, especially in this work targetting fully autonomous vehicles.

2.3 Fault-tolerance

A key component in safety-critical systems is fault-tolerance techniques. In the event of component or system failures, fault-tolerance techniques are employed to eliminate and/or mitigate those faults. Some examples of fault-tolerant systems are the National Aeronautics and Space Administration (NASA) X-38 spacecraft [21] and the Airbus A320 aircraft [41]. Each of these systems employs redundant flight computers and sensor suites to ensure high reliability and fault-tolerance of individual component failures. An unfortunate recent example of a lack of fault-tolerance is seen in the Boeing 737 Max 8 incidents [29], where a lack of adequate sensor redundancy, coding errors, and insufficient pilot training resulted in several of these aircraft crashing. As such, it is prudent to make use of fault-tolerance techniques where possible in upcoming autonomous vehicles to ensure passenger safety.

The difficulty in applying traditional fault-tolerance techniques to artificial intelligence systems comes primarily from the lack of sufficient redundant alternatives. AI techniques are often used when more traditional algorithms are insufficient, and so redundant options for comparison are few and far between. That being said, the use of simpler machine learning models and a variety of traditional algorithms may provide enough partial redundancy individually to provide complete redundancy when combined. The importance of employing fault-tolerance techniques to fully autonomous vehicles that will likely be employing complex, potentially volatile AI methods such as machine learning algorithms cannot be overstated.

2.4 Fully Autonomous Vehicle Challenges

As was alluded to in the previous section, the transition to level 4 and eventually level 5 autonomous vehicles is fraught with difficulties and concerns. The growing levels of automation in vehicles that may have to deal with life-or-death situations is of paramount concern to automotive manufacturers and potential customers. A study is being conducted at the Massachusetts Institute of Technology (MIT) dubbed the *Moral Machine* [3] in which human participants act as an autonomous vehicle under a variety of different ethically challenging scenarios. Not only do we as engineers have to worry about the ethical dilemmas level 5 autonomous vehicles pose, but challenges surrounding driving competency under all conditions further increase the difficulty of realizing safe level 5 autonomy.

AI algorithms rely on enormous databases of training data, used to extract and incorporate human-like driving capabilities into machine learning models. These algorithms represent a large percentage of computational methods that can achieve such a high level of competency on complex tasks normally achievable by humans only. However, despite the effort expended in improving the accuracy and performance of these machine learning models, the final result is largely a black box. The opaque nature of machine learning models leads to difficulties in reasoning and rationale behind algorithm behaviour. To make matters worse, a growing number of these algorithms rely on online learning, as exemplified by the rise in use of reinforcement learning (RL). This adds a layer of volatility, in which the runtime execution of the vehicle affects the vehicle's future behaviour.

The black box nature of these mechanisms coupled with their volatility

to training data represents major challenges for the safety of autonomous vehicles. As such, a framework needs to be developed to mitigate the risks associated with artificial intelligence mechanisms, in an aim to enable safe operation of level 5 autonomous vehicles. The following work focuses on that goal by developing fault-tolerance and safety mechanisms that may be applied to autonomous vehicles that are dependent on AI algorithms and incorporating those techniques into a unified software architecture. While these are bold claims and the architecture will likely be altered by others in the future, we hope that the aggregation of existing fault tolerance techniques, safety techniques, and high-autonomy AI-based software architectures into a unified framework will provide a stepping stone from which level 5 autonomy can be achieved.

Chapter 3

Relevant Research

This chapter aims to review existing literature pertaining to the safe deployment and fault diagnosis of AI components within the context of autonomous vehicles. While this review focuses on techniques to integrate AI components safely into autonomous vehicles, these methods extend to broader applications in safety-critical autonomous systems. Although considerable effort has been made to survey field-specific techniques for safe autonomy, there has yet to be a survey that aggregates all of these techniques into a comprehensive review; an area this chapter hopes to fill.

Literature in several pertinent topics in artificial intelligence is presented under the headings of *safety concerns* and *existing mitigation strategies*. Following this, we discuss literature that presents several AI software architectures that incorporate these mitigation strategies. The chapter concludes with a section that presents literature on organic computing architectures and on autonomous vehicle architectures, refining additional safety strategies for safe autonomy in autonomous vehicles.

3.1 Search Method

Each of the above research fields is far too large to be fully reviewed in detail, so the scope of this literature review has been reduced substantially by focusing on the complex interactions between artificial intelligence and autonomous vehicles, in addition to drawing insights from the broader field of autonomous systems.

Several databases were identified as reliable sources of peer-reviewed papers in the fields of artificial intelligence, autonomous vehicles, and autonomous systems. These databases include *IEEE Xplore*, *Engineering Village*, *ScienceDirect*, *SpringerLink*, and the *Wiley Online Library*. Additionally, some of the sources were originally discovered through *Google Scholar*.

The search process was conducted by identifying a set of relevant queries in the problem domain of safe AI-enabled autonomous vehicles; *autonomous vehicles*, *automotive safety*, *artificial intelligence*, *autonomous systems*, *organic computing*, *fault tolerance* and *safety-critical*. It should be noted that several combinations of these keywords were used to find source material, and thus this list is non-exhaustive.

The source material was weighted in favour of newer publications due to the fact that there has been a surge in artificial intelligence and machine learning research in recent years, and we made the assumption that newer publications will be more relevant to our work. Additionally, the level of autonomy present in existing autonomous vehicles is fairly low, so newer research has been focused on increasing their autonomy. As such, newer architectures and techniques for autonomous vehicles were weighted more highly than older non-autonomous or low autonomy alternatives.

3.2 Supervised Machine Learning

Supervised machine learning involves learning a function mapping input training data to a set of labels from labelled training data; this function is known as a *policy* in the field of machine learning. While simpler than RL algorithms in the sense that they are trained from a static set of training data and tend to remain static during deployment, there are a number of issues that arise from their complexity. It should be noted, however, that these issues are not limited to supervised machine learning models, and the same issues are found in RL models as well.

3.2.1 Training Dataset

3.2.1.1 Safety Concerns

Supervised machine learning frequently makes use of training data in order to produce the desired policy. Constructing realistic systems, however, results in a finite training set, which poses several issues for safety-critical systems. Given a training set that does not accurately capture the real probability distribution of events, the resulting policy may not generalize in real-world conditions [43, 10]. Another issue, derived from *Back Swan Theory* [40], involves a training set devoid of rare events that often results in a system unable to cope with improbable, yet plausible hazardous conditions [43, 10, 24].

3.2.1.2 Mitigation Strategies

Few mitigation strategies exist for this class of concerns. Some work has been done on theoretical bounds for the number of examples needed to ensure good generalization, but the validity of the work has been put into question [8].

The issue of training data with an inaccurate probability distribution has been addressed in the field of *transfer learning*, and methods have been proposed in the context of autonomous vehicles [11], with the main method involving ensembles created from heterogeneous datasets. This issue is still an open area of research, and the results of current mitigation strategies are largely application-dependant.

3.2.2 Model Feature Selection

3.2.2.1 Safety Concerns

The formulation of machine learning algorithms requires that certain model features be selected in conjunction with their target task. The number of neurons and layers in a neural network are examples of such model features that directly affect machine learning performance. Poor selection of these features may result in machine learning algorithms that perform suboptimally regardless of the quality of their training data [10]. Additionally, the lines of code (LOC) in model feature configuration files increases in mature systems, which adds the potential for additional error if untracked [33].

3.2.2.2 Mitigation Strategies

There is no blanket solution to model selection and solutions vary considerably by the application in question. The impact of model features should be assessed by constructing several algorithms with varying model features, and evaluating them against the same training set, cross-validation set, and test set [10]. Several techniques for feature selection have been evaluated for their accuracy as the number of features increases [4]. It should be noted,

however, that too many features results in a reduction of prediction accuracy; also known as the *curse of dimensionality*. It has been suggested that feature entanglement may produce noise or irrelevant knowledge being extracted from patterns in large feature spaces [8]. It has also been suggested that modifications to the configuration of machine learning algorithms should involve peer-reviewed code analysis [33] in order to prevent failures due to haphazard parameter configuration.

3.2.3 Optimization Strategy

3.2.3.1 Safety Concerns

When training supervised machine learning algorithms the goal is to minimize risk in making an incorrect prediction. Two central concerns to this goal are overfitting and underfitting. A machine learning model is overfit when the resulting policy varies considerably around boundary examples and rare examples. This is primarily a result of excessive parameter selection when constructing the machine learning model or when the model is trained for too long. The second issue of underfitting is when the resulting policy does not accurately capture the large scale behaviour of the intended training data. This is results from the converse of the overfitting scenario, namely when too few parameters are selected for the model or when training data or training duration are insufficient. These issues result in a machine learning model that is incapable of capturing the true policy, rendering the model useless [10]. Another issue to consider is that many machine learning algorithms incorporate a loss function as a means of tempering the algorithm to the severity of false estimates during training. It is common practice to use the same loss function

for every example in the training set, but in a safety-critical context, not all failures have the same impact [43, 10].

3.2.3.2 Mitigation Strategies

Techniques to correct for overfitting have been developed and evaluated, with the bulk of research involved in determining effective regularization functions [23]. Some other techniques involving adding noise to hidden layers of neural networks, known as dropout, have been shown to provide promising regulating effects on overfitting [39]. One potential solution to broad loss functions is to modify loss functions to incorporate application-specific metrics measuring the impact or severity of an incorrect prediction [10]. It has also been suggested that multiple loss functions may be employed on each training example in order to further reduce the likelihood and/or severity of overfitting [10].

3.3 Reinforcement Learning

Reinforcement learning is a type of machine learning characterized by acting in a way that maximizes some notion of cumulative reward. Unlike supervised machine learning discussed in the previous section, RL typically utilizes runtime data as training data, allowing the algorithm to adapt to new circumstances in the environment and continual policy improvement. This area of machine learning has been gaining massive traction across multiple domains due to the flexibility and potential for higher performance as time progresses in the field. That being said, many of the issues plaguing supervised machine learning also affect RL algorithms, as well as a host of additional issues specific to RL algorithms. The issues exclusively targeting RL algorithms are the focus

of the following subsections.

3.3.1 Exploration

3.3.1.1 Safety Concerns

The main process by which RL algorithms attempt to maximize their cumulative reward is to explore new output domains for a given input domain. This process is also known as *exploration* in the field of machine learning and is how RL algorithms can adapt to new environments and continually improve. The key issue with this core component of RL is that the exploration process may result in classification or action being chosen that results in the system being put into a dangerous state [10].

3.3.1.2 Mitigation Strategies

Several methods have been suggested to alleviate the issue of potentially dangerous unexplored states, primarily involving incorporating external knowledge sources into the RL model. One such method involves deriving a baseline policy from a training set, as was described in the section on supervised learning [13]. Another similar technique with differing execution involves learning a baseline policy from a finite set of expert demonstrations [13, 2]; an inverse reinforcement learning (IRL) approach. IRL has successfully demonstrated capturing defensive driving behaviour for replication in autonomous vehicles [37, 9]. The above two techniques involve creating a baseline policy from external knowledge. They not only reduce the need for exploration but vastly restrict the exploration space required due to a near-optimal policy being used, rather than having to learn the entire policy during runtime. Two additional

techniques pursued in the literature, involve incorporating a teacher to help guide the exploration process, where the teacher can either be a human or an automated controller. Since this paper focuses on level 5 autonomy, we will be taking only the automated controller scenario into account. These techniques take two different approaches to guiding exploration, one where the teacher plays an active role and the other a more passive approach [10, 13]. The first technique involves the teacher monitoring the RL algorithm and the environment while providing advice or correction to the RL algorithm should it deem the performance or risk unacceptable. The second technique involves the RL model having a confidence parameter, which when passing a lower threshold prompts the RL model to ask the teacher for assistance. Still another technique involves predicting and simulating potential exploratory actions before attempting to catch dangerous situations before they occur [2]. These last three techniques all involve guiding the exploration process and make some use of confidence or risk in the selection of potential actions to be taken.

3.3.2 Reward Hacking

3.3.2.1 Safety Concerns

RL models attempt to maximize cumulative reward via a reward function, which acts as an attempt to formalize the intended behaviour of the system. However, alternative behaviours may exist that satisfy the reward function that goes against the designer's original intent, potentially leading to unsafe conditions [24, 2]. For example, say we have an autonomous vehicle utilizing RL with the goal of getting from one destination to another as fast as possible. This crude reward function would likely result in the algorithm learning that

fast reckless driving allows it to maximize the reward function. On the other extreme end, if we have a reward function detailing that not harming anyone is the primary objective we might end up with a vehicle that does not move due to fear of causing harm. Simply making more complex reward functions does not necessarily guarantee that there are not unintended behaviours that can maximize the reward function.

3.3.2.2 Mitigation Strategies

One potential solution to prevent reward hacking is to construct an adversarial reward function. This involves the reward function having access to a set of data labelled by humans, and should the RL agent suggest a high reward action, the reward function could compare the action and the expected reward to the labelled data to determine whether it is valid or not. A more interesting variation of this concept suggests training multiple agents with different objectives and having them validate each other [10, 2]. Similarly, it has been suggested that combining multiple reward functions may also produce a result that is more resilient to reward hacking [10, 2]. Yet another suggestion involves model lookahead, in which the agent is at least partially rewarded for predicted beneficial future states [2]. This coincides with the safe exploration technique discussed in the previous section on simulating potential future states before selecting a final action.

3.4 Artificial Intelligence Architectures

In order to take full advantage of the previously discussed mitigation strategies for AI systems and incorporating them into autonomous vehicles, we now

review several software architectures that utilize these techniques. The architectures are drawn from more recent safe artificial intelligence and organic computing research, as well as more traditional automotive architectures so that their integration to autonomous vehicles is as seamless as possible. We briefly explain the architecture decompositions, as well as their beneficial and detrimental aspects. We then strive to take the best aspects of each when constructing the proposed architecture in Chapter 4.

3.4.1 Arguing Machines Framework

A traditional technique for increasing the reliability and safety of various systems is to incorporate some form of redundancy in order to mitigate single component failures. The arguing machines framework [12] applies this technique to artificial intelligence systems.

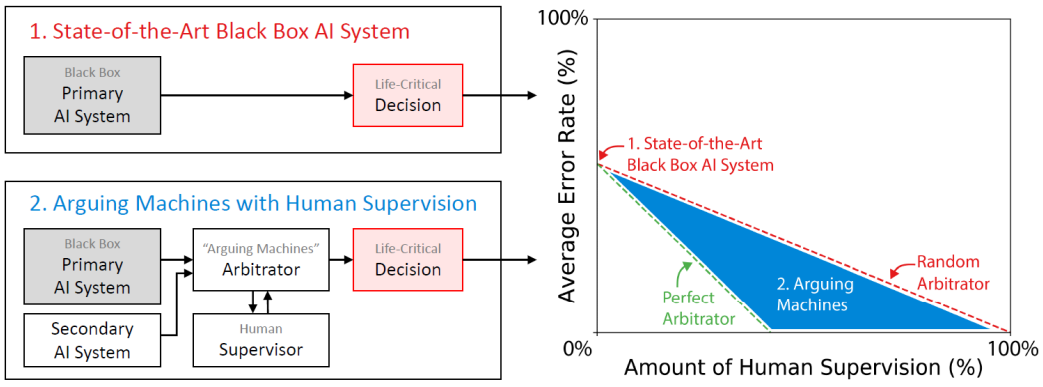


Figure 3.1: Arguing Machines Framework Diagram [12]

This architecture incorporates two AI systems with their outputs connected to an arbitrator responsible for detecting a conflict between the two artificial intelligence systems. The arbitrator attempts to resolve the conflicting AI systems through internal processing and if unsuccessful requests that a human

operator resolves the discrepancy, which then results in an action decision.

Commentary

The downside of this architecture is that it requires human intervention to resolve the conflicting artificial intelligence systems, while we are focused on level 5 autonomy that cannot rely on a human to provide aid. Another downside is that the architecture suggests an end-end approach for the utilization of machine learning. As discussed in [32], the use of end-to-end machine learning vastly increases system complexity and reduces comprehension and traceability, and as such, end-to-end systems are discouraged in order to abide by the ISO 26262 standard and general best practices in software and safety engineering. The benefit, however, is that we now have the potential for online cross-validation of the artificial intelligence systems. This is similar to our discussion on reward hacking mitigation strategies that involve using multiple agents with different reward functions. Multiple artificial intelligence systems are considered in Chapter 4 when constructing the Sentinel architecture.

3.4.2 Runtime Assurance Framework

Another similar redundancy technique used in safety-critical systems is one that involves a typically higher performing, albeit less safe controller in conjunction with a lower-performing safer controller. The runtime assurance framework [6] proposes utilizing this design to potentially mitigate the non-deterministic nature of artificial intelligence and machine learning models.

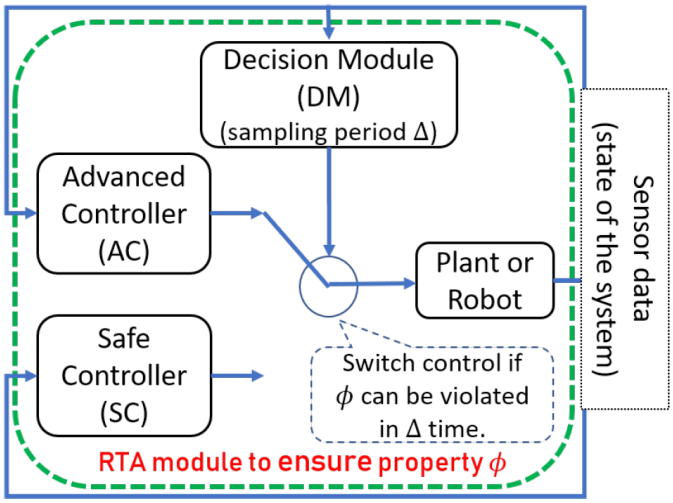


Figure 3.2: Runtime Assurance Framework Diagram [6]

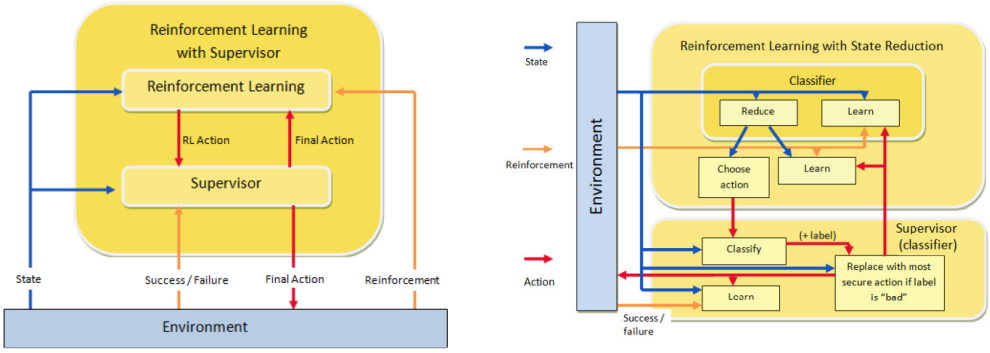
The framework discusses having an advanced controller that could in principle be a machine learning model, alongside a less capable but formally verified safe controller. In addition to the two controllers, there is a decision module that processes sensor data to make a decision as to which controller should be used for a given cycle. The general idea behind the operation is to primarily run the advanced controller output. However, when the decision module detects sensor data that indicates a safety violation has occurred, or that the system is moving far enough away from the safe controllable region, it then switches control to the safe controller for a time in order to move the system to a more predictable, safe state. This way the framework can make some claims about the ability of the system to recover to a safe state within a given time duration. The framework has been targeted towards the field of robotics, with an implementation on a quadrotor aerial robot.

Commentary

While it has yet to be applied to autonomous vehicles or multi-agent scenarios, the core ideas of having a safety monitor and a backup safe controller to mitigate the ill behaviour of riskier controllers seem to be well aligned with the goal of safety in autonomous vehicles.

3.4.3 Adaptive Supervisor Architecture

A technique discussed in the RL section involved utilizing a teacher framework in order to guide the exploration and learning process of RL algorithms. The adaptive supervisor architecture [22] is an example that utilizes this technique and has been shown to increase the speed and safety of learning optimal policies.



(a) High Level (b) Low Level

Figure 3.3: Adaptive Supervisor Architecture Diagrams [22]

In their example, the authors utilize a supervised learning classifier as the supervisor model, which acts as a guide for the untrained RL algorithm. As was discussed in previous sections, supervised learning models remain static during deployment, while RL models are in constant flux as they need to explore their environment in order to produce a policy; by themselves, there is

too much risk. The supervised learning classifier would be too brittle to handle all possible events during deployment because it does not have the capacity to learn from online experiences. On the other hand, the RL algorithm would have to blindly explore the environment in order to learn and may result in dangerous states being explored. This architecture, however, plays to each method's strengths to produce a potentially more reliable and safer system.

Commentary

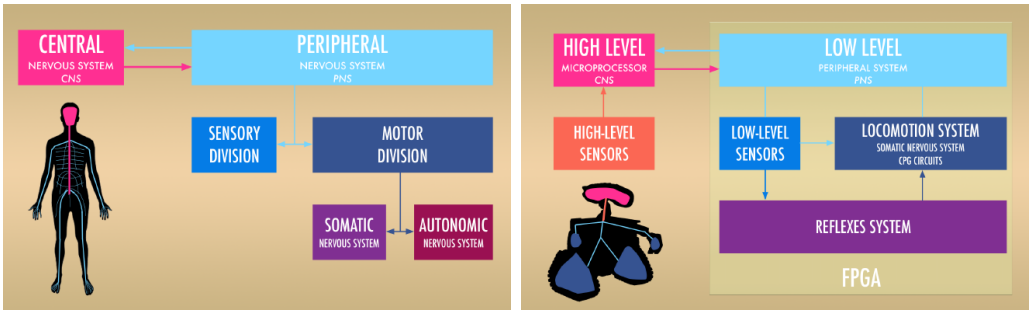
By using the supervised learning model as a teacher to guide safe action, the exploration space of the RL algorithm can be reduced significantly, while still allowing the system the flexibility to learn over time. This is an example of utilizing heterogeneous artificial intelligence models in conjunction to increase reliability, safety, and model accuracy. The main takeaway of utilizing several heterogeneous artificial intelligence models to bolster the performance and safety of the system as a whole will be taken into account when constructing the Sentinel architecture.

3.5 Organic Computing Architectures

A forerunner in the development of autonomous systems lies in the field of organic computing, derived from the autonomic computing vision of IBM in 2001. This field seeks to give computing systems capabilities normally only seen in biological systems, such as self-repair, self-reconfiguration, self-optimization, and self-protection [28].

3.5.1 Hybrid Bio-inspired Control Architecture

Biomimicry often yields interesting solutions to difficult technological issues. The hybrid bio-inspired control architecture [5] suggests remodelling more traditional robotic control architectures to reflect the organization of the human nervous system.



(a) Human Nervous System (b) Proposed Architecture

Figure 3.4: Hybrid Bio-inspired Control Architecture Diagrams [5]

This architecture was applied to several robotic platforms, yielding similar behaviour to their biological counterparts. One such platform involved a robotic eye that would reflexively blink in order to protect the sensor, similar to that of humans and other animals.

Commentary

While this experiment is interesting, the work does not investigate broader use cases of these reflexes and features of the autonomic nervous system. The autonomic nervous system is responsible for the fight-or-flight response in animals, while reflex arcs are responsible for reflexes. Both of these protective mechanisms are used by animals to handle hazardous conditions and may be similarly applied to handle hazardous conditions of autonomous vehicles. The concepts of elevated alertness or caution via an artificial fight-or-flight response

and short latency reflexes for critical situations will be considered in developing Sentinel.

3.5.2 Multi-Layer Observer/Controller Architecture

The organic computing paradigm attempts to bestow machines with the adaptability of biological organisms. Organic computing research has lead to a promising architecture [28] that not only covers several layers of adaptability but incorporates mechanisms for collaboration and goal modification.

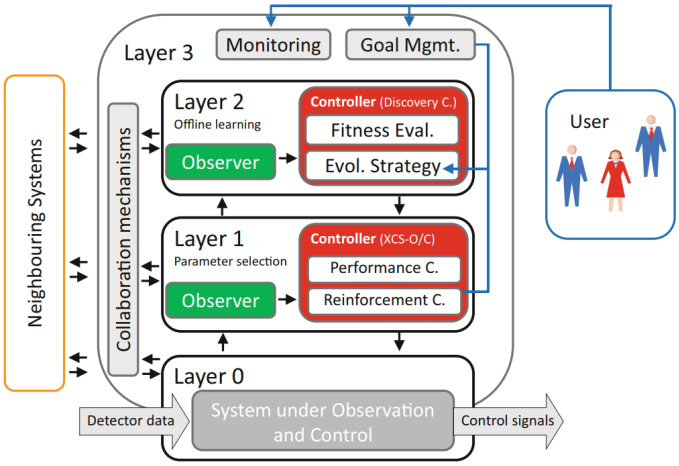


Figure 3.5: Multi-layer Observer/Controller Architecture Diagram [28]

The architecture makes use of a three layer hierarchy of decision-making, monitoring and goal management, as well as a collaboration mechanism for interfacing to adjacent systems. While four layers can be seen in Figure 3.5, layer 0 does not involve any computing. Layer 1 consists of an observer that constructs an environment model via sensor data, and a controller that incorporates a rule-based action classifier that is only adjusted via a layer 2 overwrite. Layer 2 observes layer 1 for anomalies and makes use of a simulator to predict more appropriate classifiers than the one currently operating in layer

1, in this case via a genetic algorithm. Layer 3 is concerned with high-level system monitoring and goal modification of lower layers, as well as providing a channel by which the system can incorporate external system data into its own observational models.

Commentary

The high-level goal modification and system monitoring align well with the need to alter autonomous vehicle destinations and for dispatchers or passengers to be able to monitor the vehicle's status. Adjacent system interfacing could provide vehicle-to-everything (V2X) behaviour and the use of multi-layer monitoring could allow for safer adaptive control techniques that are produced by AI systems. The key concepts for consideration in this design are simulation before modification, multi-layer monitoring, goal modification, and adjacent system interfacing.

3.6 Autonomous Vehicle Architectures

Several autonomous vehicle architectures have been put forth to solve the issues surrounding the task complexity and the goal of increased autonomy in autonomous vehicles. These architectures will be reviewed as done in previous sections, but it should be noted that their relevancy to the proposed architecture is higher than those previously discussed. This is done so that adoption and incorporation of the resulting architecture into autonomous vehicles has as low a barrier to entry as possible.

3.6.1 Cognitive ADAS Architecture

An architecture that aims to increase the autonomy of autonomous vehicles has been put forth with several variants ranging from level 3 to level 4 autonomy [7]. The goal of these architecture variants is to provide a framework and roadmap by which automotive manufacturers can safely increase the autonomy of autonomous vehicles over the coming years. We will focus on the proposed level 4 autonomy architecture, as that is closer to our goal of level 5 autonomy than the alternatives.

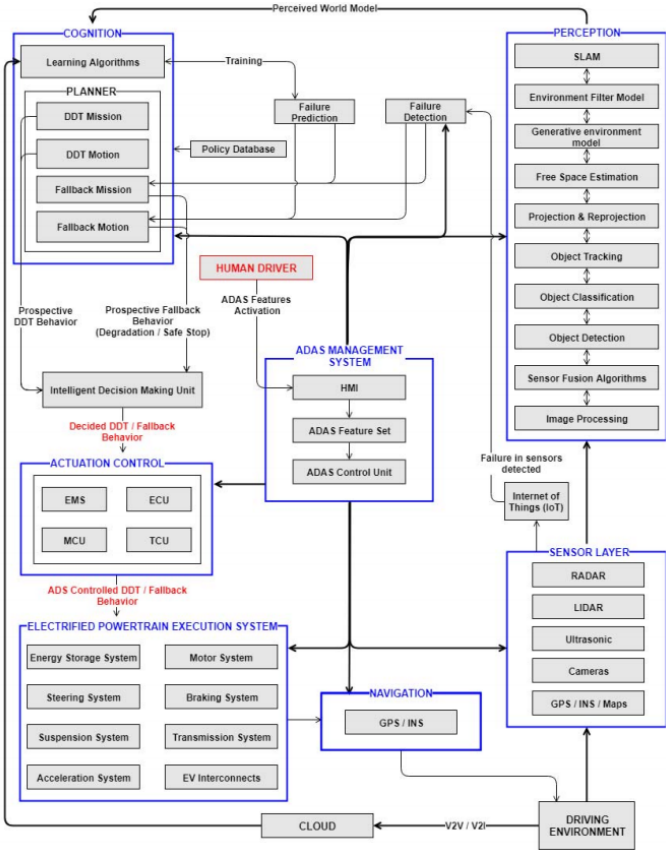


Figure 3.6: Cognitive ADAS Architecture Diagram [7]

The architecture features a traditional decomposition involving sensors, perception, planning, control, and actuators. Some of these blocks have been

further decomposed for their target of electric vehicles, but the overall data flow remains the same. A major distinction between this architecture and many similar ones is the centralized advanced driver assistance system (ADAS) management system that coordinates parameter modification and feature selection of all other modules. Two additional features of this architecture are the ability to modify behaviour according to vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) data, as well as a multimodal planner. Meanwhile, the planner not only features mission modules to define the current goal decomposition but also features fallback mission and motion modules to handle dangerous situations or system failures.

Commentary

It is unclear why passengers would be given control over which autonomous features should be active or inactive but may be rationalized by this architecture targeting level 4 autonomy where the driver can still assume control over the vehicle when in an operating environment outside the capability of the level 4 features. Utilizing V2V and V2I data allows the vehicle to expand upon its own perception of the environment and provides a mechanism to communicate more nuanced behaviour between other vehicles. One major concern arises when it is assumed that training data can be used to determine system failure of the learning algorithms in the cognition block. As was discussed at length in the previous sections on artificial intelligence safety, training data can misrepresent the true nature of reality and so is unreliable as a sole diagnostic. Additionally, RL algorithms may be used in order to glean knowledge from an ever-changing environment, which would result in initial annotated training data being an increasingly small fraction of total training data, resulting in

a worst-case scenario of no diagnostic benefit from the initial training data. From this, we conclude that key aspects for consideration are the fallback and mission module decomposition of the planning module, and V2V/V2I capabilities, alongside the perceive, plan, act framework seen here and in several other architectures.

3.6.2 Autonomous Vehicle Control Architecture

Another autonomous vehicle architecture has been proposed that shares several features with the runtime assurance framework [6], and promoted a separation of safety and control [27]. The proposed architecture utilizes a traditional sense, plan, act system, as well as a dedicated safety subsystem that attempts to avoid and/or correct for unsafe scenarios.

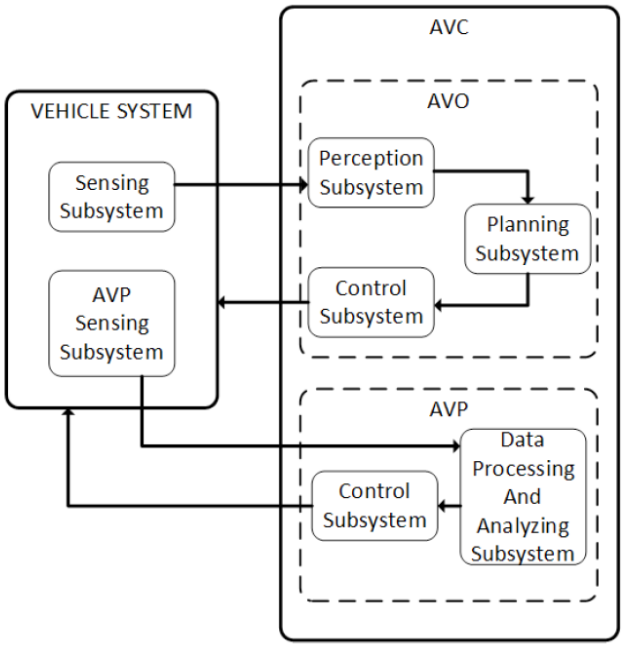


Figure 3.7: Autonomous Vehicle Control Architecture Diagram [27]

The architecture features two subsystems; Namely "autonomous vehicle operation" (AVO) and "autonomous vehicle protection" (AVP). The AVO module represents the traditional sense, plan, act framework seen in existing vehicles and robotics solutions, while the AVP module represents a safety monitor that evaluates vehicle behaviour in order to adjust current control to return the system to a safe state.

Commentary

Superficially similar to the runtime assurance framework discussed earlier, this architecture makes the distinction of having a dedicated and separate sensor suite for the AVP module. This additional separation of normal control from safety reduces the risk of cross-channel errors affecting both control and safety subsystems. It has also been suggested that safety systems utilize as few artificial intelligence mechanisms in their diagnosis as possible for determinism and clarity in their evaluation of the environment [32]. While it is true that AI techniques cannot accurately detect all faults, they have become exceptionally good at detecting mechanical and electrical faults, such as motor failure. From this, we can conclude that control and safety subsystems should be kept as separate components to avoid undesirable emergent behaviour from their interactions, as well as minimal use of artificial intelligence techniques when attempting to detect more complex system faults.

3.6.3 Autonomous Vehicle Decomposition Framework

While not an architecture that directly shows interconnections and data flow between modules, it is important to understand the necessary functional de-

composition of autonomous vehicles, which is provided in a proposed functional architecture [34].

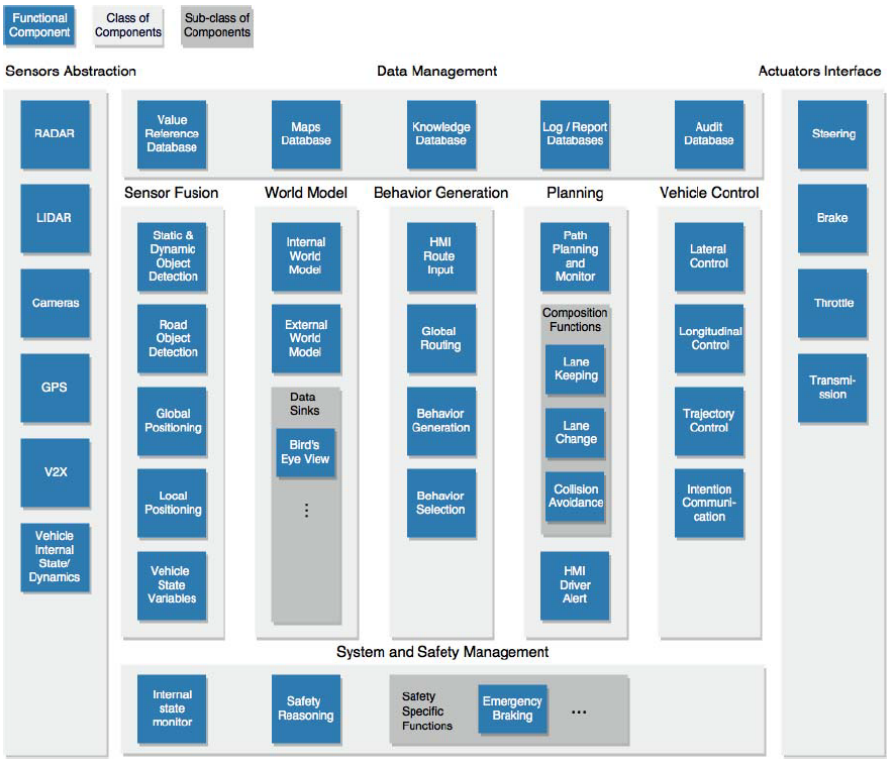


Figure 3.8: Autonomous Vehicle Functional Decomposition Diagram [34]

This architecture captures many modules that have already been discussed in previous architectures and frameworks, with a few notable exceptions that should be discussed. One such decomposition involves the world model, where it has been decomposed into internal and external world views as well as data sinks giving different world model viewpoints (ex: bird's eye view). Behaviour generation is partially modelled in [7], and in both papers is intended as the highest cognitive level of decision-making, involving broad goal-based planning which is further decomposed in the following modules. An human machine interface (HMI) driver alert module is presented for vehicles level 4 and below to prompt the driver for assistance. A largely neglected component discussed

is the data management component which holds several modules for knowledge representation as well as data logging.

Commentary

The separation of world model viewpoints may allow for more nuanced decision-making, especially from a safety standpoint where internal faults versus external hazards may need to be treated differently. While our work is focused on level 5 autonomy where a human is not needed for any functionality, an HMI driver alert module may still be useful in notifying the driver of unrecoverable vehicle faults that may impair the vehicle's ability to maintain passenger safety. Due to the volatility of artificial intelligence systems, it may be important to use the data management module to log input/action pairs, self-modification, and any other complex decision-making processes associated with the artificial intelligence components in question. This can not only be useful for diagnosing accidents that may be at the fault of artificial intelligence, but also important data, showcasing how online learning mechanisms evolve in the real world when compared to simulated models. Additionally, this data storage could be used to store snapshots of RL model parameters to allow for a rollback feature should the RL model begin performing poorly. It should be noted, however, that this architecture places V2X and HMI functionality in the sensor abstraction and planning abstraction, respectively. It can be argued that bidirectional functionality is required from both of these modules, and so it may be prudent to distinguish these in an interface layer as done in [28].

3.6.4 Safe-AV Architecture

Currently, the majority of autonomous vehicle architectures have been aimed at level 3 and level 4 autonomy, while our own aims to achieve level 5 autonomy. One architecture had been identified as targeting level 5 autonomy, namely the Safe-AV architecture [35].

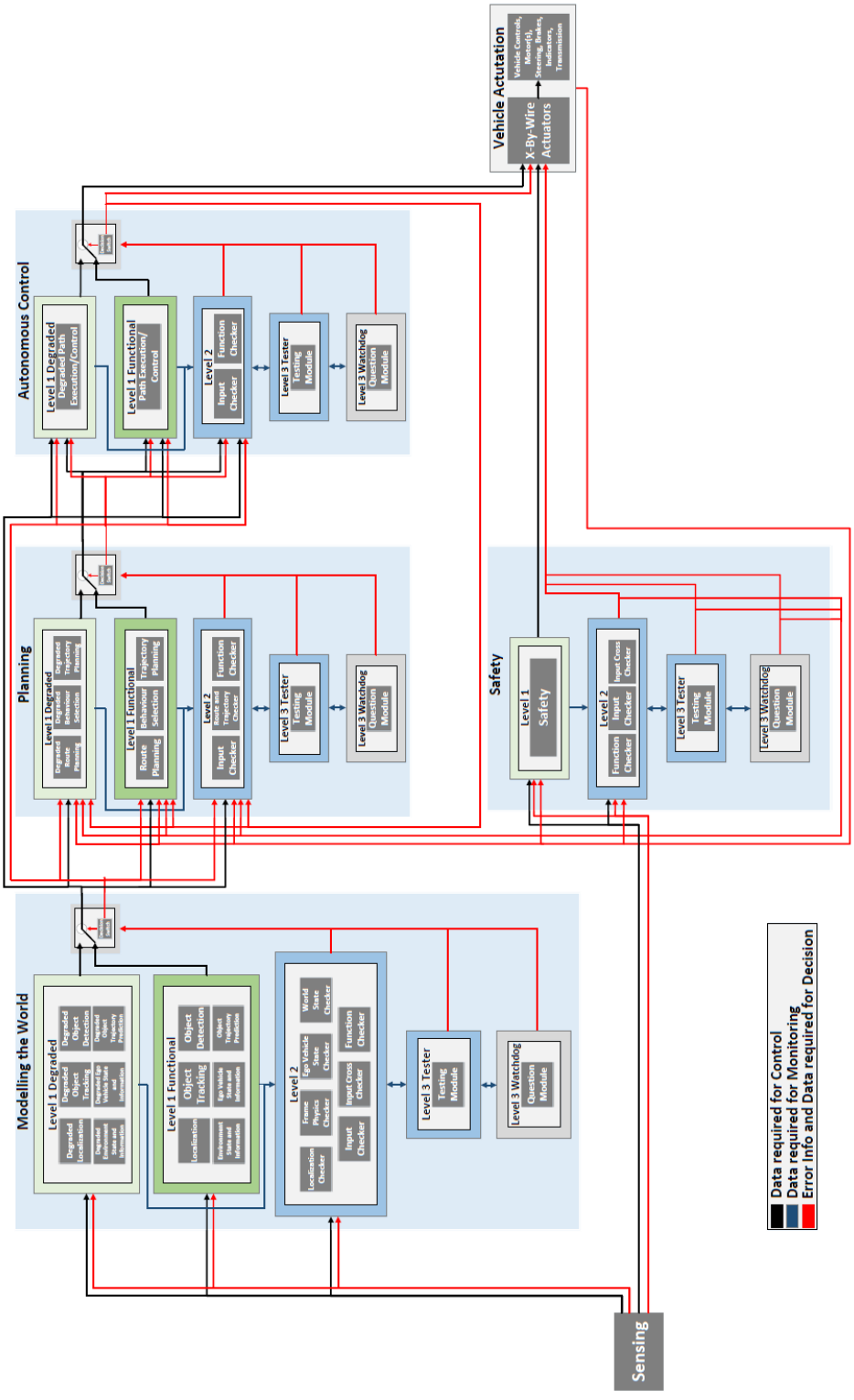


Figure 3.9: Safe-AV Architecture Diagram [35]

As seen in previously discussed architectures, Safe-AV makes use of the traditional sense-plan-act framework and incorporates a safety system operating in parallel, thereby providing a means for safe operation should the main pathway fail. Several features distinguishing Safe-AV from other architectures are apparent, the first of which incorporates E-Gas monitoring concepts. E-Gas monitoring [46] is a framework developed and adopted by several German automotive original equipment manufacturers (OEMs) to improve vehicle safety and reliability. One of the key concepts developed through E-Gas and incorporated into Safe-AV is the three level monitoring framework. Level 1 is composed of software modules responsible for the functional operation of the vehicle. Level 2 is composed of software modules responsible for testing whether level 1 modules are operating correctly, normally via validation of level one outputs against a set of independently computed output bounds. Level 3 modules are responsible for determining whether level 2 modules are operating correctly, normally via a question/answer framework. Periodically level 3 will prompt level 2 with a question to test whether level 2 modules are operating correctly, and level 3 is separated on another physical platform to isolate single-point failure of the primary hardware. This use of redundancy and online validation of functional components is used in Safe-AV for fault detection of software modules in each system in the sense-plan-act chain, and provides a mechanism by which the architecture can switch to a degraded performance mode in level 1 should the normal operation be deemed faulty. Lastly, Safe-AV makes the distinction that physical information channels should be separated based on the criticality of the data transmitted. As such, critical operational and monitoring data is placed on one transmission medium and non-critical data on another.

Commentary

As was alluded to previously, Safe-AV makes use of degraded performance modes in each of the sense-plan-act chain. It is intended that these degraded performance modes should collectively be able to return the system to a safe state should the normal performance modes be deemed faulty. Safe-AV claims that conventional software and/or more simple machine learning software be utilized in the degraded performance modes. This, however, does not account for many of the issues intrinsic to AI discussed in previous sections, specifically regarding the difficulty in detecting faulty AI and the lack of available alternatives to supplant their functionality. It is also suggested that for object detection specifically, that simpler machine learning algorithms can be trained to detect whether an object is dangerous or not rather than specifically what type of object is observed (pedestrians, vehicles, animals, etc). This oversimplification of AI capabilities faces a major issue of over-extrapolation, where the machine learning algorithm in question will likely be unable to converge to a policy on "dangerous or not" due to the ambiguity of the classification problem, as well as all the normal issues of insufficient training data. It should be noted, however, that the concept of utilizing software modules exhibiting degraded performance and complexity when compared to higher performance equivalents still has value. Rather than take an all or nothing approach, these degraded performance modules could instead be utilized to validate more complex AI modules in parallel. For example, an object detection module utilizing machine learning may suggest that there is a 60% chance of a pedestrian being present in front of the vehicle and a conventional algorithm may utilize radar data to determine that there is an obstacle ahead of the vehicle. The com-

bination of this information provides a better means of determining whether there is an obstacle and/or a pedestrian in front of the vehicle. This utilization of degraded conventional algorithms in conjunction with AI components coincides with previously discussed designs utilizing additional AI components for cross-validation.

While the additional data transmission pathway is an excellent step in ensuring data throughput for mission-critical operation, it could also be suggested that further increasing the number of physical channels for redundancy and correction of erroneous transmissions may be needed. Vehicles already have a vast number of electrical and electro-mechanical components producing interference, and the issue is only being further exacerbated by the increasing number of wireless communication signals in the surrounding environment. The use of redundant physical channels with voting mechanisms and/or timing redundancy via retransmission of signals may be important to consider.

3.7 Hazard Analysis

In previous sections of this chapter, we discussed the unique properties and concerns associated with several AI techniques as well as several architectures we draw inspiration from that offer mitigation strategies to these concerns. Before moving into the design of our architecture it would also be prudent to review any existing hazard analyses for high autonomy autonomous vehicles, which will aid in the construction of the finalized architecture. An extensive systems theoretic process analysis (STPA) hazard analysis of high autonomy (SAE J3016^{TM} level 4/5) autonomous vehicles was conducted by Shah [35] and will form the basis of this discussion.

In this hazard analysis, several undesired control actions were identified and iteratively refined into a finalized subset. From here a set of causal factors for each undesired control action were identified, followed by a set of safety requirements that mitigate those causal factors. We will compare these safety requirements to the information contained in previous sections of this chapter to identify missing mitigation strategies that may provide additional benefits to our architecture. The following table contains a copy of the safety requirements of the aforementioned hazard analysis and corresponding architectures from previous sections that satisfy the requirements.

ID	Safety Requirement extracted from [35]	Supporting Architectures
SR-1	Modelling the world (perception) must cross check/validate outgoing environment detections (e.g. if detected lane markers match a road map)	3.4.1, 3.4.3, 3.5.2, 3.6.4
SR-2	Modelling the world (perception) must cross check/validate incoming data (e.g. if a radar detection matches a lidar detection)	3.5.2, 3.6.1, 3.6.4
SR-3	Modelling the world (perception) must report its confidence on each applicable output	3.4.1, 3.5.2, 3.4.3, 3.6.4
SR-4	Modelling the world (perception) must monitor its internal functions and report failures, faults or errors	3.4.1, 3.6.4

SR-5	Where feasible, Sensing must use redundant (both homogeneous and heterogeneous) sensors	3.6.4
SR-6	PAC (planning and control) must validate incoming data (e.g. if reported ego vehicle speed shows an impossible change)	3.5.2, 3.6.4
SR-7	PAC (planning and control) must monitor its internal functions and report failures, faults or errors	3.4.1, 3.6.4
SR-8	PAC (planning and control) must take into account fault info and the confidence of the incoming data when performing its function	3.4.1, 3.5.2, 3.6.4
SR-9	PAC (planning and control) must validate outgoing values (e.g. if the braking actuation command is within limits)	3.5.2, 3.6.4
SR-10	PAC (planning and control) and MTW (perception) must provide, where feasible, degraded functionality in the presence of critical failures	3.4.2, 3.4.3, 3.6.4
SR-11	A backup safety mechanism that makes use of redundant sensors and processing must exist. This system must bring the vehicle to a safe state if the main control loop fails to keep the vehicle safe	3.4.2, 3.6.4

SR-12	Modelling the world (perception) and Sensing must perform relevant sensor diagnostics checks (e.g. calibration check, position check, coverage check, cross checks) at startup and TBD intervals during operation	3.6.4
SR-13	There must be redundant communication lines for critical data	3.6.4
SR-14	Communication lines must be protected from noise (high Signal-to-noise Ratio (SNR))	N/A
SR-15	Communication protocols that support a TBD bandwidth and a TBD Baud rate must be used. The protocols shall support fault detection (e.g. Cyclic Redundancy Check (CRC), Message Counter (MC) check), fault tolerance and recovery, arbitration, error reporting	N/A
SR-16	Critical data must use separate communication paths from non critical data (e.g. multimedia data is separated from braking signal value)	3.6.4
SR-17	Diagnostic checks (memory checks, CPU checks, etc.) must be performed at startup and at TBD intervals or based on TBD conditions	3.6.4

SR-18	Modelling the world (perception) must cross check/validate ego vehicle properties (e.g. if ego vehicle speed is within limits)	3.5.2, 3.6.4
SR-19	XBW system must validate its output commands (e.g. if the actuation signals are within limits)	N/A
SR-20	XBW system must monitor its functions and health and report faults to PAC	N/A
SR-21	XBW system must validate incoming data (e.g. if the actuation requests from PAC are within limits)	N/A
SR-22	A scheduling algorithm that protects against priority inversion and provides a timing guarantee for critical tasks must be used	N/A
SR-23	System software must not have more than TBD utilization	N/A
SR-24	A High ASIL Real-time Operating System (RTOS) must be used	N/A
SR-25	PAC (planning and control) must delay an acceleration commands by TBD seconds after an object in its path disappears suddenly	N/A

SR-26	Where feasible, MTW (perception) and PAC (planning and control) must make use of multiple sources of information when performing their functionality (e.g. MTW, to detect a lane, uses both a camera and a high definition map)	3.5.2, 3.6.4
SR-27	System must have sufficient hardware backups to remain operational in the presence of hardware failures	3.5.1, 3.6.4
SR-28	PAC (planning and control), MTW (perception) and their internals must be sufficiently isolated, both in software and hardware to prevent single points of failure and common failure modes	3.6.4
SR-29	Sensing must report its failures, faults or errors	3.6.4
SR-30	PAC must command the vehicle to start slowing down if an object in its path remains unidentified	N/A
SR-31	MTW must take into account error/failure information from Sensing when performing its functionality (e.g. if Sensing reports that a Camera's position has changed, MTW has less confidence on its function)	3.5.2, 3.6.4

Table 3.1: Hazard Analysis Safety Requirements and Supporting Architectures

It should be noted that the safety requirements listed with **N/A** as supporting architectures are requirements placed on the specific implementation of the autonomous vehicle architecture. Due to the Safe-AV [35] architecture deriving its design from this hazard analysis, it can be seen that it is listed as a supported architecture for all of the safety requirements. However, several safety requirements may not be fully met for level 5 behaviour. For example, SR-4 is likely to remain unmet under Safe-AV for both supervised learning and RL algorithms. This is because the multi-stage monitoring in Safe-AV can only test level 1 modules with pre-known input/output pairs. But as we've seen with machine learning algorithms, online testing via curated test data may not be able to determine whether the component is failing or if the original training data was insufficient.

Some of the requirements that have yet to be discussed in the previous architecture-specific sections involve the need to perform periodic checks (ex: sensor diagnostic, messages, memory, and CPU), sufficient hardware backups, and redundant communication lines for critical and non-critical data. Each of these topics emphasizes hardware in addition to software. This work focuses primarily on the software aspects of level 5 autonomy, and so will consider these when possible while acknowledging that additional hardware design and implementation-specific needs are outside the scope of this work. Interestingly, the need for sufficient hardware backups may also imply sufficient software backups, as seen in n-version programming. This topic could be of great

importance during the development of our architecture, as a fault software component such as a safety subsystem may result in the vehicle being placed into a more hazardous condition.

3.8 Summary

Although the fields of autonomous systems, autonomous vehicles, and artificial intelligence are rife with safety concerns, many mitigation strategies exist with several architectures demonstrating an improvement in safe behaviour. The safety techniques and architectures discussed in the reviewed literature will be utilized in the proposed architecture in the following chapter.

Chapter 4

Sentinel

This chapter presents a breakdown of the Sentinel architecture for level 5 autonomous vehicles. Sentinel makes use of several safety-oriented architectures from a wide range of domains related to autonomous vehicles, with the specific focus on improving safety and reliability of vehicles that rely on AI components, when developing level 5 autonomy. These mechanisms, discussed in Chapter 3, have been combined to create a unified architecture. Guiding this design process, general best software practices and fault-tolerance practices have been exercised. Fail-safe operation, graceful degradation, separation of safety and control, AI redundancy, distributed sensory and planning redundancy, probabilistic modelling, and proactive safety are some of the features Sentinel exhibits. These features and the general structure of Sentinel will be provided in the following sections. The architecture synthesis will be provided at the end of this chapter to explain the rationale for finalized features of the architecture and explores the iterative design process used to design Sentinel. An assurance case analysis of Sentinel is then provided in Chapter 5 to delineate the assumptions made and the evidence for SAE J3016 compliance of the

Sentinel architecture.

4.1 Design Methodology

Several promising features of existing related architectures were identified in Chapter 3, which were to be combined in some manner. Due to the number of reviewed architectures and the potential for negative properties spawning from combining several architectures at once, an iterative design process was chosen. Upon each iteration, one reference architecture was incorporated into the current Sentinel version. This process involved determining similar structures in either design, the beneficial structures of the reference architecture discussed in the corresponding section in Chapter 3, and merging these structures into a new Sentinel version. This process was chosen to limit the potential side effects of the additions to one new reference architecture at a time. Upon completing this merge, an analysis of the additions or modifications to Sentinel was performed on the remaining unaltered components to determine if new undesirable effects may have been unintentionally introduced. Identified undesirable effects and deficiencies were then compared against remaining reference architectures to determine if the concerns could be mitigated in a future design iteration. If no solutions from reference architectures were discovered, general fault-tolerant engineering practices were attempted to rectify the issue. If both of these steps failed to mitigate deficiencies, the modifications were removed from Sentinel. This iterative process was repeated until all reference architectures had been assimilated, culminating in the proposed final version of the Sentinel architecture.

4.2 Architectural Overview

The Sentinel architecture consists of twelve software modules at the highest level of abstraction, seen in Figure 4.2. These modules and their layout are a direct result of the aforementioned design process and shaped by the beneficial structures present in the related architectures discussed in Chapter 3. These twelve modules can be grouped into three categories; Operational, Support, and Safeguard. Operational modules are responsible for the normal operation of the vehicle and consist of the sensors, perception, planning, plan execution, actuator interface and actuators modules. Support modules provide several services to the operational modules to aid in vehicle safety and performance but are not strictly required for constant vehicle operation. Modules in this category are the agent interface, data management, simulation, and proactive safety modules. Safeguard modules provide a safe control bypass from sensors to actuators, thereby allowing the vehicle to safely react to unanticipated hazardous conditions. The reactive safety module and the control mode selector constitute the safeguard category. Each of these modules is discussed in detail in upcoming sections, and the remainder of this section discusses the overall operation and features of the architecture. It should be noted that the rationale for the composition of the architecture as a whole and of individual modules is spread throughout different layers of abstraction, the majority of which are described in the revision history section.

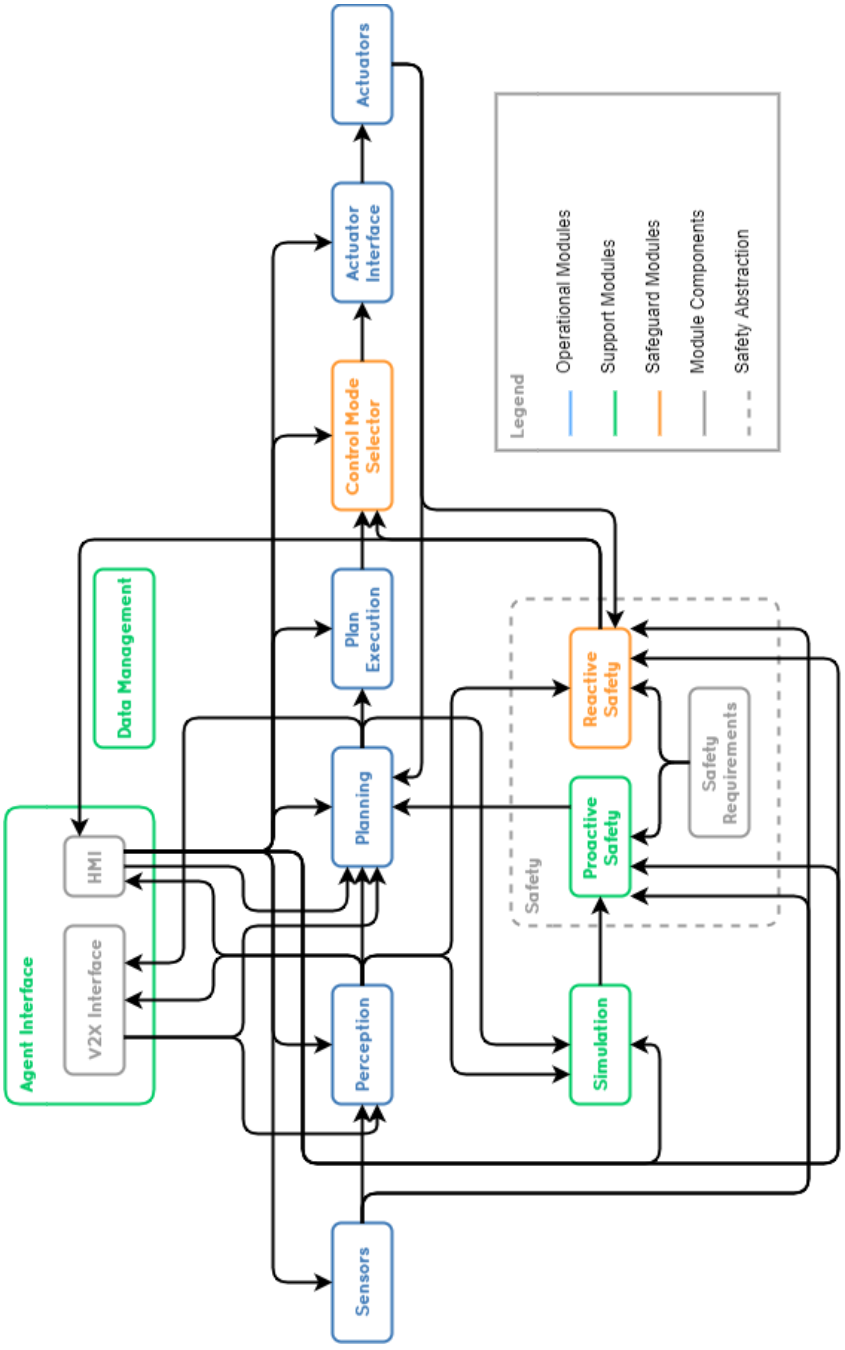


Figure 4.1: Sentinel Abstraction Level 1

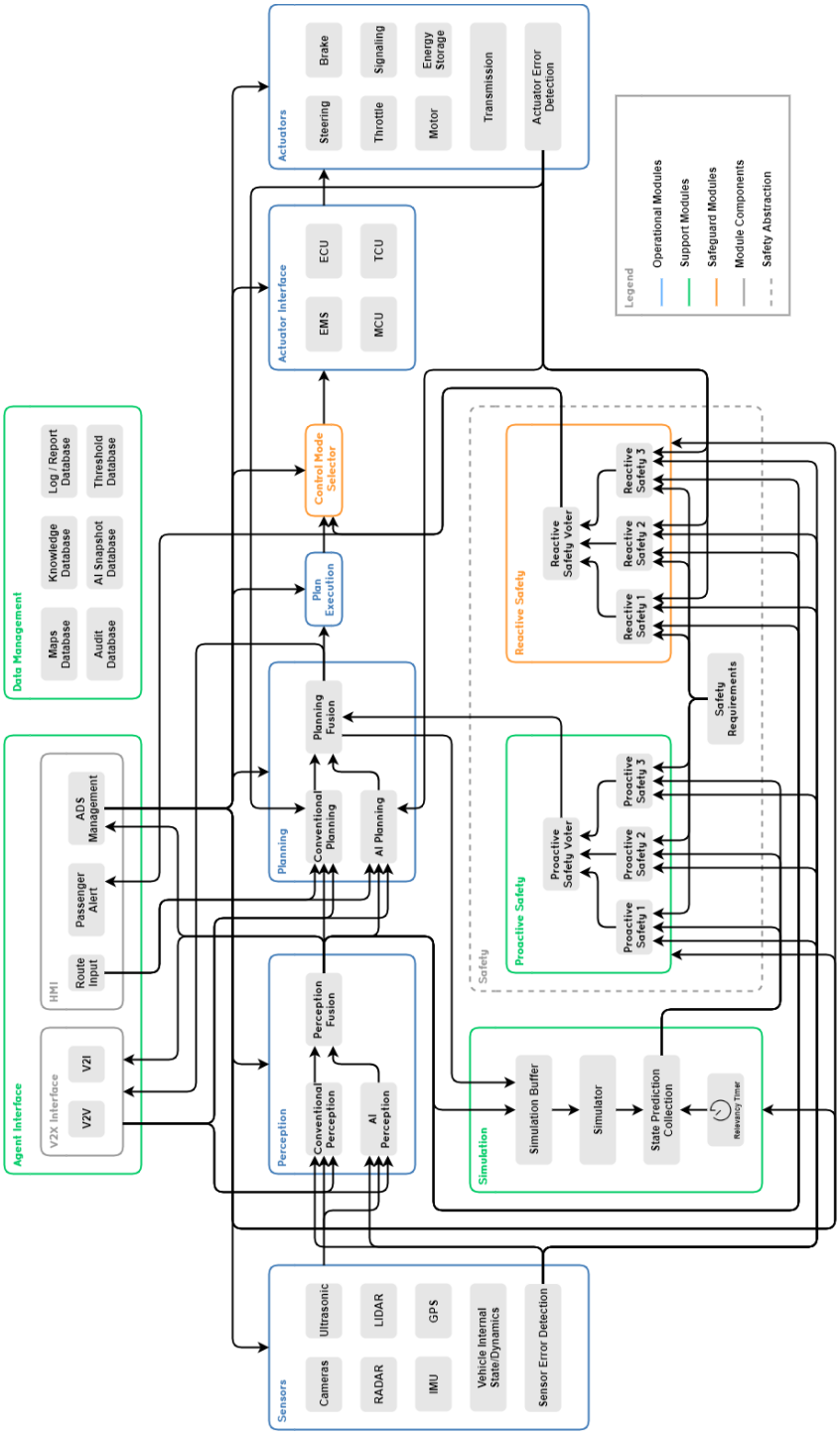


Figure 4.2: Sentinel Abstraction Level 2

4.2.1 Operational Modules

Modules in this category perform duties that enable normal vehicle operation and are modelled after the common sense-plan-act chain seen in several related architectures. Sensory data is collected via sensors such as camera feeds, radar, and lidar, which is then sent to the perception module. The perception module is tasked with creating a model of reality, composed of a model of the external environment and a model of the ego vehicle state. The perception module readily receives data from the sensory module but may also receive sensory data through external channels (V2X) to improve the accuracy and reliability of generated reality models. This information is then sent to the planning module, which is responsible for taking high-level goals and a reality model to produce driving routes and other planning actions not solely related to vehicle movement. This module also receives aid from several support modules (agent interface, simulation, safety) to improve generated plans, but as with the perception module, are not strictly necessary for normal planning operation. Once a plan has been finalized, this information is then sent to the path execution module, which is responsible for converting a high-level action plan into corresponding actuation commands. These actuation commands are then sent to the actuator interface which schedules commands for various onboard actuators and makes any necessary conversions before sending the resulting control signals to the actuators. Finally, this set of control signals is sent to the actuators in order to control the vehicle.

Superficially this structure seems to have no novel aspects, and from this level of abstraction that is not entirely incorrect. However, as we will see later, the supporting modules and the internals of the perception, planning, and plan

execution modules provide much more functionality and safety with respect to incorporating AI components than has been previously put into practice. The motivation for utilizing the sense-plan-act chain is twofold, namely encapsulation and low barrier to entry. While several autonomous driving platforms have been proposed that utilize end-to-end control, as was discussed in Chapter 3 this leads to an incredibly complex black-box and is subject to a host of issues pertaining to diagnosability and fragility. The low barrier to entry aspect is a direct result of this framework being commonplace in existing vehicular software as well as many related fields such as robotics and aviation. As such, utilizing this framework allows the architecture to be easily adopted and improves comprehension, maintenance, and diagnosis of software components.

4.2.2 Support Modules

Modules in this category provide additional functionality and redundancy to operational modules and are derived from several of the beneficial structures in previously discussed architectures. The agent interface, data management, simulation, and proactive safety modules constitute the support modules category. The agent interface allows bidirectional communication of sensory and planning data between the ego vehicle and external sources. This additional information provides redundant perceptual and planning data sources to the ego vehicle, intended to augment sensor fusion in the ego vehicle perception module and allow for route negotiation in the ego vehicle planning module. The data management module is responsible for storing initial parameter values, logging operational data, storing snapshots of RL parameters, storing map data and any other data storage needs of the vehicle. This module provides

many of the other modules with static data required for sufficient operation of the vehicle, provides data logging for diagnostics, and allows for the progressive rollback of potentially volatile RL algorithms. Finally, the simulation and proactive safety modules support the planning module by anticipating hazardous conditions before they occur. The simulation module is responsible for taking current perceptual and planning information and predicting how the environment will change within a short timespan. This information is then analyzed by the proactive safety module to determine if the future states violate safety requirements, and to what degree. This information is then sent to the planning module to assist with the selection of a final route and action plan for the next few timesteps.

These support modules provide several beneficial features to operational modules and make extensive use of traditional fault-tolerance mechanisms as well as anticipatory safety analysis. Similar to the note of low barrier to entry in the operational modules section, these modules have been incorporated into Sentinel because they are either currently being utilized in autonomous vehicles for other purposes or they are currently being developed and integrated into new lines of vehicles. Many of these components, however, have been used for other tasks such as over-the-air updates in the agent interface, or traditional data storage and logging capabilities of the data management module. While these two modules seem to have a strong foothold in existing and in-development vehicles, the same cannot be said for the simulation and proactive safety modules. However, an organic computing architecture [28] seen in Chapter 3 makes use of a simulator to avoid performing untested potentially dangerous actions and has been demonstrated in several robotics projects. Further evidence to support the notion of proactive safety aiding ve-

hicle planning will be given in the coming sections for the respective modules.

4.2.3 Safeguard Modules

The two modules in this category are responsible for ensuring fail-safe operation of the vehicle should the operational modules fail or the system is otherwise placed into an unsafe condition. The reactive safety module and control mode selector are modules that constitute this category. The reactive safety module is responsible for detecting unsafe conditions via sensory data and producing a corresponding set of safe plan execution commands to be sent to the control mode selector. This is done constantly so that in the event of unsafe conditions the reactive safety module will have plan execution commands at the ready to be enacted as quickly as possible. The control mode selector is a simple bypass of the main sense-plan-act chain, allowing the reactive safety module to take over in the event of unsafe conditions. If unsafe conditions are detected via the reactive safety module, the current dynamic driving task (DDT) fallback plan execution commands are sent from the reactive safety module to the control mode selector module. Upon receiving this DDT fallback, the control mode selector will switch to the DDT fallback mode until it no longer receives commands from the reactive safety module, indicating that a safe state has been reached. An important distinction to acknowledge is that the path from the sensory module, through the reactive safety module and control mode selector to the actuation module is a separate channel to the operational and support modules (excluding sensory and actuation modules). Some application domains, such as nuclear power, use logical and physical separation of safety and control to enhance safety in the presence

of increasingly complex control systems. Strict enforcement of this principle does not seem possible in other domains, and automotive vehicles is one such domain. However, the concept is incredibly useful, and there seems to be an opportunity to employ it in different ways. For our application, we recognized the possibility of separating important safety components from control components. The safety components we identified can be classified as those designed to ensure fail-safe behaviour. This separation of fail-safe safety from control allows for much higher clarity, maintainability and simplicity of these critical safety components. This is important to note because at first glance the architecture does not exhibit this separation due to the proactive safety component of the safety module having substantial interaction with the planning module. This grouping has been done to highlight the similarities between the two safety modules and does not detract from the aforementioned separation from control exhibited by the reactive safety component of the safety module.

4.3 Module Decomposition

This section describes each of the aforementioned software modules in greater detail and provides references to techniques and ongoing research that enable the corresponding features each module is intended to exhibit. Brief statements on positive and negative aspects are provided, that are eventually expanded in a more thorough discussion in Chapter 6, after an analysis of the architecture is presented.

4.3.1 Sensors

The sensory module encapsulates the on-board sensor suite and corresponding software components interfacing those sensors to other vehicle software. Lidar, radar, global positioning system (GPS), inertial measurement unit (IMU), and camera feeds are examples of sensors being deployed on autonomous vehicles that belong to this software module. V2X capabilities are relegated to the agent interface due to the potential for two-way communication, while this module only involves sensing the environment or internal vehicle state. The data collection and any preprocessing performed on raw sensor data belong to this module. The use of redundant sensors is advised but not explicitly required, as the additional cost to automotive OEMs is prohibitive and potential external sources of sensor redundancy have been incorporated. Additionally, sensor error detection is captured in this module, to be used by safety modules for initiating a DDT. This module is responsible for producing a set of raw and/or preprocessed sensor data to be sent to the perception module for sensor fusion and reality model generation tasks.

4.3.2 Perception

Broadly-speaking, the perception module is responsible for constructing a model of reality to be utilized by the planning module. This task normally involves sensor fusion and sensor data consolidation and involves multiple features in the environment being identified.

Initially, the issue of reliability and lack of redundant sensors on-board the ego vehicle was troubling, to say the least. However, many autonomous vehicles are beginning to incorporate V2X capabilities, where some redundancy

may be leveraged. The perception module receives sensor data from the ego vehicle sensor module, as well as perceptual data from the V2X component of the agent interface. Neighbouring vehicles and/or infrastructure may send perceptual data to the ego vehicle, thereby providing redundancy to the on-board sensors, and mitigating potential blind spots. An example of blind spot mitigation through V2X has been demonstrated in recent research [30], giving some credence to the use of V2X to expand individual vehicle perception. The concern of security and accuracy of said data is valid and will be discussed in subsection 4.3.7.

Not being able to rely on external sensory data at all times, perceptual redundancy on the ego vehicle sensor data is imperative, especially when utilizing artificial intelligence techniques such as machine learning. The perception module makes use of both artificial intelligence algorithms and traditional algorithms for perception, with the intent of partial feature space overlap between any artificial intelligence algorithm and at least one traditional algorithm. This is done as discussed through a variety of sources in Chapter 3 to reduce the risk of any singular AI component failing by performing cross-validation against traditional alternatives. As an example, a camera may be utilizing a neural network for pedestrian detection, while a more traditional algorithm may utilize RADAR data for obstacle detection. Since pedestrians are a subset of the obstacle feature space, we can utilize the more traditional algorithms to partially validate the neural network. These additional algorithms will likely increase the computational complexity of the perception module, but the redundancy and effects down the sense-plan-act chain may justify this cost.

Another distinction between the proposed solution and existing ones is that

probabilistic outputs should be utilized when possible for each of the perception algorithms in the module. This is done so that the perceptual fusion component can properly combine probabilities of each feature to construct a final more accurate probabilistic reality model. Many machine learning algorithms already supply probabilistic outputs [26], and neural networks can readily be modified to produce probabilistic outputs [38]. Each of these probabilistic outputs on each feature is then sent to the perceptual fusion component for consolidation. A concern with these outputs is the validity of the probabilistic outputs they provide, with the vast majority of them relying on statistical analysis methods based on their training data, which as discussed earlier, may not accurately reflect reality. This concern is mitigated by utilizing many conventional and AI algorithms in conjunction to reduce the chance of finalized outputs poorly reflecting reality. A tentative solution to consolidating these probabilistic outputs is through the use of Dempster-Shafer theory, a framework for reasoning with uncertainty. Dempster-Shafer theory has been successfully applied to the context of sensor fusion for autonomous vehicles [14], and has also been applied to risk detection of external vehicles [17]. Utilizing this technique and potentially others from the fields of probabilistic logic and decision theory, a finalized collection of probabilities on each feature is presented as the constructed reality model. Whichever technique is used, probabilistic outputs allow the planning and plan execution modules to make more nuanced decisions. Recent work compared the discrete outputs of neural networks to the probabilistic outputs of bayesian deep learning and found that the probabilistic outputs allow the planning component of the system to act more cautiously in uncertainty [26]. This more nuanced decision-making process utilizing probabilities combined with the benefit of redundant sensory

and perceptual sources are the guiding mechanisms of the perception module.

4.3.3 Planning

The planning module is responsible for generating a set of routes and associated high-level actions, to be utilized by the plan execution module to generate high-level plan execution commands. These tasks require a minimum of the aforementioned reality model and a specified goal the plan aims to achieve. Several other modules are utilized to improve the safety of generated plans.

Since planning algorithms may use traditional or AI methods to generate plans, the framework proposed in the perception module carries over to this planning module. Multiple planning algorithms may be utilized to provide additional redundancy and may be used for cross-validation by comparing the geometries of each generated path to one another and in the context of objects in the reality model. To further increase redundancy and allow for more nuanced decision-making planning algorithms may utilize planning data received from V2X sources. This allows the ego vehicle to receive other vehicle plans, planning instructions from infrastructure, and other planning data that may aid the ego vehicle in constructing its own plans. Substantial research has been put towards cooperative planning through V2X [18][36], validating some of the benefits discussed here. Whichever planning algorithms the planning module makes use of, the HMI route input of the agent interface must supply a target goal. This can be a destination, several destinations, waiting times, etc. and are needed for the vehicle to make a corresponding action plan to achieve that goal. In the event of the autonomous vehicle being used in a service-oriented manner (taxis, goods transportation, etc), an additional "default"

goal such as returning to a service depot can be added to the data management module.

The majority of existing planning techniques solely rely on the current reality model and goal for plan generation. Humans, however, do not plan vehicle routes this way. Humans take note of other vehicles and entities in the environment, track their trajectories, and plan according to a predicted future state. For example, if a lane-change manoeuvre is desired and another vehicle is approaching in the other lane at a high speed, a human would often wait for the vehicle to pass before initiating a lane change. Operating on only the current reality model may lead to an unsafe condition where the lane change manoeuvre is undertaken while the vehicle in the neighbouring lane is approaching. To mitigate this, some recent research has been done on predictive planning methods [20], rather than the traditional reactive approach. To enable predictive planning, we first send the collection of generated routes to a simulation module, where the environmental state and routes are used to predict near-term future states of entities in the environment. This information is then sent to the proactive safety module for a safety analysis to be conducted on those future states. This information is then sent back to the planning module to restrict the output set of plans to those which have the least predicted risk. This proactive safety approach to planning is intended to avoid unsafe conditions, and to avoid relying too heavily on the reactive safety component of the vehicle.

A concern with regards to the planning algorithms is that different algorithms may utilize a different subset of the reality model, of which each component has an associated probability of belief. As such, the collective probability of reality model components utilized by each planning algorithm

should carry through to the final generated plans. This is done so that plans generated on high probability environmental data are favoured over those that utilize lower probability data. From this, we conclude with the output of the planning module being a collection of routes/action plans, as well as associated predicted risk and probabilities of belief derived from the utilized reality model data.

4.3.4 Plan Execution

The plan execution module is responsible for generating a final set of control execution commands that enacts one of the provided action plans. These commands are to be sent through the control mode selector module to the actuator interface module, to be utilized for actuator control.

Due to the lack of AI techniques being utilized for vehicle control outside of the already discussed end-to-end control, this path execution module does not make use of parallel conventional and AI algorithms. This not only avoids unnecessary complexity in path/plan execution but also reduces the computational complexity of this component of the architecture. Some work has been done on utilizing machine learning to capture end-to-end defensive driving behaviour [25] as well as estimating driving comfort [15] which may play a role in selecting a final set of control outputs. As was discussed in Chapter 3 end-to-end control should be avoided due to the numerous safety and diagnosability concerns it presents, whereas estimating driving comfort should be placed in the perceptual module.

To reduce the set of action plans to a singular output set of high-level control signals, it is proposed that the action plans be processed in order of

the highest probability of belief and lowest predicted risk combination. This allows the plans generated on the most accurate data and the plans with the lowest predicted risk to take precedence over others. Each of these plans, once processed by the path execution algorithms, are then to be evaluated against vehicle dynamics data from the data management module. The first set of control signals to pass this final check are then sent to the actuators module, and the remaining plans are discarded. This is done to reduce the computational load of the path execution calculations, and to avoid wasting time on less promising action plans.

4.3.5 Actuator Interface

The actuator interface module encapsulates the actuator-specific electronic control unit (ECU) modules responsible for executing specific portions of the path execution signals provided by the control mode selector module. Path execution commands are sent to corresponding ECUs and converted into timed low-level control signals to be sent to the actuators.

4.3.6 Actuators

The actuation module encapsulates the on-board actuator suite and corresponding software components interfacing those actuators to other vehicle software. Engine throttle, steering, braking, and signalling are examples of actuators necessary for vehicular control. The conversion of low-level control signals from the actuator interface into actuator-specific control signals is the primary responsibility of this module. Many of these devices do not have redundant counterparts, and so presumably have high enough reliability and

robustness to not be of concern over some specified operating time and environmental conditions. Additionally, actuator error detection is captured in this module, to be used by safety modules for initiating a DDT fallback.

4.3.7 Agent Interface

The agent interface module is responsible for allowing direct communication with external agents, namely other autonomous vehicles, infrastructure, and passengers. This communication with other agents allows the vehicle to make more informed decisions and aids in passenger safety and satisfaction.

The V2X component houses the V2V and V2I subcomponents responsible for communicating with external autonomous vehicles and infrastructure, respectively. As was mentioned in subsection 4.3.2 and subsection 4.3.3, this communication enables additional redundancy for perception and planning algorithms and may reduce blind spots as well as providing a negotiating mechanism for planning. A major concern around this, however, is that of security and validity of external data. Communication channels may be compromised by malevolent agents, with false data potentially leading to erroneous vehicle behaviour. Additionally, even if the data is securely communicated the validity of the data is unknown by the ego vehicle. As such, if V2X data is to be utilized, strict security mechanisms must be used and the data must be given second-class privilege when compared to on-board data. Research on evaluating safety and security of this information is ongoing, with some promising results demonstrating an end-to-end testing framework [44]. Yet another difficulty is the format of the data received, where each automotive OEM may utilize a different format for transmitted data. This issue may

hamper the immediate development of V2V communication across different OEMs, and regulations may need to be imposed on what data is available for optional transmission, and what format it should take to standardize this across different manufacturers' vehicles.

The second component of the agent interface is the HMI, which enables the vehicle to communicate with passengers. This is the primary mechanism by which the vehicle can produce and execute an action plan, as destination(s) are required to generate routes the vehicle should follow. The HMI also includes a passenger alert component, responsible for alerting the passengers to non-recoverable faults, such as engine malfunction. This is done so that if the reactive safety component detects such faults, the passengers can be brought to a safe state and instructed on steps to ensure their continued safety. This is done only in the most critical of non-recoverable states, such as engine failure or detection of fire. This mechanism is inspired by safety notification systems onboard aircraft that direct passengers to safety in the event of a crash.

4.3.8 Data Management

The data management module is responsible for storing and allowing access to data pertaining to vehicle operation. Several components provide a support role to the sense-plan-act chain, such as knowledge databases and maps databases for localization and route planning tasks. An additional AI snapshot database has been added based on issues discussed in Chapter 3 on RL and traditional fault-tolerance techniques. To avoid extreme overfitting cases this component may take snapshots of the current parameters of each RL algorithm being used. Should the performance of any RL algorithm falter, the

algorithm could be rolled back to a previous iteration. This sacrifices some of the knowledge learned by the algorithm since the last snapshot but may provide a mechanism to avoid the overfitting issue.

Additionally, a large component of data management involves data logging of vehicle state, software faults, malfunctioning hardware, etc. These data logging mechanisms could additionally be used to monitor adaptive mechanisms such as machine learning algorithms in the sense-plan-act chain. This information could then be retrieved from the vehicle for offsite analysis, potentially providing an excellent diagnostic and evaluation source for deployed machine learning algorithms.

4.3.9 Simulation

The simulation module is responsible for producing a set of predicted near-term future states for each action plan it simulates. As inputs, it receives perceptual and planning information from the respective modules in the sense-plan-act chain. Utilizing this information, object tracking and predictive planning mechanisms enable near-term future locations of objects in the environment are possible.

Current techniques for planning often rely solely on current environmental data, relying on the speed and frequency of planning updates to react to environmental circumstances. However, this reactive approach leads to erratic vehicle behaviour that could otherwise be avoided. To avoid undesirable scenarios from the onset, predicting near-term environmental changes is beneficial.

Recent work on model predictive control and predictive planning are en-

abling these near-term future states to be predicted [26]. Simulation models are often simplified to allow for the simulation to be computed in real-time. A concern surrounding this involves the accuracy of such models, especially as the model becomes more simplified. This concern can largely be relegated to future increases in computing capabilities and custom hardware solutions, a trend that seems to be continuing in the automotive domain [42].

4.3.10 Safety

This section deals with modules that can be grouped under the larger abstraction of safety. These modules are responsible for ensuring vehicle safety under a variety of conditions, taking on support and safeguard roles. Proactive safety takes on a support role, meaning it is interconnected with the main sense-plan-act chain. Reactive safety and the control mode selector are safeguard modules, forming a short-circuit of the sense-plan-act chain in the event of severe and/or immediate safety concerns.

It should be noted that both proactive and reactive safety modules are replicated in triplicate. This redundancy of safety modules, especially the reactive safety module is done to ensure that a single point failure of any safety computation can be tolerated. Without this measure, a single point failure could result in extreme safety-oriented behaviour being taken when it shouldn't, resulting in a best-case scenario of passenger discomfort, and a worst-case scenario of the erratic behaviour causing harm.

4.3.10.1 Proactive Safety

The proactive safety module is the second stage in the predictive planning process. This module receives predicted near-term future environmental states from the simulation module and a set of safety requirements common to both the proactive and reactive safety modules. This module is responsible for performing a safety analysis on the set of predicted future states linked to each candidate action plan initiated by the planning module.

The safety analysis associated with each action plan candidate involves checking predicted states against a set of safety requirements to determine the number and severity of safety violations in each set of predicted states. This is done for each set of predicted states and the final collection of safety reports is sent to the planning module. The planning module can then use this information to decide on a subset of candidate action plans that pose the least amount of risk to future scenarios.

4.3.10.2 Reactive Safety

The reactive safety module is responsible for sensing unsafe conditions, providing continual fail-safe control actions, and assuming control over actuators in the event of unsafe conditions. This module receives environmental information from the perception module and safety requirements from the common safety requirements data store. The limitation of this approach is that it relies on perceptual data rather than raw sensor data, where the former may have inaccuracies introduced via the AI algorithms contained within the perception module. This was chosen since the reactive safety module may not have enough information to make an appropriate decision with raw sensor data. The dis-

tinction between a pedestrian and a plastic bag, for example, is an important one in a reactive scenario and must be dealt with accordingly. Unless predictable alternatives to the currently used machine learning algorithms can be developed to handle these situations, we will likely have to make reactionary decisions based on perceptual data rather than raw sensor data.

Using this perceptual data and the safety requirements shared with the proactive safety module, the reactive safety module should determine whether a safety violation has occurred, and a suitable response. Regardless of if a safety violation has occurred, the reactive safety module should always have a set of control actions ready should there be too little time for re-computation. In the event of a safety violation, the reactive safety module sends its set of control actions to the control mode selector, enabling the safety bypass of the sense-plan-act chain. Once a safe state is reached, the control signals are no longer sent to the control mode selector.

4.3.10.3 Control Mode Selector

The control mode selector is responsible for switching control between the main sense-plan-act chain and the reactive safety module. This selector enables control for the reactive safety module so long as it is receiving control actions from reactive safety. Once control signals cease from the reactive safety module, control is switched back to the main sense-plan-act chain. This enables fail-safe operation of the vehicle under unanticipated hazardous conditions.

4.4 Revision History

This section describes the revision history of Sentinel to highlight how the final version was reached. The intent here is to highlight many of the features considered and the progression from one version to another. While a multitude of revisions were made, four of those versions showcase substantial changes from previous iterations, and these are the focus of this section.

4.4.1 Milestone 1

In one of the earliest versions of Sentinel we find some common traits carried through to the final architecture. In this first milestone, seen in Figure 4.3, we see an N-version end-to-end AI system controlling the vehicle. This was a direct result of the information in Chapter 3 pertaining to the use of multiple AI systems for cross-validation and redundancy. These outputs merge into a collection to be sent to an arbitrator module as was done in the arguing machines framework (subsection 3.4.1), where one of the end-to-end outputs is to be selected for output to actuators. Even in these early stages the notion of proactive safety via predictive planning can be seen in the arbitrator state prediction and safeguard safety analysis, intending to help the arbitrator select the control output with the highest predicted safety value.

Several glaring issues existed with this version, however. Utilizing end-to-end AI systems leads to an inability to diagnose erroneous system behaviour. The arbitrator module encompassed too many duties, specifically state prediction, later more aptly revised into a simulation layer. Additionally, only utilizing AI methods severely limits the incorporation of existing more conventional software into the system. At this stage no fail-safe bypass of the

main control loop and no methods for external agent interaction are present. The overall structure of solely utilizing end-to-end control also did not coincide well with existing automotive and robotics architectures currently in use, potentially leading to a barrier to adoption.

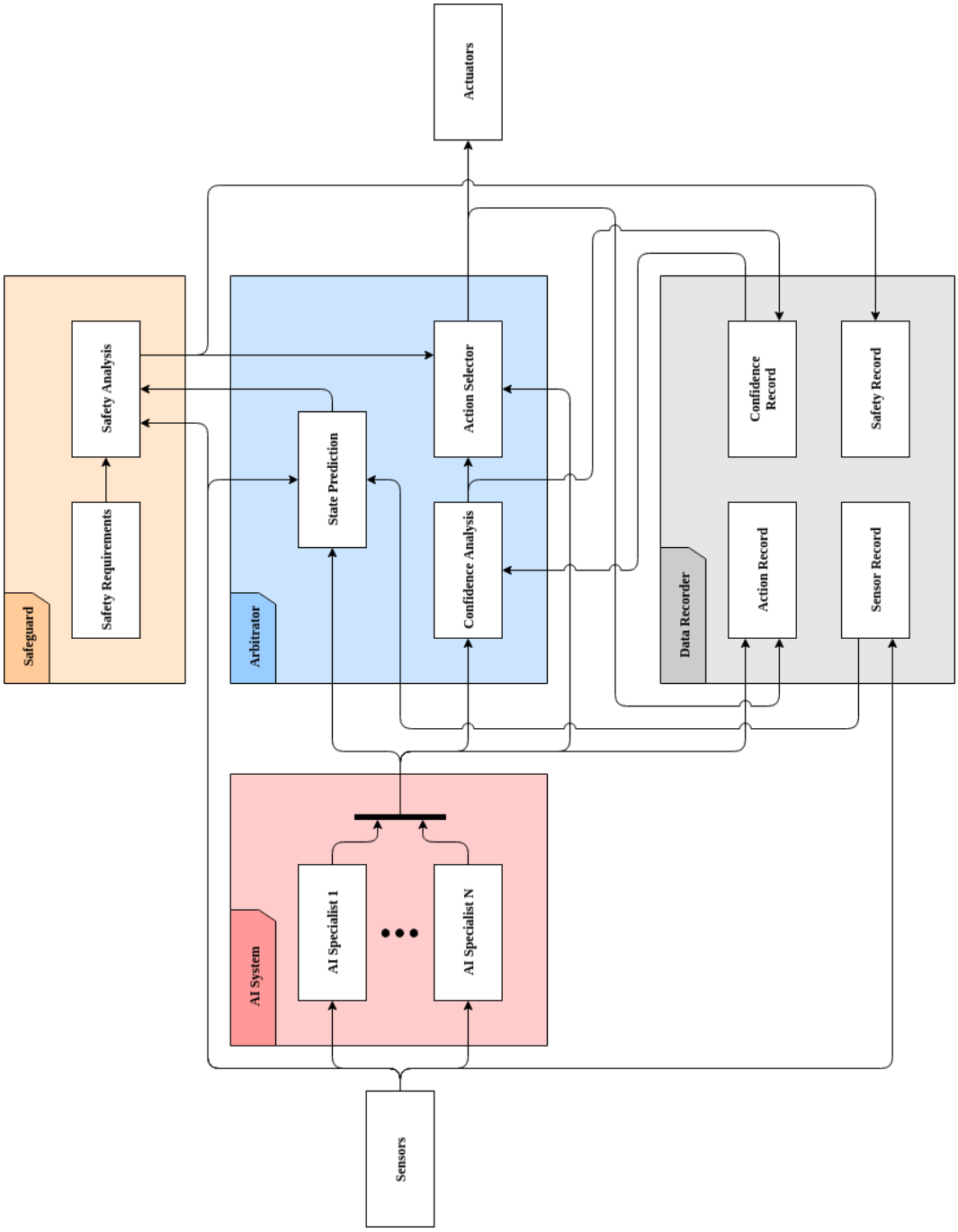


Figure 4.3: Sentinel Milestone 1

4.4.2 Milestone 2

Milestone 2 (Figure 4.4) features a major revision of the first milestone into an architecture better suited to the automotive domain. The end-to-end AI system was decomposed into the traditional sense-plan-act chain, allowing AI algorithms utilized in each component to be refined in scope improving diagnosability, as was seen in several of the architectures discussed in Chapter 3. This decomposition aligns well with existing automotive architectures, increasing the chance for adoption among automotive OEMs. Auxiliary software modules were incorporated alongside AI modules to aid arbitrators in determining the validity of each AI module. This notion of cooperatively utilizing more traditional software modules alongside AI-based methods carried through to the final version. The AI modules were decomposed into a policy and a monitor for that policy, intended to determine whether the policy performed sufficiently well on a subset of the original training data.

While substantially improved over milestone 1, this version also had quite a few failures. Like milestone 1, this version lacks a fail-safe bypass for if the main control loop fails to keep the system in a safe state, and has no means of interfacing with external agents. Additionally, the monitor component of the AI modules was only intended for RL algorithms, which could change in dangerous ways during runtime. This notion still applies to the final version but was later abstracted and relegated as a potential feature if RL was used for any AI component, rather than being a standard feature for all AI-based methods.

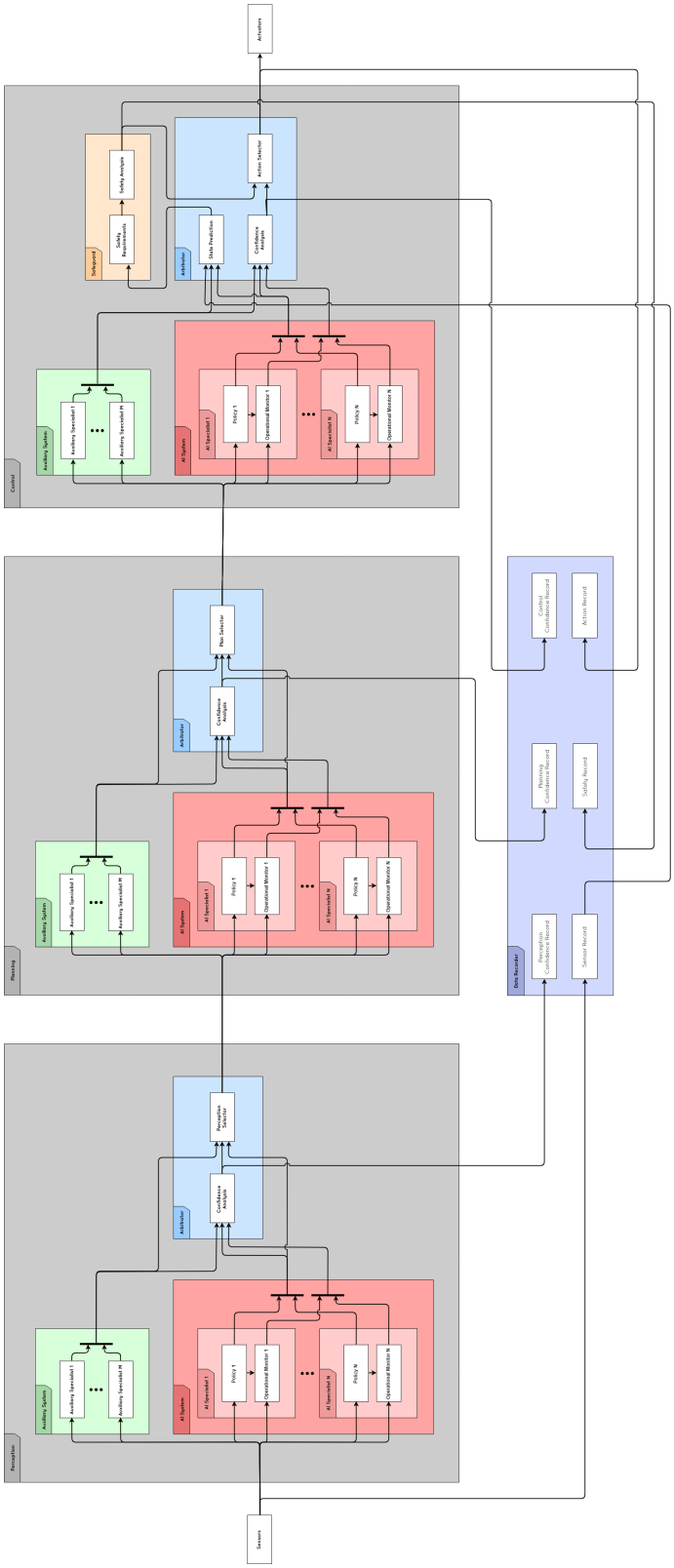


Figure 4.4: Sentinel Milestone 2

4.4.3 Milestone 3

Milestone 3 (Figure 4.5) further refined previous versions by distinguishing between proactive and reactive safety components, and adding a fail-safe bypass of the main control loop, as seen in the RTA framework (subsection 3.4.2) and the Safe-AV architecture (subsection 3.6.4). State prediction was removed from the control module and amalgamated into the proactive safety module. In hindsight, this was just shifting the responsibility on another module. AI components were decomposed into a baseline and a protective version, intending to capture cautious behaviour. This exercise was inspired by the biomimicry architecture (subsection 3.5.1), attempting to capture normal, cautious, and reflexive behaviour seen in biological organisms. Switches were added between the baseline and protective AI modules so that should danger be detected by the reactive safety module, it could switch to a more cautious operating mode.

Still, a few concerns remained with this version of Sentinel, namely that it still had no means of interacting with external agents, including passengers. Additionally, the use of protective AI systems in the sense and control modules was deemed impractical. The rationale in biological organisms to exhibit this behaviour such as heightened senses in dangerous situations is that the brain only has so much processing power and must divert resources to sensing and control. Given the continued trends in computational improvement of computers and custom solutions for autonomous vehicles, sensing and control can be computed in realtime with exceptional precision. The cautious mode for planning was ultimately relegated to a byproduct of predictive planning and proactive safety. The high coupling of safety and control was still of concern, especially with regards to the reactive safety module.

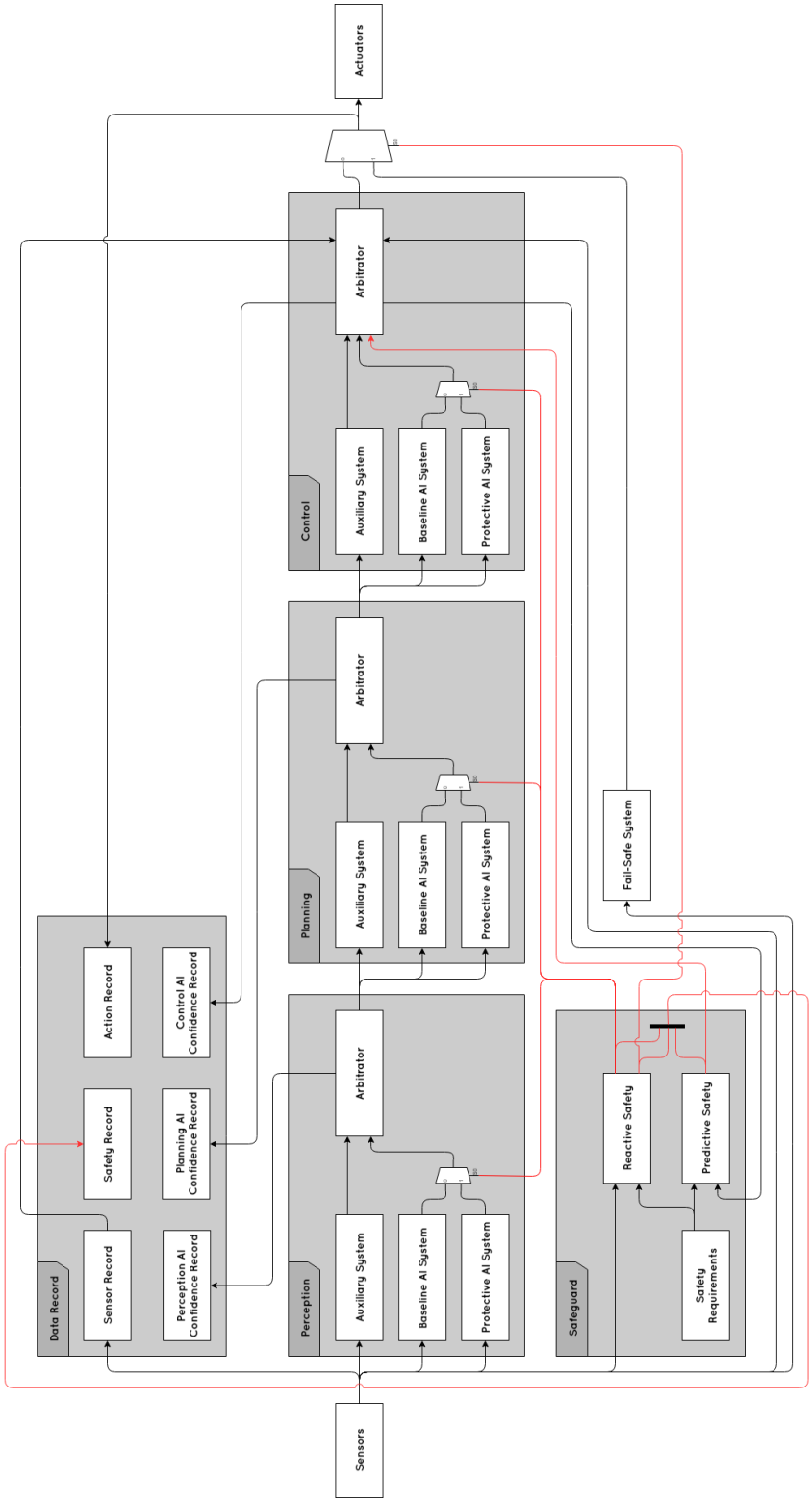


Figure 4.5: Sentinel Milestone 3

4.4.4 Milestone 4

At last in milestone 4 (Figure 4.6) we see an architecture that resembles the final version. State prediction was expanded into a simulation module, similar in design to the organic computing architecture (subsection 3.5.2). Safety subsystems were replicated in triple modular redundancy (TMR) to ensure safe operation if any singular safety subsystem clone failed. An agent interface was added to improve the redundancy of sensory and planning algorithms and to allow for more nuanced decision-making strategies in a multi-agent environment. We also see that proactive safety takes on the role of switching to more cautious modes, while the reactive safety subsystem is isolated from the main sense-plan-act chain. At this point switches and protective AI modules are still present, to be removed in the final version. A secondary sensor suite was added for redundancy and further separation of safety from the main sense-plan-act chain.

Some minor concerns still existed at this point, leading to the changes seen in the final version of Sentinel. The secondary sensor suite would likely incur massive costs to automotive OEMs and so was removed. As discussed in milestone 3, the use of protective AI modules in the perception and control modules was deemed inconsequential, and the protective functionality of the planning module stems more so from the proactive safety feature than specific AI algorithms. Some other minor connection issues were identified and the overall clarity of the diagram was then improved to arrive at the final version of the architecture shown in Figure 4.2.

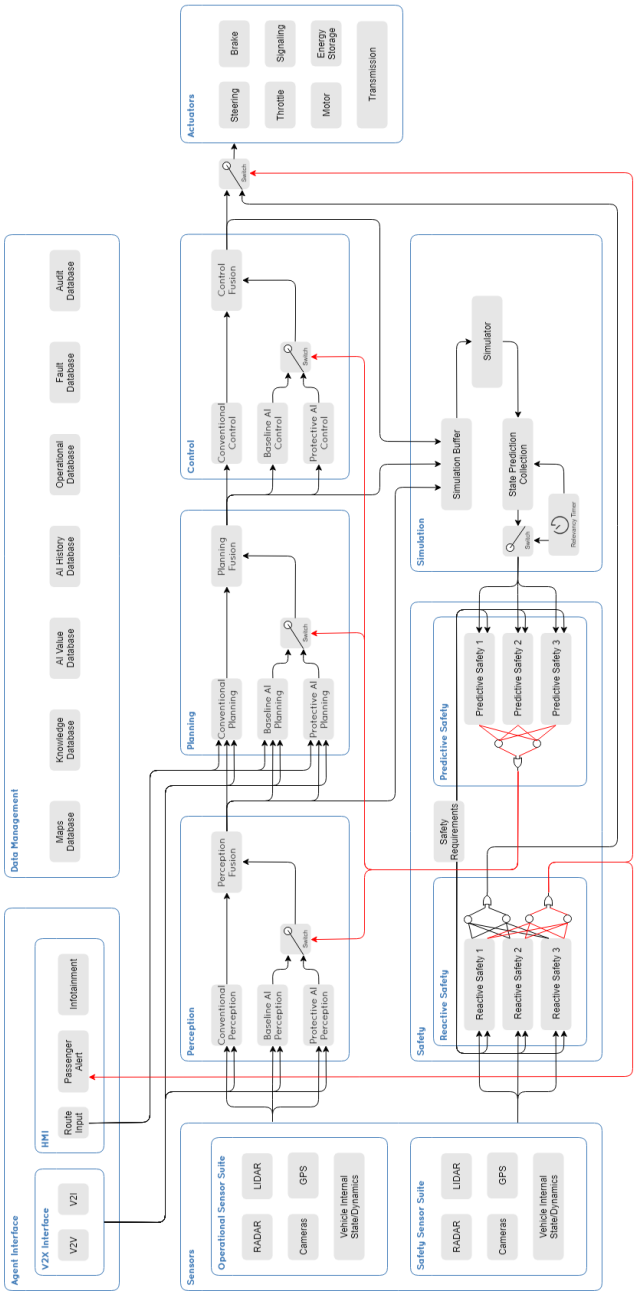


Figure 4.6: Sentinel Milestone 4

Chapter 5

Assurance Case

An assurance case is a structured argument supported by rationale, assumptions, and evidence, and is used to demonstrate that a system exhibits some property within its intended operating environment [45]. These arguments are sometimes presented in a graphical fashion, with goal structuring notation (GSN) being one of the more popular formats. Shown in Figure 5.1 below is an example assurance case of wide area multilateration used in military aviation [31].

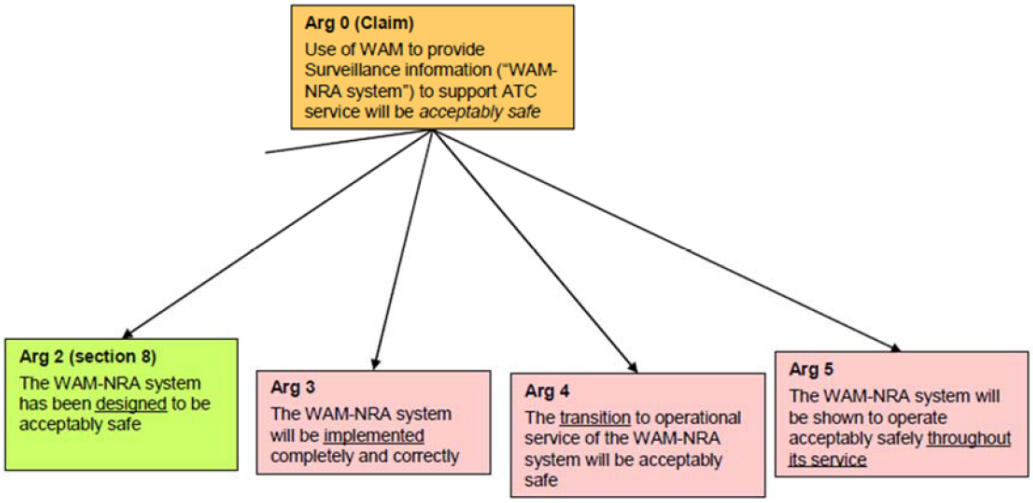


Figure 5.1: Excerpt from Eurocontrol WAM PSC High-Level Argument [31]

Completing a quantitative evaluation of Sentinel was deemed excessive within the scope of this thesis, and relegated to future work. Thus a qualitative analysis of the architecture was needed, and an assurance case was chosen to fit that role. This assurance case delineates the assumptions made and evidence required to claim SAE J3016 level 5 compliance.

5.1 Assurance Case Diagrams

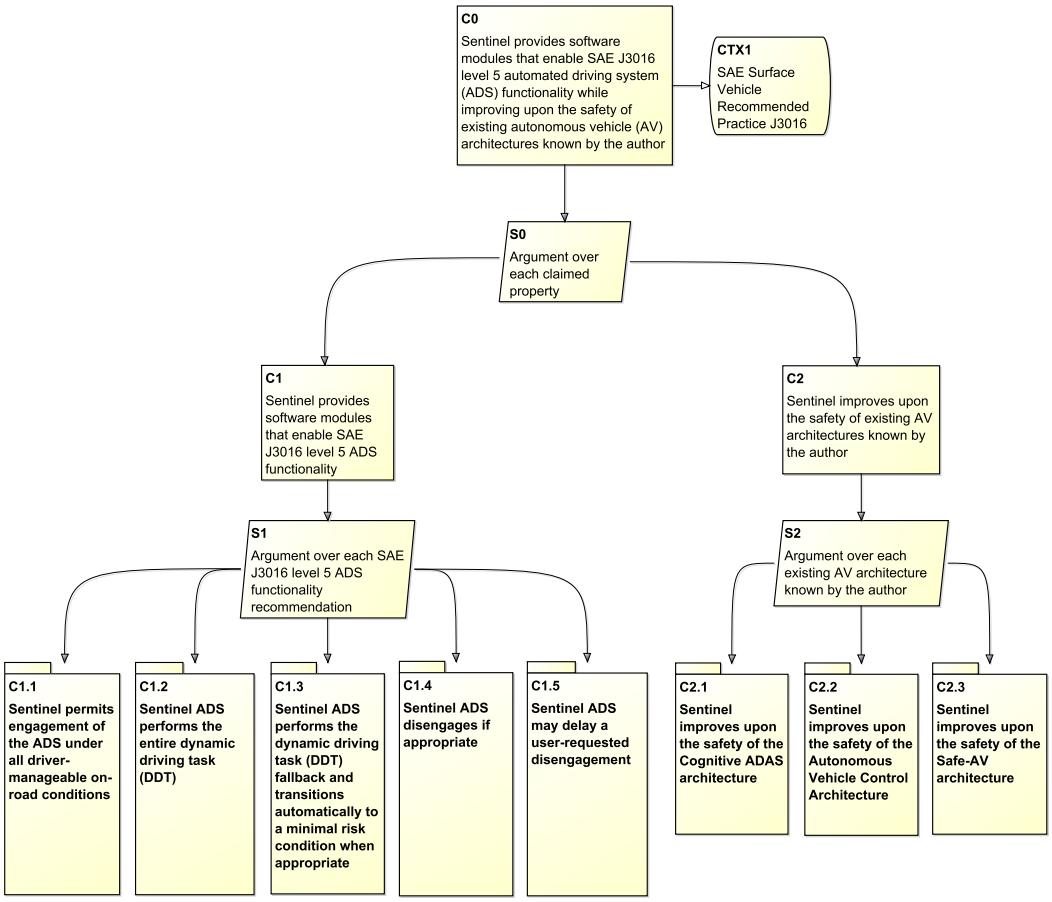


Figure 5.2: Top-level claim of J3016 compliance and safety enhancement

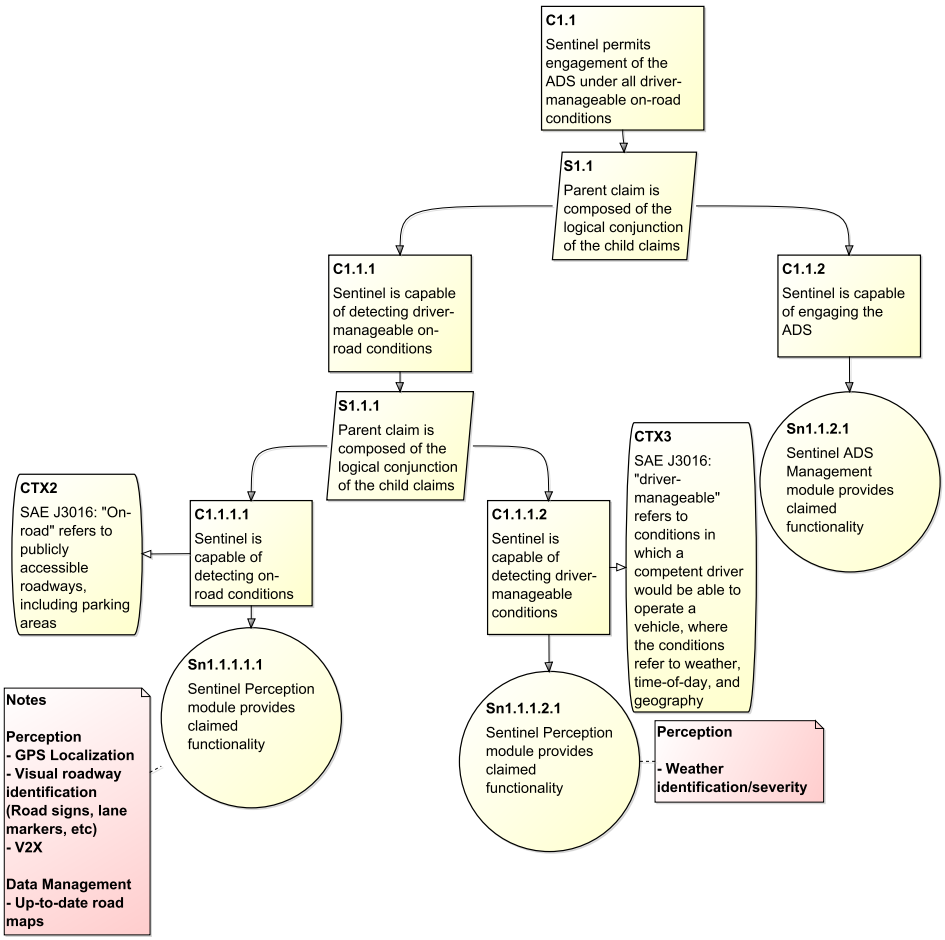


Figure 5.3: J3016 compliance claim 1/5

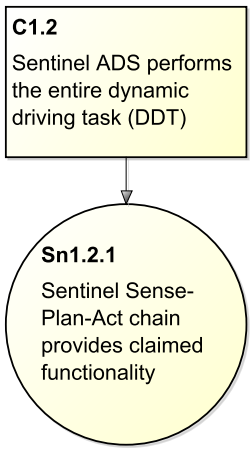


Figure 5.4: J3016 compliance claim 2/5

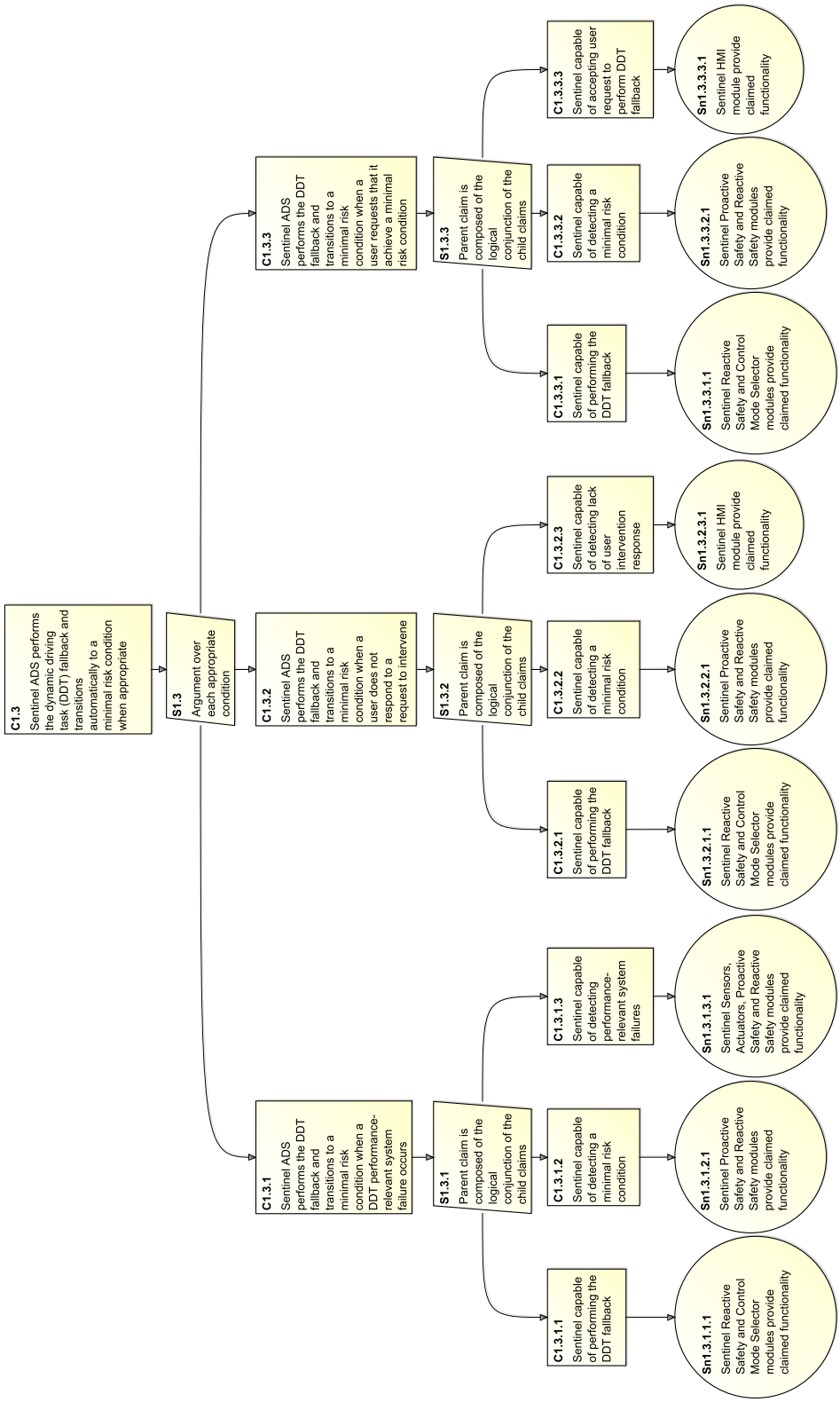


Figure 5.5: J3016 compliance claim 3/5

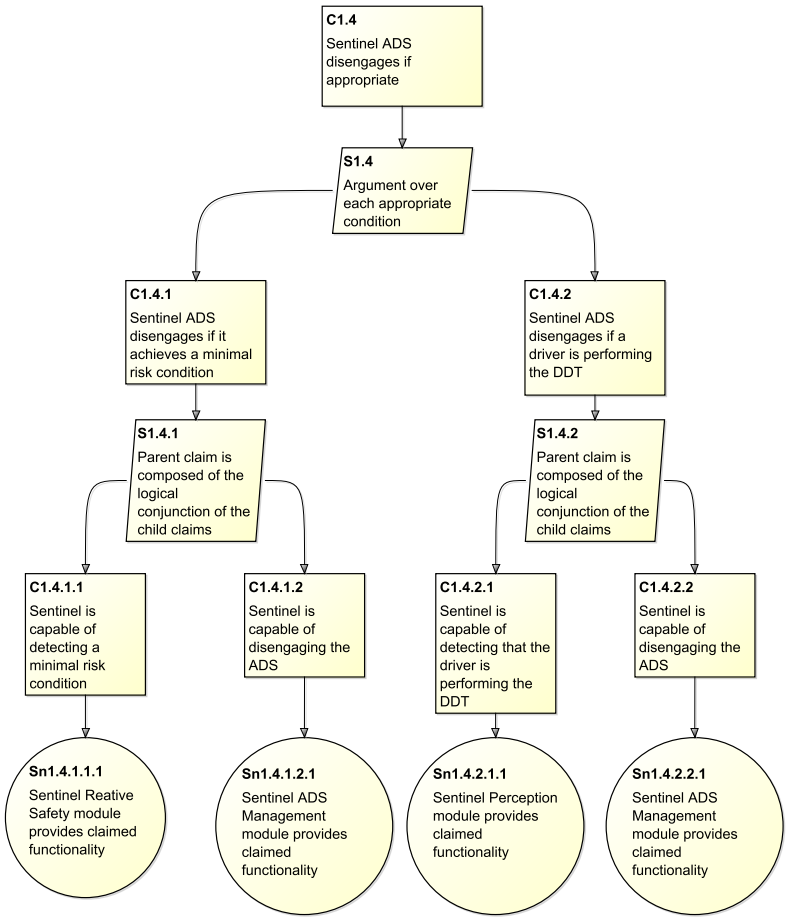


Figure 5.6: J3016 compliance claim 4/5

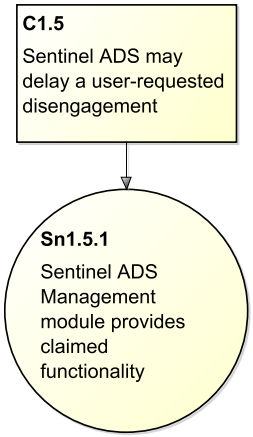


Figure 5.7: J3016 compliance claim 5/5

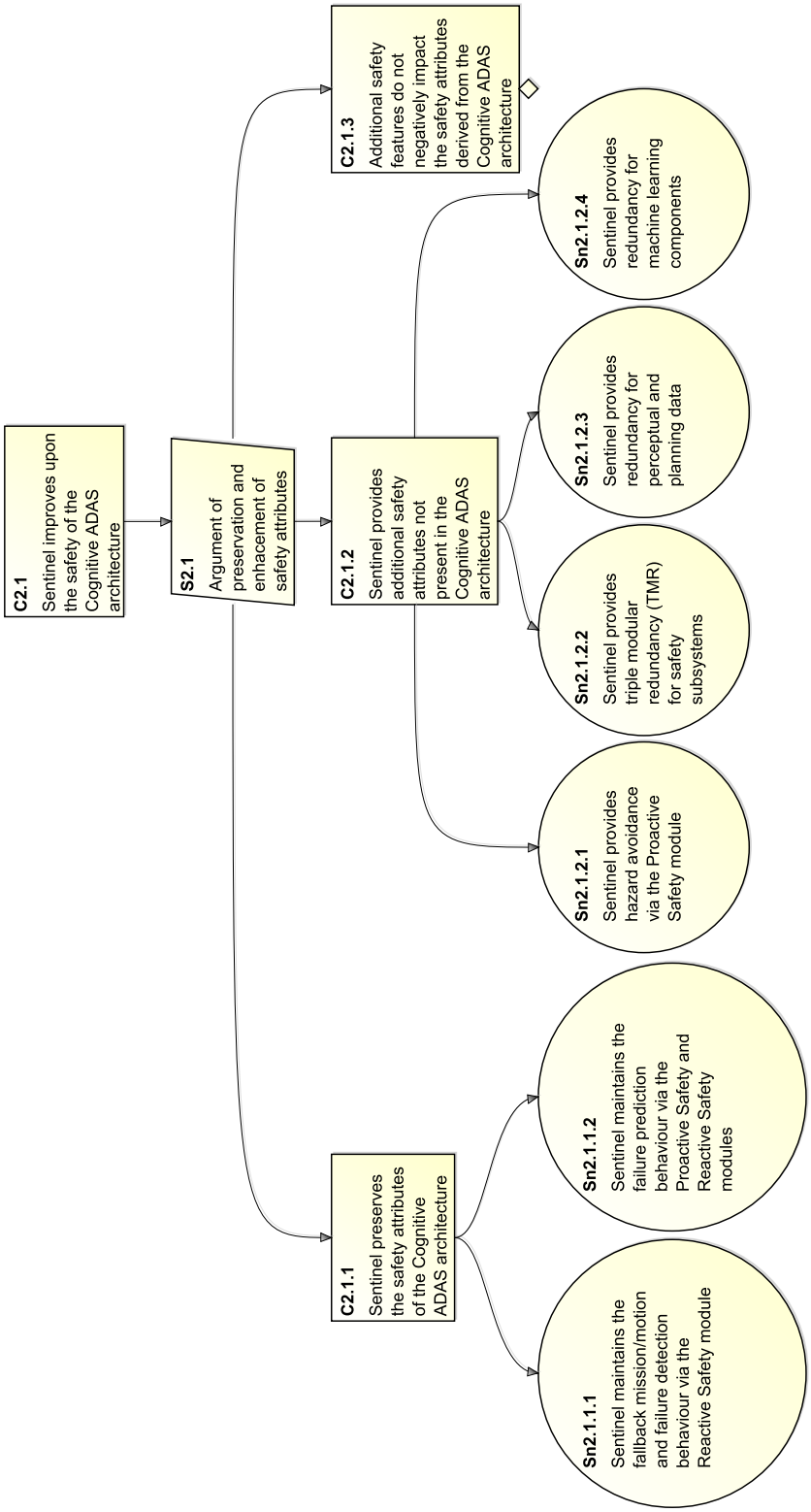


Figure 5.8: Improvement in safety attributes over existing architectures 1/3

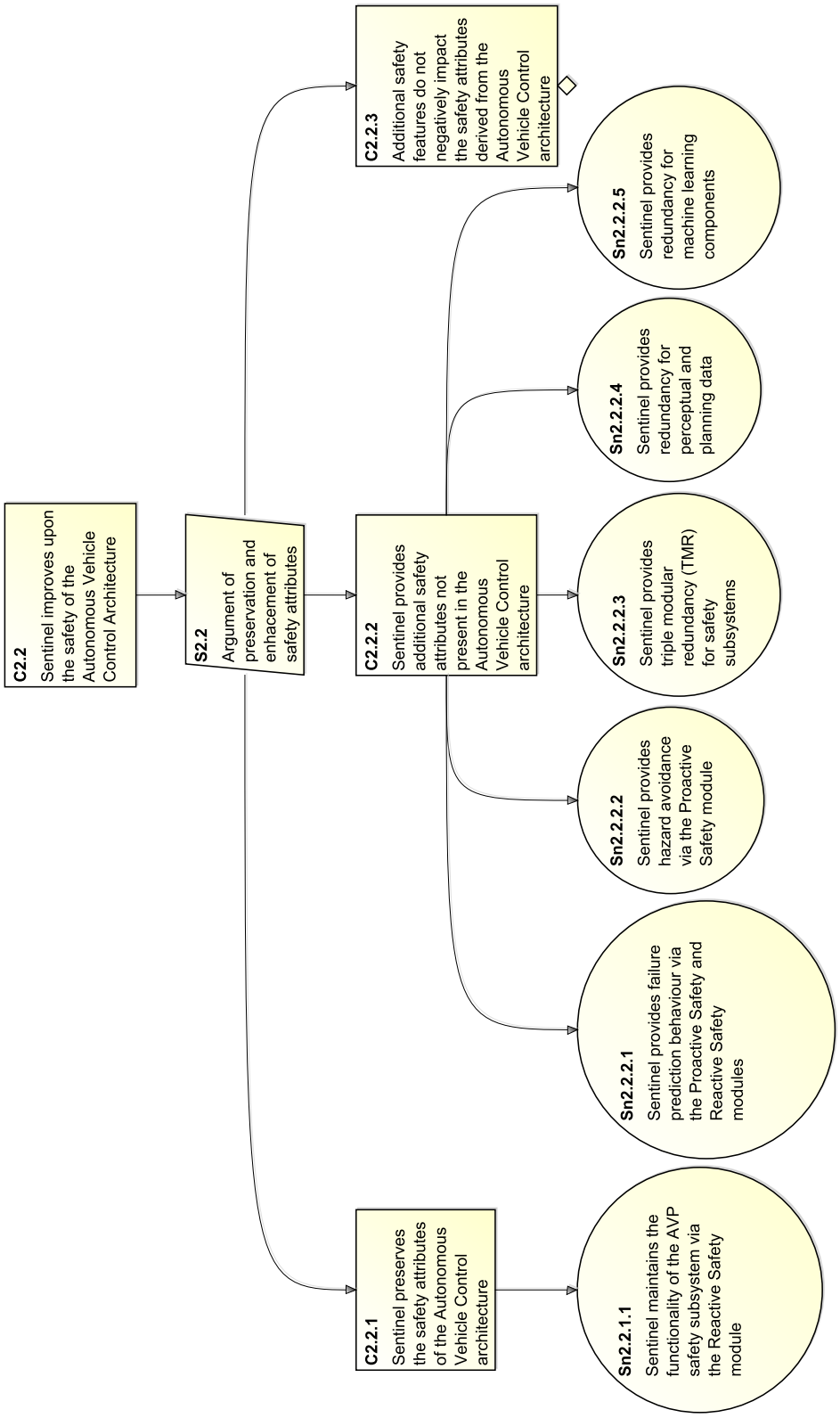


Figure 5.9: Improvement in safety attributes over existing architectures 2/3

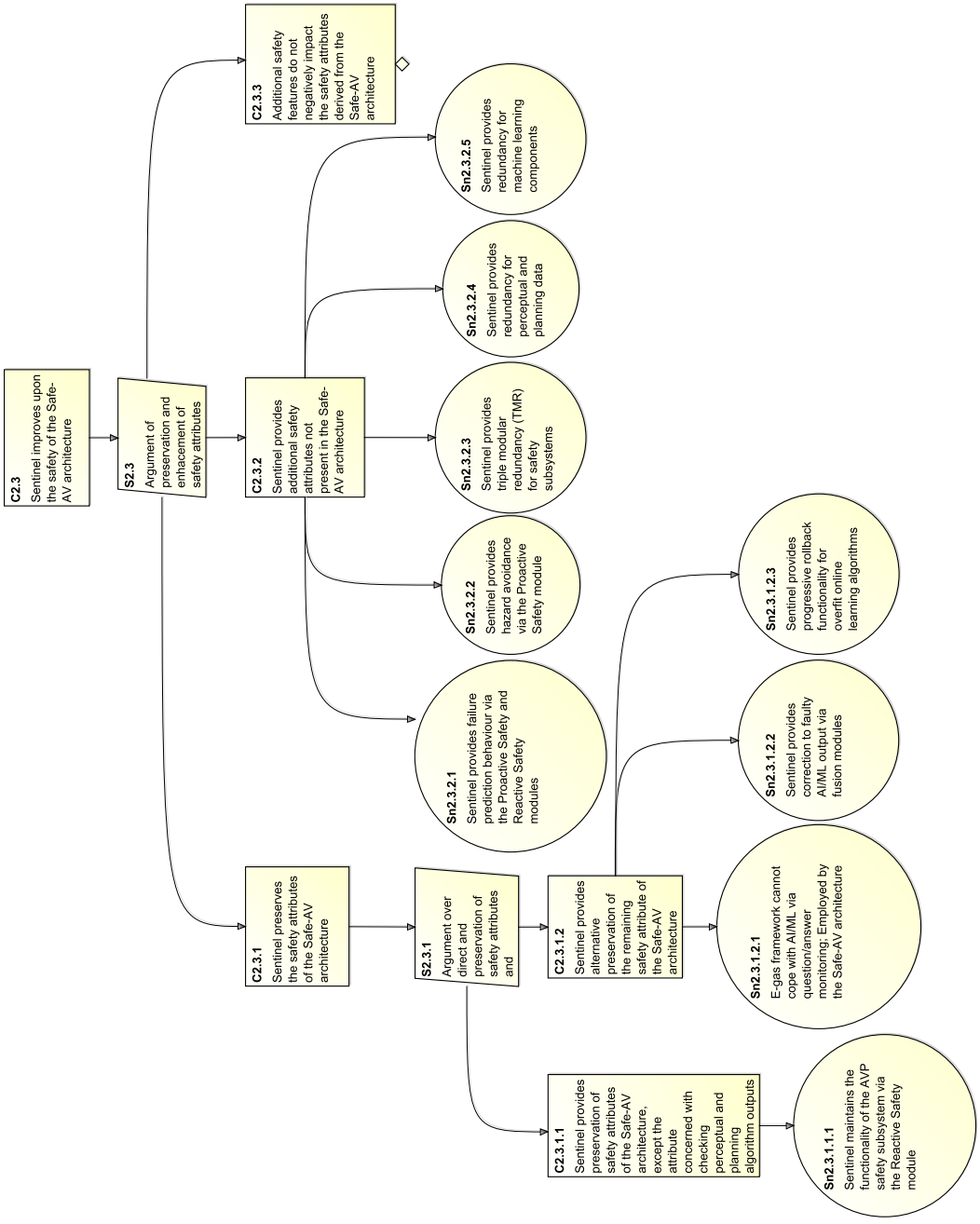


Figure 5.10: Improvement in safety attributes over existing architectures 3/3

5.2 Discussion

The assurance case in the previous section demonstrates which modules in the Sentinel architecture contribute to claims about SAE J3016 functionality as well as improved safety behaviour over existing known autonomous vehicle architectures. Some notes have been provided on some of the evidence nodes to delineate additional decomposition and techniques that may be used to provide sufficient evidence node coverage. Through this assurance case, we can see that Sentinel provides modules that cover all of the SAE J3016 level 5 recommendations, where each of the recommendations was decomposed into lower-level claims. Alongside this, each decomposition provides a strategy for the breakdown, intending to justify why the child claims prove the parent claim.

The justification for improved safety over existing known autonomous vehicle architectures was more difficult to conceptualize since Goal Structuring Notation does not contain negative evidence for a claim. While many of the safety features of the reviewed architectures were incorporated into Sentinel, some of them were not. For instance the 3-level monitoring behaviour of the Safe-AV [35] architecture was not included in Sentinel due to the massive computational overhead of periodically testing machine learning components, but also since the testing would likely have to be based on previously curated data which as discussed in 3 does not necessarily tell us anything about whether the machine learning model is failing or if that training data did not capture the current runtime events.

This assurance case shows that at a high-level Sentinel provides module support for J3016 level 5 compliance and an improvement in safety features

over existing autonomous vehicle architecture known by the author. It is hoped that this assurance case aids future development of Sentinel and other autonomous vehicle architectures by providing a framework for necessary level 5 functionality and safety considerations.

Chapter 6

Conclusion

6.1 Results

In this thesis, we presented a novel safety-oriented software architecture, Sentinel, for level 5 autonomous vehicles that use artificial intelligence components. We reviewed the current literature in several related domains and found that most AI-based autonomous systems focused primarily on performance and accuracy, with a clear lack of development towards the safety and fault diagnosis of those same AI components. In response to these issues, we developed the Sentinel architecture with safety and fault-tolerance features, particularly when considering the artificial intelligence mechanisms that would likely be used in level 5 autonomous vehicles. The proposed architecture was constructed through an iterative design process, taking inspiration from several existing architectures. With each design iteration, beneficial aspects were incorporated into the proposed architecture, and emergent negative properties were identified. Additionally, an existing autonomous vehicle hazard analysis was evaluated for applicability to Sentinel and was used to identify missing mit-

igation strategies and concerns that had not been addressed through previous autonomous vehicle architectures. The final version of Sentinel is presented with an accompanying assurance case, used to assert compliance with SAE J3016 level 5 recommendations as well as an improvement in safety features over known existing autonomous vehicle architectures.

6.2 Discussion

Sentinel incorporates many features that enable higher levels of autonomy with a greater focus on safety. The inherent risks of using AI techniques in safety-critical systems with ever-growing levels of autonomy are impeding the progress towards fully autonomous vehicles. Sentinel attempts to mitigate these risks by utilizing external sensory and planning information, algorithm redundancy, proactive safety planning, and a fail-safe reactive safety system.

Utilizing external sensory and planning information allows Sentinel to reduce the vehicle's blind spots, allows for cooperative planning, and gives an additional source of redundancy for sensing and planning algorithms. Automotive OEMs not only need to produce safe and reliable vehicles but need to achieve this while avoiding features that would incur excessive monetary costs. Due to these cost constraints, utilizing V2X information may provide a cheaper alternative to increased onboard sensor redundancy. Utilizing V2X information may also allow for advanced planning, whereby vehicles and infrastructure can negotiate more effective routes for all vehicles involved. A major concern with this feature, however, involves the security and validity of V2X data, as this could produce unsafe behaviour if inaccurate information is used in sensing or planning.

A separate issue addressed by Sentinel involves the use of machine learning algorithms and other artificial intelligence techniques. Sentinel proposes utilizing several AI algorithms and conventional algorithms with overlapping domains, each feeding into a fusion module that produces a probabilistic output. The goal of this is to mitigate the uncertainty associated with individual AI components by having partial redundancy spread over several other algorithms. The final result aims to represent a more accurate model than would be produced by simply using one method over another. An obvious concern with running so many algorithms in parallel is computability. However, the recent trend towards custom computing solutions for machine learning algorithms onboard autonomous vehicles and the general trend of increased computational power over time mitigates against this concern in the coming years. An important advantage of this probabilistic approach is that it avoids the alternative technique of monitoring AI behaviour in order to switch to a more deterministic approach when required. The very real problem with that approach is that it is extremely difficult to decide *when* to make that switch.

Yet another issue seen in many vehicles and autonomous systems is the lack of preventative safety. The vast majority of autonomous systems use a normal operational mode and a fail-safe mode if unsafe conditions are detected. While this is an important feature for unforeseen circumstances, the avoidance of unsafe scenarios may be just as important for vehicles aiming to supplant human capabilities. As such, Sentinel makes use of predictive planning to refine a set of safe routes. While predictive planning offers the avoidance of unsafe trajectories, the major concerns involve the computability and accuracy of the predictive models. The former concern was discussed in the previous paragraph, yet the latter concern may be more difficult to pin down. Current

techniques involve applying linear dynamic models to external agents to predict their paths. Insufficient sensory information, incorrect trajectory models or non-linear behaviour may cause issues for predictive planning. Much work has been done on vehicle tracking and pedestrian tracking, but additional work may be needed for rare examples outside these domains.

A common concern with probabilistic logic and associated statistical methods is how accurate the results are when compared to reality. If the results are inaccurate then poor decisions may be made, and in the context of this work may result in dangerous vehicle actions being taken. While this concern is certainly valid, the broad scope of machine learning algorithms being used, the assistance of conventional algorithms and external data sources being used will likely provide much more accurate results than any singular method would on its own. In addition to this, the scope of the problems these algorithms are intended to solve is often of a sufficiently high abstraction that statistical methods generalize well.

Aside from concerns derived from the architectural design, another concern exists stemming from the evaluation of this work. Due to the scope of the proposed architecture, the evaluation process was restricted to a qualitative approach. The lack of concrete quantitative results signals that the proposed architecture must be further evaluated to determine the efficacy of the beneficial features and to determine the severity of detrimental properties.

This work provides several contributions to the field of autonomous vehicles and artificial intelligence, primarily towards improving safety in both domains. A literature review of artificial intelligence safety concerns, mitigation strategies, and existing software architectures applicable to fully autonomous vehicles was conducted. Stemming from this research the Sentinel

architecture was constructed to address safety concerns intrinsic to artificial intelligence methods being used in fully autonomous vehicles. The architecture combined mitigation strategies from several applicable architectures, utilized an existing hazard analysis to guide feature incorporation, and was formulated in a manner such that the resulting framework provided mechanisms for each recommendation in the SAE J3016 recommendations for level 5 autonomous vehicles. To give further credence to the architecture, an assurance case was constructed to demonstrate J3016 compliance and improvement in safety over known autonomous vehicle architectures.

6.3 Future Research

Level 5 autonomy is still in the early phases of research and development, where this work aims to contribute to the foundation. Since extensive additional research needs to be conducted for level 5 autonomy and artificial intelligence safety a quantitative evaluation of Sentinel is not suitable at this stage. Many of the techniques and components discussed in Sentinel need to be evaluated for their efficacy, where the Sentinel architecture may provide a framework for combined evaluation rather than in isolation. As such, it is suggested that future evaluation of the Sentinel architecture via simulation should be conducted to determine the efficacy of the proposed safety systems and their relation to the AI components used.

Several directions involving the evaluation of the architecture with respect to AI components used are apparent. Determining the efficacy of utilizing conventional algorithms and overlapping AI algorithms to guide the exploration process of RL algorithms is one potential avenue. Upon completing an eval-

uation in simulation, further evaluations may be conducted on scale model vehicles, eventually leading to an evaluation on full-scale autonomous vehicles.

A suggested core area of investigation involves the proposed proactive safety system and predictive planning. Simulation techniques for predictive planning must be investigated for computability and validity with respect to real-world behaviour before such systems can be incorporated into full-scale vehicles. The vast majority of current architectures solely make use of a sense-plan-act system and a fail-safe system, with little predictive features in either. Human cognition allows for mental playthroughs of potential scenarios and acts as a mechanism to avoid acting in a way that would lead to near-term unsafe conditions. Should autonomous vehicles aim to exceed human-level performance under a variety of scenarios, predictive planning and avoidance of future hazards are undoubtedly necessary features and provide many opportunities for further research.

A separate investigation may be centred around the evaluation of the V2X interface to supply additional sensory or planning data to the ego vehicle. Questions surrounding multi-agent perceptual schemas, planning negotiation between agents and infrastructure, and overall security of utilizing external agent data are open to investigation. While current work suggests that incorporating sensory and planning information from external agents has the potential for benefits, the types of sensory and planning information being communicated and methods to determine the validity of the information being received are still an active field of research.

While substantial research must be conducted to determine the efficacy of the proposed architecture, the realization of state-of-the-art techniques for autonomous vehicle safety into a unified architecture provides future researchers

with a foundation to expand upon.

Bibliography

[1] ISO/TC 22/SC 32/WG 8. *ISO 26262 2nd Ed. (FDIS) Road Vehicles - Functional Safety Parts 1-12*. Standard. Geneva, CH: International Organization for Standardization, Dec. 2018 (cit. on p. 4).

[2] Dario Amodei et al. "Concrete Problems in AI Safety". In: *arXiv e-prints*, arXiv:1606.06565v2 (July 2016). arXiv: 1606.06565v2 [cs.AI] (cit. on pp. 20–22).

[3] Edmond Awad et al. "The moral machine experiment". In: *Nature* 563 (2018), p. 59. ISSN: 1476-4687 (cit. on p. 12).

[4] Jie Cai et al. "Feature selection in machine learning: A new perspective". In: *Neurocomputing* 300 (July 2018), pp. 70–79. ISSN: 0925-2312 (cit. on p. 17).

[5] Julián Caro Linares, Antonio Barrientos, and Enric Mayas Márquez. "Hybrid Bio-Inspired Architecture for Walking Robots Through Central Pattern Generators Using Open Source FPGAs". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'18)*. Madrid, ES, Oct. 2018, pp. 7071–7076. ISBN: 978-1-5386-8094-0 (cit. on p. 28).

[6] Ankush Desai et al. "SOTER: A Runtime Assurance Framework for Programming Safe Robotics Systems". In: *arXiv e-prints*, arXiv:1808.07921v3 (Apr. 2019). arXiv: 1808.07921v3 [cs.RO] (cit. on pp. 24, 25, 33).

[7] Kavya Prabha Divakarla, Ali Emadi, and Saiedeh Razavi. "A Cognitive Advanced Driver Assistance Systems Architecture for Autonomous-Capable Electrified Vehicles". In: *IEEE Transactions on Transportation Electrification* 5.1 (Mar. 2019), pp. 48–58. ISSN: 2332-7782 (cit. on pp. 31, 35).

[8] Pedro Domingos. "A Few Useful Things to Know About Machine Learning". In: *Communications of the ACM* 55.10 (Oct. 2012), 78—87. ISSN: 0001-0782 (cit. on pp. 16, 18).

[9] Alexey Dosovitskiy et al. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning (CoRL'17)*. Ed. by Sergey Levine, Vincent Vanhoucke, and Ken Goldberg. Vol. 78. Proceedings of Machine Learning Research. Mountain View, CA, USA, Nov. 2017, pp. 1–16 (cit. on p. 20).

[10] José Faria. "Machine Learning Safety: An Overview". In: *Safety-critical Systems Symposium 2018 (SSS'18)*. York, UK, Feb. 2018, pp. 4–18 (cit. on pp. 16–22).

[11] Christofer Fellicious. "Transfer Learning and Organic Computing for Autonomous Vehicles". In: *arXiv e-prints*, arXiv:1808.05443v1 (Aug. 2018). arXiv: 1808.05443v1 [cs.LG] (cit. on p. 17).

[12] Lex Fridman et al. "Arguing Machines: Human Supervision of Black Box AI Systems That Make Life-Critical Decisions". In: *arXiv e-prints*, arXiv:1710.04459 (Oct. 2017). arXiv: 1710.04459 [cs.AI] (cit. on p. 23).

[13] Javier García and Fernando Fernández. "A Comprehensive Survey on Safe Reinforcement Learning". In: *Journal of Machine Learning Research* 16.42 (Aug. 2015), pp. 1437–1480. ISSN: 1533-7928 (cit. on pp. 20, 21).

[14] Fernando Garcia et al. "Sensor fusion methodology for vehicle detection". In: *IEEE Intelligent Transportation Systems Magazine* 9.1 (2017), pp. 123–133. ISSN: 1941-1197 (cit. on p. 61).

[15] Gaspar L. Gil Gómez et al. "Machine learning to classify and predict objective and subjective assessments of vehicle dynamics: the case of steering feel". In: *Vehicle system dynamics* 56.1 (2018), pp. 150–171. ISSN: 1744-5159 (cit. on p. 64).

[16] BMW Group. "Autonomous Driving at the BMW Group: #NEXTGen 19". In: *PressClub Global* (June 2019). URL: http://press.bmwgroup.com/xYg6p (cit. on p. 8).

[17] Gültekin Gündüz et al. "Driving pattern fusion using dempster-shafer theory for fuzzy driving risk level assessment". In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. Los Angeles, CA, USA, June 2017, pp. 595–599. ISBN: 978-1-5090-4804-5 (cit. on p. 61).

[18] Laurens Hobert et al. "Enhancements of V2X communication in support of cooperative autonomous driving". In: *IEEE Communications Magazine* 53.12 (Dec. 2015), pp. 64–70. ISSN: 1558-1896 (cit. on p. 62).

[19] SAE International. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. June 2018 (cit. on p. 8).

[20] Christos Katrakazas et al. "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions". In: *Transportation Research Part C: Emerging Technologies* 60 (Nov. 2015), pp. 416–442. ISSN: 0968-090X (cit. on p. 63).

[21] Coy K. Kouba, Deborah Elaine Buscher, and Joseph L. Busa. *The X-38 Spacecraft Fault-Tolerant Avionics System*. Tech. rep. Houston, TX, USA: National Aeronautics and Space Administration, Johnson Space Center, Aug. 2003 (cit. on p. 11).

[22] Mateusz Krzvsztoń. "Adaptive Supervisor: Method of Reinforcement Learning Fault Elimination by Application of Supervised Learning". In: *2018 Federated Conference on Computer Science and Information Systems (FedCSIS'18)*. Poznan, PL, July 2018, pp. 139–143. ISBN: 978-83-949419-5-6 (cit. on p. 26).

[23] Jake Lever, Martin Krzywinski, and Naomi Altman. "Points of Significance: Regularization". In: *Nature Methods* 13.10 (2016). ISSN: 1548-7105 (cit. on p. 19).

[24] Piergiuseppe Mallozzi, Patrizio Pelliccione, and Claudio Menghi. "Keeping intelligence under control". In: *Proceedings of the 1st International Workshop on Software Engineering for Cognitive Services (SE4COG'18)*. Gothenburg, SE, May 2018, pp. 37–40 (cit. on pp. 16, 21).

[25] Clara Marina Martinez et al. "Driving style recognition for intelligent vehicle control and advanced driver assistance: A survey". In: *IEEE Transactions on Intelligent Transportation Systems* 19.3 (2017), pp. 666–676. ISSN: 1558-0016 (cit. on p. 64).

[26] Rowan McAllister et al. "Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*. Melbourne, AU, 2017, pp. 4745–4753 (cit. on pp. 61, 69).

[27] Caroline Bianca Santos Tancredi Molina et al. "Assuring Fully Autonomous Vehicles Safety by Design: The Autonomous Vehicle Control (AVC) Module Strategy". In: *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. Denver, CO, USA, June 2017, pp. 16–21. ISBN: 978-1-5386-2272-8 (cit. on p. 33).

[28] Christian Müller-Schloer and Sven Tomforde. *Organic Computing – Technical Systems for Survival in the Real World*. Autonomic Systems. Birkhäuser, Dec. 2017 (cit. on pp. 27, 29, 36, 56).

[29] Andy Pasztor et al. "How Boeing's 737 MAX Failed". In: *The Wall Street Journal* (Mar. 2019) (cit. on p. 11).

[30] Bardo Peters and Klaus Rick. "I Spy: V2X Technology Can Provide Information about Objects and Conditions Although They May be Obscured". In: *Vision Zero International* (Jan. 2017). ISSN: 2014-4447 (cit. on p. 60).

[31] David J. Rinehart, John C. Knight, and Jonathan Rowanhill. *Understanding what it Means for Assurance Cases to "work"*. Tech. rep. Hampton, VA, USA: National Aeronautics and Space Administration, Langley Research Center, Apr. 2017 (cit. on pp. 81, 82).

[32] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. "An Analysis of ISO 26262: Machine Learning and Safety in Automotive Software". In:

WCX World Congress Experience. SAE International, Apr. 2018 (cit. on pp. 24, 34).

[33] D. Sculley et al. "Machine Learning: The High Interest Credit Card of Technical Debt". In: *SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)*. 2014 (cit. on pp. 17, 18).

[34] Alexandru Constantin Serban, Erik Poll, and Joost Visser. "A Standard Driven Software Architecture for Fully Autonomous Vehicles". In: *2018 IEEE International Conference on Software Architecture Companion (ICSA-C)*. Seattle, WA, USA, Apr. 2018, pp. 120–127. ISBN: 978-1-5386-6585-5 (cit. on p. 35).

[35] Syed Asim Shah. "Safe-AV: A Fault Tolerant Safety Architecture for Autonomous Vehicles". MA thesis. Hamilton, CA: McMaster University, 2019 (cit. on pp. 37, 38, 41, 42, 47, 90).

[36] Yanjun Shi et al. "A 5G-V2X Based Collaborative Motion Planning for Autonomous Industrial Vehicles at Road Intersections". In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC'18)*. Miyazaki, JP, Oct. 2018, pp. 3744–3748. ISBN: 978-1-5386-6650-0 (cit. on p. 62).

[37] Masamichi Shimosaka et al. "Predicting driving behavior using inverse reinforcement learning with multiple reward functions towards environmental diversity". In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. Seoul, KR, June 2015, pp. 567–572. ISBN: 978-1-4673-7266-4 (cit. on p. 20).

[38] Donald F. Specht. "Probabilistic neural networks". In: *Neural Networks* 3.1 (1990), pp. 109–118. ISSN: 0893-6080 (cit. on p. 61).

[39] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15 (June 2014), pp. 1929–1958. ISSN: 1533-7928 (cit. on p. 19).

[40] Nassim Nicholas Taleb. *The Black Swan: The Impact of the Highly Improbable*. Second Edition. Penguin Random House, 2010. ISBN: 978-0812973815 (cit. on p. 16).

[41] Wilfredo Torres-Pomales. *Software fault tolerance: A tutorial*. Tech. rep. Hampton, VA, USA: National Aeronautics and Space Administration, Langley Research Center, Oct. 2000 (cit. on p. 11).

[42] Matthias Traub, Alexander Maier, and Kai L. Barbehön. "Future Automotive Architecture and the Impact of IT Trends". In: *IEEE Software* 34.3 (May 2017), pp. 27–32. ISSN: 1937-4194 (cit. on p. 69).

[43] Kush R. Varshney. "Engineering Safety in Machine Learning". In: *2016 Information Theory and Applications Workshop (ITA'16)*. San Diego, CA, USA, Jan. 2016, pp. 1–5. ISBN: 978-1-5090-2529-9 (cit. on pp. 16, 19).

[44] Jian Wang et al. "A Survey of Vehicle to Everything (V2X) Testing". In: *Sensors* 19.2 (2019). ISSN: 1424-8220 (cit. on p. 66).

[45] Alan Wassyng et al. "New Standards for Trustworthy Cyber-Physical Systems". In: *Trustworthy Cyber-Physical Systems Engineering*. Ed. by Alexander Romanovsky and Fuyuki Ishikawa. New York, NY, USA: CRC Press, Oct. 2016, pp. 337–368. ISBN: 9781315369105 (cit. on p. 81).

[46] E-GAS Workgroup. *Standardized E-Gas Monitoring Concept for Gasoline and Diesel Engine Control Units*. Tech. rep. Version 6.0. July 2015 (cit. on p. 39).